Theoretically Optimal Datalog Rewritings for *OWL 2 QL* Ontology-Mediated Queries

M. Bienvenu[1], S. Kikot[2], R. Kontchakov[2], V. Podolskii[3], and M. Zakharyaschev[2]

[1] CNRS & University of Montpellier, France (meghyn@lirmm.fr)
[2] Birkbeck, University of London, U.K. ({kikot,roman,michael}@dcs.bbk.ac.uk)
[3] Steklov Mathematical Institute, Moscow, Russia (podolskii@mi.ras.ru)

Abstract. We show that, for *OWL 2 QL* ontology-mediated queries with (*i*) ontologies of bounded depth and conjunctive queries of bounded treewidth, (*ii*) ontologies of bounded depth and bounded-leaf tree-shaped conjunctive queries, and (*iii*) arbitrary ontologies and bounded-leaf tree-shaped conjunctive queries, one can construct and evaluate nonrecursive datalog rewritings by, respectively, LOGCFL, NL and LOGCFL algorithms, which matches the optimal combined complexity.

1 Introduction

Ontology-based data access (OBDA) via query rewriting [18] reduces the problem of finding answers to conjunctive queries (CQs) mediated by *OWL 2 QL* ontologies to standard database query answering. The question we are concerned with here is whether this reduction is optimal with respect to the combined complexity of query evaluation. Figure 1 (a) summarises what is known about the size of positive existential (PE), nonrecursive datalog (NDL) and first-order (FO) rewritings of *OWL 2 QL* ontology-mediated queries (OMQs) depending on the existential depth of their ontologies and the shape of their CQs [13, 9, 12, 3]. Figure 1 (b) shows the combined complexity of OMQ evaluation for the corresponding classes of OMQs [5, 14, 12, 3]. Thus, we see, for example, that PE-rewritings for OMQs with ontologies of bounded depth and CQs of bounded treewidth can be of super-polynomial size, and so not evaluable in polynomial time, while the evaluation problem for these OMQs is decidable in LOGCFL $\subseteq$ P. On the other hand, the OMQs in this class enjoy polynomial-size NDL-rewritings. However, these rewritings were defined using an argument from circuit complexity [3], and it has been unclear whether they can be constructed and evaluated in LOGCFL. The same concerns the class of OMQs with ontologies of bounded depth and bounded-leaf tree-shaped queries, which can be evaluated in NL, and the class of OMQs with arbitrary ontologies and bounded-leaf tree-shaped queries, which can be evaluated in LOGCFL.

In this paper, we consider OMQs in these three classes and construct NDL-rewritings that are theoretically optimal in the sense that the rewriting and evaluation can be carried out by algorithms of optimal combined complexity, that is, from the complexity classes LOGCFL, NL and LOGCFL, respectively. Such algorithms are known to be space efficient and highly parallelisable. We compared our optimal NDL rewritings with those produced by query rewriting engines Clipper [8] and Rapid [6], using a sequence of OMQs with linear CQs and a fixed ontology of depth 1.

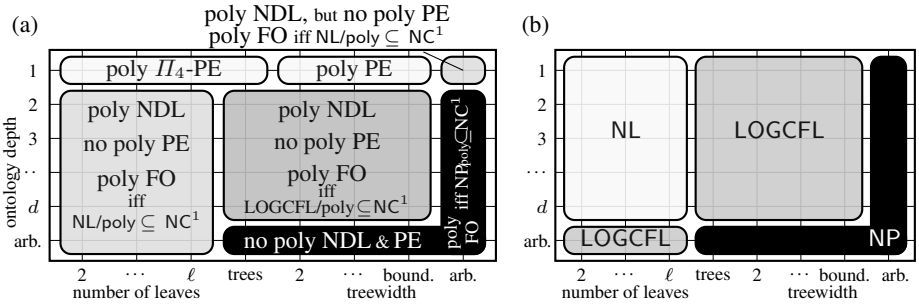


Fig. 1. (a) Size of OMQ rewritings; (b) combined complexity of OMQ evaluation.

2 Preliminaries

We give *OWL 2 QL* in the DL syntax with *individual names* a_i, *concept names* A_i, and *role names* P_i ($i \geq 1$). *Roles* R and *basic concepts* B are defined by

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

A *TBox*, $\mathcal{T}$, is a finite set of inclusions of the form

$$B_1 \sqsubseteq B_2, \qquad B_1 \sqcap B_2 \sqsubseteq \bot, \qquad R_1 \sqsubseteq R_2, \qquad R_1 \sqcap R_2 \sqsubseteq \bot.$$

An *ABox*, $\mathcal{A}$, is a finite set of atoms of the form $A_k(a_i)$ or $P_k(a_i, a_j)$. We denote by $\mathsf{ind}(\mathcal{A})$ the set of individual names in $\mathcal{A}$, and by $\boldsymbol{R}_\mathcal{T}$ the set of role names occurring in $\mathcal{T}$ and their inverses. We use $A \equiv B$ for $A \sqsubseteq B$ and $B \sqsubseteq A$. The semantics for *OWL 2 QL* is defined in the usual way based on interpretations $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ [2].

For every role $R \in \boldsymbol{R}_\mathcal{T}$, we take a fresh concept name A_R and add $A_R \equiv \exists R$ to $\mathcal{T}$. The resulting TBox is said to be in *normal form*, and we assume, without loss of generality, that all our TBoxes are in normal form. The subsumption relation induced by $\mathcal{T}$ is denoted by $\sqsubseteq_\mathcal{T}$: we write $S_1 \sqsubseteq_\mathcal{T} S_2$ if $\mathcal{T} \models S_1 \sqsubseteq S_2$, where S_1, S_2 are both either concepts or roles. We write $R(a, b) \in \mathcal{A}$ if $P(a, b) \in \mathcal{A}$ and $R = P$, or $P(b, a) \in \mathcal{A}$ and $R = P^-$; we also write $(\exists R)(a) \in \mathcal{A}$ if $R(a, b) \in \mathcal{A}$ for some b. An ABox $\mathcal{A}$ is called *H-complete with respect to* $\mathcal{T}$ in case

$$P(a, b) \in \mathcal{A} \;\text{ if }\; R(a, b) \in \mathcal{A}, \text{ for roles } P \text{ and } R \text{ with } R \sqsubseteq_\mathcal{T} P,$$
$$A(a) \in \mathcal{A} \;\text{ if }\; B(a) \in \mathcal{A}, \text{ for a concept name } A \text{ and basic concept } B \text{ with } B \sqsubseteq_\mathcal{T} A.$$

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 $A_k(z_1)$ or $P_k(z_1, z_2)$ with $z_i \in \boldsymbol{x} \cup \boldsymbol{y}$ (without loss of generality, we assume that CQs do not contain constants). We denote by $\mathsf{var}(\boldsymbol{q})$ the variables $\boldsymbol{x} \cup \boldsymbol{y}$ of $\boldsymbol{q}$ and by $\mathsf{avar}(\boldsymbol{q})$ the *answer variables* $\boldsymbol{x}$. An *ontology-mediated query* (OMQ) is a pair $\boldsymbol{Q}(\boldsymbol{x}) = (\mathcal{T}, \boldsymbol{q}(\boldsymbol{x}))$, where $\mathcal{T}$ is a TBox and $\boldsymbol{q}(\boldsymbol{x})$ a CQ. A tuple $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{A})$ is a *certain answer* to $\boldsymbol{Q}(\boldsymbol{x})$ over an ABox $\mathcal{A}$ if $\mathcal{I} \models \boldsymbol{q}(\boldsymbol{a})$ for all models $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$; in this case we write $\mathcal{T}, \mathcal{A} \models \boldsymbol{q}(\boldsymbol{a})$. If $\boldsymbol{x} = \emptyset$, then a certain answer to $\boldsymbol{Q}$ over $\mathcal{A}$ is 'yes' if $\mathcal{T}, \mathcal{A} \models \boldsymbol{q}$ and 'no' otherwise. We often regard a CQ $\boldsymbol{q}$ as the set of its atoms.

Every consistent *OWL 2 QL knowledge base* (KB) $(\mathcal{T}, \mathcal{A})$ has a *canonical model* $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ with the property that $\mathcal{T}, \mathcal{A} \models \boldsymbol{q}(\boldsymbol{a})$ iff $\mathcal{C}_{\mathcal{T},\mathcal{A}} \models \boldsymbol{q}(\boldsymbol{a})$, for any CQ $\boldsymbol{q}$ and any $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{A})$. Thus, CQ answering in *OWL 2 QL* amounts to finding a homomorphism from the given CQ into the canonical model. Informally, $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ is obtained from $\mathcal{A}$ by repeatedly applying the axioms in $\mathcal{T}$, introducing fresh elements as needed to serve as witnesses for the existential quantifiers. According to the standard construction (cf. [16]), the domain $\Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ of $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ consists of words of the form $aR_1 \dots R_n$ $(n \geq 0)$ with $a \in \mathsf{ind}(\mathcal{A})$ and $R_1 \dots R_n \in \boldsymbol{R}^*_{\mathcal{T}}$ such that (*i*) $\mathcal{T}, \mathcal{A} \models \exists R_1(a)$ and (*ii*) $\exists R_i^- \sqsubseteq_{\mathcal{T}} \exists R_{i+1}$ and $R_i^- \not\sqsubseteq_{\mathcal{T}} R_{i+1}$, for $1 \leq i < n$. We let $\mathbf{W}_{\mathcal{T}}$ consist of all words $R_1 \dots R_n \in \boldsymbol{R}^*_{\mathcal{T}}$ satisfying condition (*ii*). A TBox $\mathcal{T}$ is *of depth* ω if $\mathbf{W}_{\mathcal{T}}$ is infinite, and *of depth* $d < \omega$, if d is the maximum length of the words in $\mathbf{W}_{\mathcal{T}}$.

A *datalog program*, Π, is a finite set of Horn clauses $\forall \boldsymbol{z}\, (\gamma_0 \leftarrow \gamma_1 \wedge \dots \wedge \gamma_m)$, where each γ_i is an atom $S(\boldsymbol{y})$ with $\boldsymbol{y} \subseteq \boldsymbol{z}$ or an equality $(z = z')$ with $z, z' \in \boldsymbol{z}$. (As usual, when writing clauses, we omit $\forall \boldsymbol{z}$.) The atom γ_0 is the *head* of the clause, and $\gamma_1, \dots, \gamma_m$ its *body*. All variables in the head must also occur in the body, and $=$ can only occur in the body. The predicates in the heads of clauses in Π are *IDB predicates*, the rest (including $=$) *EDB predicates*. A predicate S *depends* on S' in Π if Π has a clause with S in the head and S' in the body; Π is a *nonrecursive datalog* (NDL) *program* if the (directed) *dependence graph* of the dependence relation is acyclic.

An *NDL query* is a pair $(\Pi, G(\boldsymbol{x}))$, where Π is an NDL program and $G(\boldsymbol{x})$ a predicate. A tuple $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{A})$ is an *answer to* $(\Pi, G(\boldsymbol{x}))$ *over* an ABox $\mathcal{A}$ if $G(\boldsymbol{a})$ holds in the first-order model with domain $\mathsf{ind}(\mathcal{A})$ obtained by closing $\mathcal{A}$ under the clauses in Π; in this case we write $\Pi, \mathcal{A} \models G(\boldsymbol{a})$. The problem of checking whether $\boldsymbol{a}$ is an answer to $(\Pi, G(\boldsymbol{x}))$ over $\mathcal{A}$ is called the *query evaluation problem*. The *arity of* Π is the maximal arity, $\mathsf{r}(\Pi)$, of predicates in Π. The *depth* of $(\Pi, G(\boldsymbol{x}))$ is the length, $\mathsf{d}(\Pi, G)$, of the longest directed path in the dependence graph for Π starting from G. NDL queries are *equivalent* if they have exactly the same answers over any ABox.

An NDL query $(\Pi, G(\boldsymbol{x}))$ is an *NDL-rewriting of an OMQ* $\boldsymbol{Q}(\boldsymbol{x}) = (\mathcal{T}, \boldsymbol{q}(\boldsymbol{x}))$ *over H-complete ABoxes* in case $\mathcal{T}, \mathcal{A} \models \boldsymbol{q}(\boldsymbol{a})$ iff $\Pi, \mathcal{A} \models G(\boldsymbol{a})$, for any H-complete ABox $\mathcal{A}$ and any tuple $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{A})$. Rewritings *over arbitrary ABoxes* are defined by dropping the condition that the ABoxes are H-complete. Let $(\Pi, G(\boldsymbol{x}))$ be an NDL-rewriting of $\boldsymbol{Q}(\boldsymbol{x})$ over H-complete ABoxes. Denote by Π^* the result of replacing each predicate S in Π with a fresh predicate S^* and adding the clauses $A^*(x) \leftarrow B'(x)$, for $B \sqsubseteq_{\mathcal{T}} A$, and $P^*(x, y) \leftarrow R'(x, y)$, for $R \sqsubseteq_{\mathcal{T}} P$, where $B'(x)$ and $R'(x, y)$ are the obvious first-order translations of B and R (for example, $B'(x) = \exists y\, R(x, y)$ if $B = \exists R$). It is easy to see that $(\Pi^*, G^*(\boldsymbol{x}))$ is an NDL-rewriting of $\boldsymbol{Q}(\boldsymbol{x})$ over arbitrary ABoxes.

It is well-known [4] that, without loss of generality, we can only consider NDL-rewritings of OMQs $(\mathcal{T}, \boldsymbol{q}(\boldsymbol{x}))$ over ABoxes $\mathcal{A}$ that are *consistent* with $\mathcal{T}$.

We call an NDL query $(\Pi, G(x_1, \dots, x_n))$ *ordered* if each of its IDB predicates S comes with fixed variables $x_{i_1}, \dots, x_{i_k}$ $(1 \leq i_1 < \dots < i_k \leq n)$, called the *parameters of* S, such that (*i*) every occurrence of S in Π is of the form $S(y_1, \dots, y_m, x_{i_1}, \dots, x_{i_k})$, (*ii*) the x_i are the parameters of G, and (*iii*) if $\boldsymbol{x}'$ are all the parameters in the body of a clause, then the head has $\boldsymbol{x}'$ among its parameters. The *width* $\mathsf{w}(\Pi, G)$ of an ordered (Π, G) is the maximal number of non-parameter variables in a clause of Π. All our NDL-rewritings in Secs. 4–6 are ordered, so we now only consider ordered NDL queries.

3 NL and LOGCFL Fragments of Nonrecursive Datalog

In this section, we identify two classes of (ordered) NDL queries with the evaluation problem in the complexity classes NL and LOGCFL for combined complexity. Recall [1] that an NDL program is called *linear* if the body of its every clause contains at most one IDB predicate (remember that equality is an EDB predicate).

Theorem 1. *Fix some* $\mathsf{w} > 0$. *The combined complexity of evaluating linear NDL queries of width at most* w *is* NL*-complete.*

Proof. Let $(\Pi, G(\boldsymbol{x}))$ be a linear NDL query. Deciding whether $\Pi, \mathcal{A} \models G(\boldsymbol{a})$ is reducible to finding a path to $G(\boldsymbol{a})$ from a certain set X in the *grounding graph* $\mathfrak{G}(\Pi, \mathcal{A}, \boldsymbol{a})$ constructed as follows. The vertices of the graph are the ground atoms obtained by taking an IDB atom from Π, replacing each of its parameters by the corresponding constant from $\boldsymbol{a}$, and replacing each non-parameter variable by some constant from $\mathcal{A}$. The graph has an edge from $S(\boldsymbol{c})$ to $S'(\boldsymbol{c}')$ iff the grounding of Π contains a clause $S'(\boldsymbol{c}') \leftarrow S(\boldsymbol{c}) \land E_1(\boldsymbol{e}_1) \land \dots \land E_k(\boldsymbol{e}_k)$ with $E_j(\boldsymbol{e}_j) \in \mathcal{A}$, for $1 \leq j \leq k$ (we assume that $(c = c) \in \mathcal{A}$). The set X consists of all vertices $S(\boldsymbol{c})$ with IDB predicates S being of in-degree 0 in the dependency graph of Π for which there is a clause $S(\boldsymbol{c}) \leftarrow E_1(\boldsymbol{e}_1) \land \dots \land E_k(\boldsymbol{e}_k)$ in the grounding of Π with $E_j(\boldsymbol{e}_j) \in \mathcal{A}$ $(1 \leq j \leq k)$. Bounding the width of (Π, G) ensures that $\mathfrak{G}(\Pi, \mathcal{A}, \boldsymbol{a})$ is of polynomial size and can be constructed by a deterministic Turing machine with separate input, write-once output and logarithmic-size working tapes. ❑

The transformation of NDL-rewritings over H-complete ABoxes into rewritings for arbitrary ABoxes in Section 2 does not preserve linearity. However, we can still show that it suffices to consider the H-complete case:

Lemma 2. *For any fixed* $\mathsf{w} > 0$*, there is an* L^{NL}*-transducer that, given a linear NDL-rewriting of an OMQ* $\boldsymbol{Q}(\boldsymbol{x})$ *over H-complete ABoxes that is of width at most* w*, computes a linear NDL-rewriting of* $\boldsymbol{Q}(\boldsymbol{x})$ *over arbitrary ABoxes whose width is at most* $\mathsf{w} + 1$.

The complexity class LOGCFL can be defined in terms of *nondeterministic auxiliary pushdown automata* (NAuxPDAs) [7], which are nondeterministic Turing machines with an additional work tape constrained to operate as a pushdown store. Sudborough [19] proved that LOGCFL coincides with the class of problems that are solved by NAuxPDAs running in logarithmic space and polynomial time (the space on the pushdown tape is not subject to the logarithmic bound).

We call an NDL query (Π, G) *skinny* if the body of any clause in Π has ≤ 2 atoms.

Lemma 3. *For any skinny NDL query* $(\Pi, G(\boldsymbol{x}))$ *and ABox* $\mathcal{A}$*, query evaluation can be done by an NAuxPDA in space* $\log |\Pi| + \mathsf{w}(\Pi, G) \cdot \log |\mathcal{A}|$ *and time* $2^{O(\mathsf{d}(\Pi, G))}$.

Proof. Let $\Pi^{\boldsymbol{a}}_{\mathcal{A}}$ be the set of ground clauses obtained by first replacing each parameter in Π by the corresponding constant from $\boldsymbol{a}$, and then performing the standard grounding of Π using the constants from $\mathcal{A}$. Consider the monotone Boolean circuit $\boldsymbol{C}(\Pi, \mathcal{A}, \boldsymbol{a})$ constructed as follows. The output of $\boldsymbol{C}(\Pi, \mathcal{A}, \boldsymbol{a})$ is $G(\boldsymbol{a})$. For every atom γ occurring in the head of a clause in $\Pi^{\boldsymbol{a}}_{\mathcal{A}}$, we take an OR-gate whose output is γ and inputs are the

bodies of the clauses with head γ; for every such body, we take an AND-gate whose inputs are the atoms in the body. We set an input gate γ to 1 iff $\gamma \in \mathcal{A}$. Clearly, $\boldsymbol{C}(\Pi, \mathcal{A}, \boldsymbol{a})$ is a semi-unbounded fan-in circuit (where OR-gates have arbitrarily many inputs, and AND-gates two inputs) with $O(|\Pi| \cdot |\mathcal{A}|^{\mathsf{w}(\Pi,G)})$ gates and depth $O(\mathsf{d}(\Pi, G))$. It is known that the nonuniform analog of LOGCFL can be defined using families of semi-unbounded fan-in circuits of polynomial size and logarithmic depth. Moreover, there is an algorithm that, given such a circuit $\boldsymbol{C}$, computes the output using an NAuxPDA in logarithmic space in the size of $\boldsymbol{C}$ and exponential time in the depth of $\boldsymbol{C}$ [20, pp. 392–397]. Observing that $\boldsymbol{C}(\Pi, \mathcal{A}, \boldsymbol{a})$ can be computed by a deterministic logspace Turing machine, we conclude that the query evaluation problem can be solved by an NAuxPDA in space $\log |\Pi| + \mathsf{w}(\Pi, G) \cdot \log |\mathcal{A}|$ and time $2^{O(\mathsf{d}(\Pi,G))}$. ❑

A function ν from the predicate names in Π to $\mathbb{N}$ is a *weight function for* an NDL-query $(\Pi, G(\boldsymbol{x}))$ if $\nu(P) > 0$, for any IDB P in Π, and $\nu(P) \geq \nu(Q_1) + \cdots + \nu(Q_n)$, for any $P(\boldsymbol{z}) \leftarrow Q_1(\boldsymbol{z}_1) \wedge \cdots \wedge Q_n(\boldsymbol{z}_n)$ in Π.

Lemma 4. *If $(\Pi, G(\boldsymbol{x}))$ has a weight function ν, then it is equivalent to a skinny NDL query $(\Pi', G(\boldsymbol{x}))$ such that $|\Pi'|$ is polynomial in $|\Pi|$, $\mathsf{d}(\Pi', G) \leq \mathsf{d}(\Pi, G) + \log \nu(G)$ and $\mathsf{w}(\Pi', G) \leq \mathsf{w}(\Pi, G)$.*

Proof. The proof is by induction on $\mathsf{d}(\Pi, G)$. If $\mathsf{d}(\Pi, G) = 0$, we take $\Pi' = \Pi$. Suppose Π contains a clause ψ of the form $G(\boldsymbol{z}) \leftarrow P_1(\boldsymbol{z}_1) \wedge \cdots \wedge P_k(\boldsymbol{z}_k)$ and, for each $1 \leq j \leq k$, we have an NDL query (Π'_{P_j}, P_j) which is equivalent to (Π, P_j) and such that

$$\mathsf{d}(\Pi'_{P_j}, P_j) \;\leq\; \mathsf{d}(\Pi_{P_j}, P_j) + \log \nu(P_j) \;\leq\; \mathsf{d}(\Pi, G) - 1 + \log \nu(P_j). \tag{1}$$

We construct the Huffman tree [11] for the alphabet $\{1, \dots, k\}$, where the frequency of j is $\nu(P_j)/\nu(G)$ (by definition, $\nu(G) > 0$). The Huffman tree is binary and has k leaves, denoted $1, \dots, k$, and $k - 1$ internal nodes (including the root, g), and the length of the path from g to any leaf j at most $\lceil \log(\nu(G)/\nu(P_j)) \rceil$. For each internal node v of the tree (but the root), we take a predicate $P_v(\boldsymbol{z}_v)$, where $\boldsymbol{z}_v$ is the union of $\boldsymbol{z}_u$ for all descendants u of v; for the root g, we take $P_g(\boldsymbol{z}_g) = G(\boldsymbol{z})$. Let Π'_ψ be the extension of the union of Π'_{P_j}, for $1 \leq j \leq k$, with clauses $P_v(\boldsymbol{z}_v) \leftarrow P_{u_1}(\boldsymbol{z}_{u_1}) \wedge P_{u_2}(\boldsymbol{z}_{u_2})$, for each v with immediate successors u_1 and u_2. The number of the new clauses is $k - 1$. Consider the NDL query $(\Pi'_\psi, G(\boldsymbol{z}))$. By (1), we have:

$$\begin{aligned}\mathsf{d}(\Pi'_\psi, G) \leq \max_j \{\lceil \log(\nu(G)/\nu(P_j)) \rceil + \mathsf{d}(\Pi'_{P_j}, P_j)\} \leq \\ \max_j \{\log(\nu(G)/\nu(P_j)) + \mathsf{d}(\Pi, G) + \log \nu(P_j)\} = \log \nu(G) + \mathsf{d}(\Pi, G).\end{aligned}$$

Let Π' be the result of applying this transformation to each clause in Π with head $G(\boldsymbol{z})$. It is readily seen that (Π', G) is as required; in particular, $|\Pi'| = O(|\Pi|^2)$. ❑

Theorem 5. *Fix $c \geq 1$, $\mathsf{w} \geq 1$ and a polynomial p. Query evaluation for NDL queries $(\Pi, G(\boldsymbol{x}))$ with a weight function ν such that $\nu(G) \leq p(|\Pi|)$, $\mathsf{w}(\Pi, G) \leq \mathsf{w}$ and $\mathsf{d}(\Pi, G) \leq c \log \nu(G)$ is in* LOGCFL *for combined complexity.*

Proof. By Lemma 4, (Π, G) is equivalent to a skinny NDL query (Π', G') with $|\Pi'|$ polynomial in $|\Pi|$, $\mathsf{w}(\Pi', G) \leq \mathsf{w}$, and $\mathsf{d}(\Pi', G') \leq (c+1)\log \nu(G)$. By Lemma 3, query evaluation for (Π', G') over $\mathcal{A}$ is solved by an NAuxPDA in space $\log |\Pi'| + \mathsf{w}(\Pi', G) \cdot \log |\mathcal{A}| = O(\log |\Pi| + \log |\mathcal{A}|)$ and time $2^{O(\mathsf{d}(\Pi', G'))} \leq 2^{O(\log \nu(G))} = (\nu(G))^{O(1)} \leq p'(|\Pi|)$, for some polynomial p'. ❑

Corollary 6. *Suppose there is an algorithm that, given any OMQ $\boldsymbol{Q}(\boldsymbol{x})$ from some class $\mathcal{C}$, constructs its NDL-rewriting $(\Pi, G(\boldsymbol{x}))$ over H-complete ABoxes having a weight function ν with $\nu(G) \leq |\boldsymbol{Q}|$ and $\mathsf{d}(\Pi, G) \leq c \log \nu(G)$, and such that $\mathsf{w}(\Pi, G) \leq \mathsf{w}$ and $|\boldsymbol{Q}| \leq |\Pi| \leq p(|\boldsymbol{Q}|)$, for some fixed constants c, w and polynomial p. Then the evaluation problem for the NDL-rewritings $(\Pi^*, G^*(\boldsymbol{x}))$ of the OMQs in $\mathcal{C}$ over arbitrary ABoxes (defined in Section 2) is in* LOGCFL *for combined complexity.*

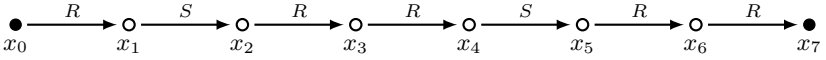
4 Bounded Treewidth CQs and Bounded-Depth TBoxes

With every CQ $\boldsymbol{q}$, we associate its *Gaifman graph* $\mathcal{G}$ whose vertices are the variables of $\boldsymbol{q}$ and edges are the pairs $\{u, v\}$ such that $P(u, v) \in \boldsymbol{q}$, for some P. We call $\boldsymbol{q}$ *tree-shaped* if $\mathcal{G}$ is a tree; $\boldsymbol{q}$ is *connected* if the graph $\mathcal{G}$ is connected. A *tree decomposition* of an undirected graph $\mathcal{G} = (V, E)$ is a pair (T, λ), where T is an (undirected) tree and λ a function from the set of nodes of T to 2^V such that the following conditions hold:

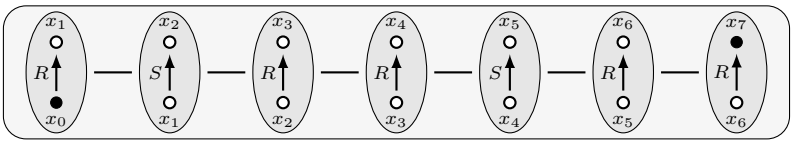
- for every $v \in V$, there exists a node t with $v \in \lambda(t)$;
- for every $e \in E$, there exists a node t with $e \subseteq \lambda(t)$;
- for every $v \in V$, the nodes $\{t \mid v \in \lambda(t)\}$ induce a connected subtree of T.

We call the set $\lambda(t) \subseteq V$ a *bag for* t. The *width* of (T, λ) is $\max_{t \in T} |\lambda(t)| - 1$. The *treewidth of a graph* $\mathcal{G}$ is the minimum width over all tree decompositions of $\mathcal{G}$. The *treewidth of a CQ* is the treewidth of its Gaifman graph.

Example 7. Consider CQ $\boldsymbol{q}(x_0, x_7)$ depicted below (black nodes are answer variables):



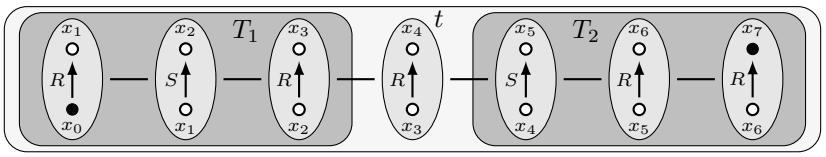
Its natural tree decomposition of treewidth 1 is based on the the chain T of 7 vertices, which are represented as bags as follows:



Fix a connected CQ $\boldsymbol{q}(\boldsymbol{x})$ and a tree decomposition (T, λ) of its Gaifman graph $\mathcal{G} = (V, E)$. Let D be a subtree of T. The *size* of D is the number of nodes in it. We call a node t of D *boundary* if T has an edge $\{t, t'\}$ with $t' \notin D$, and let the *degree* $\mathsf{deg}(D)$ of D be the number of its boundary nodes. Note that T itself is the only subtree of T of degree 0. We say that a node t *splits* D into subtrees $D_1, \dots, D_k$ if the D_i partition D without t: each node of D except t belongs to exactly one D_i.

Lemma 8 ([3]). *Let D be a subtree of T of size $m > 1$.*
If $\mathsf{deg}(D) = 2$, then there is a node t splitting D into subtrees of size $\leq m/2$ and degree ≤ 2 and, possibly, one subtree of size $< m - 1$ and degree 1.
If $\mathsf{deg}(D) \leq 1$, then there is t splitting D into subtrees of size $\leq m/2$ and degree ≤ 2.

In Example 7, t splits T into T_1 and T_2 depicted below:



We define recursively a set $\mathsf{sub}(T)$ of subtrees of T, a binary relation $\prec$ on $\mathsf{sub}(T)$ and a function σ on $\mathsf{sub}(T)$ indicating the splitting node. We begin by adding T to $\mathsf{sub}(T)$. Take $D \in \mathsf{sub}(T)$ that has not been split yet. If D is of size 1 then let $\sigma(D)$ be the only node of D. Otherwise, by Lemma 8, we find a node t in D that splits it into $D_1, \dots, D_k$. We set $\sigma(D) = t$ and, for each $1 \leq i \leq k$, add D_i to $\mathsf{sub}(T)$ and set $D_i \prec D$; then, we apply the procedure recursively to each of $D_1, \dots, D_k$. In Example 7 with t splitting T, we have $\sigma(T) = t$, $T_1 \prec T$ and $T_2 \prec T$.

For each $D \in \mathsf{sub}(T)$, we recursively define a set of atoms $\boldsymbol{q}_D$ by taking

$$\boldsymbol{q}_D \;=\; \big\{S(\boldsymbol{v}) \in \boldsymbol{q} \mid \boldsymbol{v} \subseteq \lambda(\sigma(D))\big\} \,\cup\, \bigcup\nolimits_{D' \prec D} \boldsymbol{q}_{D'}.$$

By the definition of tree decomposition, $\boldsymbol{q}_T = \boldsymbol{q}$. Denote by $\boldsymbol{x}_D$ the subset of $\mathsf{avar}(\boldsymbol{q})$ that occur in $\boldsymbol{q}_D$. In our running example, $\boldsymbol{x}_T = \{x_0, x_7\}$, $\boldsymbol{x}_{T_1} = \{x_0\}$ and $\boldsymbol{x}_{T_2} = \{x_7\}$. Denote by ∂D the union of all $\lambda(t) \cap \lambda(t')$ for a boundary node t of D and its unique neighbour t' in T *outside* D. If D is a singleton $\{d\}$, then ∂D consists of those variables in $\lambda(d)$ that occur in at least one other bag. In our example, $\partial T = \emptyset$, $\partial T_1 = \{x_3\}$ and $\partial T_2 = \{x_4\}$.

Let $\mathcal{T}$ be a TBox of finite depth k. A *type* is a partial map $\boldsymbol{w}$ from V to $\mathbf{W}_{\mathcal{T}}$; its domain is denoted by $\mathsf{dom}(\boldsymbol{w})$. By $\boldsymbol{\varepsilon}$ we denote the unique partial type with $\mathsf{dom}(\boldsymbol{\varepsilon}) = \emptyset$. We use types to represent how variables are mapped into $\mathcal{C}_{\mathcal{T},\mathcal{A}}$, with $\boldsymbol{w}(u) = w$ indicating that u is mapped to an element of the form aw (for some $a \in \mathsf{ind}(\mathcal{A})$), and with $\boldsymbol{w}(u) = \varepsilon$ that u is mapped to an ABox individual. We say that a type $\boldsymbol{w}$ is *compatible* with a bag t if, for all $u, v \in \lambda(t) \cap \mathsf{dom}(\boldsymbol{w})$, we have

- if $v \in \mathsf{avar}(\boldsymbol{q})$, then $\boldsymbol{w}(v) = \varepsilon$;
- if $A(v) \in \boldsymbol{q}$, then either $\boldsymbol{w}(v) = \varepsilon$ or $\boldsymbol{w}(v) = wR$ with $\exists R^- \sqsubseteq_{\mathcal{T}} A$;
- if $R(v, u) \in \boldsymbol{q}$, then $\boldsymbol{w}(v) = \boldsymbol{w}(u) = \varepsilon$, or $\boldsymbol{w}(u) = \boldsymbol{w}(v)R'$ with $R' \sqsubseteq_{\mathcal{T}} R$, or $\boldsymbol{w}(v) = \boldsymbol{w}(u)R'$ with $R' \sqsubseteq_{\mathcal{T}} R^-$.

In the sequel, we abuse notation and use sets of variables in place of sequences assuming that they are ordered in some (fixed) way. For example, we use $\boldsymbol{x}_D$ for a tuple of variables in the set $\boldsymbol{x}_D$ (ordered in some way). Also, given a tuple $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{A})$ of length $|\boldsymbol{x}_D|$ and $x \in \boldsymbol{x}_D$, we write $\boldsymbol{a}(x)$ to refer to the element of $\boldsymbol{a}$ that corresponds to x (that is, to the component of the tuple with the same index).

Let $\Pi_{\boldsymbol{Q}}$ be an NDL program that—for any $D \in \mathsf{sub}(T)$, any types $\boldsymbol{w}$ and $\boldsymbol{s}$ for which $\mathsf{dom}(\boldsymbol{w}) = \partial D$, $\mathsf{dom}(\boldsymbol{s}) = \lambda(\sigma(D))$, $\boldsymbol{s}$ is compatible with $\sigma(D)$ and agrees with

$\boldsymbol{w}$ on their common domain—contains the clause

$$G_D^{\boldsymbol{w}}(\partial D, \boldsymbol{x}_D) \;\leftarrow\; \mathsf{At}^{\boldsymbol{s}} \wedge \bigwedge\nolimits_{D' \prec D} G_{D'}^{(\boldsymbol{s}\cup\boldsymbol{w})\restriction\partial D'}(\partial D', \boldsymbol{x}_{D'}), \tag{2}$$

where $\boldsymbol{x}_D$ are the parameters of predicate $G_D^{\boldsymbol{w}}$, $(\boldsymbol{s}\cup\boldsymbol{w})\restriction\partial D'$ is the restriction[1] of the union $\boldsymbol{s}\cup\boldsymbol{w}$ of $\boldsymbol{s}$ and $\boldsymbol{w}$ to $\partial D'$, and $\mathsf{At}^{\boldsymbol{s}}$ is defined as follows:

$$\mathsf{At}^{\boldsymbol{s}} \;=\; \bigwedge_{\substack{A(u)\in\boldsymbol{q}\\ \boldsymbol{s}(u)=\varepsilon}} A(u) \;\wedge\; \bigwedge_{\substack{R(u,v)\in\boldsymbol{q}\\ \boldsymbol{s}(u)=\boldsymbol{s}(v)=\varepsilon}} R(u,v) \;\wedge\; \bigwedge_{\substack{R(u,v)\in\boldsymbol{q}\\ \boldsymbol{s}(u)\neq\varepsilon \text{ or } \boldsymbol{s}(v)\neq\varepsilon}} (u=v) \;\wedge\; \bigwedge_{\substack{\boldsymbol{s}(u)=Sw'\\ \text{for some } w'}} A_S(u). \tag{3}$$

The first two conjunctions in $\mathsf{At}^{\boldsymbol{s}}$ ensure that atoms all of whose variables are assigned ε are present in the ABox. The third conjunction ensures that if one of the variables in a role atom is not mapped to ε, then the images of the variables share the same initial individual. Finally, atoms in the final conjunction ensure that if a variable is to be mapped to aSw', then the individual a satisfies $\exists S$ (so aSw' is part of the domain of $\mathcal{C}_{\mathcal{T},\mathcal{A}}$).

Example 9. Now we fix an ontology $\mathcal{T}$ with the following axioms:

$$A \equiv \exists P, \qquad P \sqsubseteq S, \qquad P \sqsubseteq R^-, \qquad\qquad B \equiv \exists Q, \qquad Q \sqsubseteq R, \qquad Q \sqsubseteq S^-.$$

Since $\lambda(t) = \{x_3, x_4\}$, there are only three types compatible with t:

$$\boldsymbol{s}_1 \colon x_3 \mapsto \varepsilon, x_4 \mapsto \varepsilon, \qquad \boldsymbol{s}_2 \colon x_3 \mapsto P, x_4 \mapsto \varepsilon \quad \text{and} \quad \boldsymbol{s}_3 \colon x_3 \mapsto \varepsilon, x_4 \mapsto Q.$$

So, $\mathsf{At}^{\boldsymbol{s}_1} = R(x_3,x_4)$, $\mathsf{At}^{\boldsymbol{s}_2} = A(x_3) \wedge (x_3 = x_4)$, $\mathsf{At}^{\boldsymbol{s}_3} = B(x_4) \wedge (x_3 = x_4)$. Thus, predicate $G_T^{\boldsymbol{\varepsilon}}$ is defined by the following clauses, for $\boldsymbol{s}_1$, $\boldsymbol{s}_2$ and $\boldsymbol{s}_3$, respectively:

$$\begin{aligned}
&G_T^{\boldsymbol{\varepsilon}}(x_0,x_7) \leftarrow G_{T_1}^{x_3\mapsto\varepsilon}(x_3,x_0) \wedge R(x_3,x_4) \wedge G_{T_2}^{x_4\mapsto\varepsilon}(x_4,x_7),\\
&G_T^{\boldsymbol{\varepsilon}}(x_0,x_7) \leftarrow G_{T_1}^{x_3\mapsto P}(x_3,x_0) \wedge A(x_3) \wedge (x_3 = x_4) \wedge G_{T_2}^{x_4\mapsto\varepsilon}(x_4,x_7),\\
&G_T^{\boldsymbol{\varepsilon}}(x_0,x_7) \leftarrow G_{T_1}^{x_3\mapsto\varepsilon}(x_3,x_0) \wedge B(x_4) \wedge (x_3 = x_4) \wedge G_{T_2}^{x_4\mapsto Q}(x_4,x_7).
\end{aligned}$$

By induction on $\prec$ on $\mathsf{sub}(T)$, we show that $(\Pi_{\boldsymbol{Q}}, G_T^{\boldsymbol{\varepsilon}})$ is a rewriting of $\boldsymbol{Q}(\boldsymbol{x})$.

Lemma 10. *For any ABox $\mathcal{A}$, any $D \in \mathsf{sub}(T)$, any type $\boldsymbol{w}$ with $\mathsf{dom}(\boldsymbol{w}) = \partial D$, any $\boldsymbol{b} \in \mathsf{ind}(\mathcal{A})^{|\partial D|}$ and $\boldsymbol{a} \in \mathsf{ind}(\mathcal{A})^{|\boldsymbol{x}_D|}$, we have $\Pi_{\boldsymbol{Q}}, \mathcal{A} \models G_D^{\boldsymbol{w}}(\boldsymbol{b},\boldsymbol{a})$ iff there is a homomorphism $h \colon \boldsymbol{q}_D \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that*

$$h(x) = \boldsymbol{a}(x), \text{ for } x \in \boldsymbol{x}_D, \qquad \text{and} \qquad h(v) = \boldsymbol{b}(v)\boldsymbol{w}(v), \text{ for } v \in \partial D.$$

Fix now k and t, and consider the class of OMQs $\boldsymbol{Q}(\boldsymbol{x}) = (\mathcal{T}, \boldsymbol{q}(\boldsymbol{x}))$ with $\mathcal{T}$ of depth $\leq k$ and $\boldsymbol{q}$ of treewidth $\leq t$. Let T be a tree decomposition of $\boldsymbol{q}$ of treewidth $\leq t$. We take the following weight function: $\nu(G_D^{\boldsymbol{w}}) = |D|$. Clearly, $\nu(G_T^{\boldsymbol{\varepsilon}}) \leq |\boldsymbol{Q}|$. By Lemma 8, $\mathsf{d}(\Pi_{\boldsymbol{Q}}, G_T^{\boldsymbol{\varepsilon}}) \leq 2\log|T| = 2\log\nu(G_T^{\boldsymbol{\varepsilon}}) \leq 2\log|\boldsymbol{Q}|$. Since $|\mathsf{sub}(T)| \leq |T|^2$ and there are at most $|\mathcal{T}|^{2tk}$ options for $\boldsymbol{w}$, there are polynomially many predicates $G_D^{\boldsymbol{w}}$, and so $\Pi_{\boldsymbol{Q}}$ is of polynomial size. Thus, by Corollary 6, the obtained NDL-rewriting over arbitrary ABoxes can be evaluated in LOGCFL. Finally, we note that a tree decomposition of treewidth $\leq t$ can be computed using an $\mathsf{L}^{\mathsf{LOGCFL}}$-transducer [10], and so the NDL-rewriting can also be constructed by an $\mathsf{L}^{\mathsf{LOGCFL}}$-transducer.

[1] By construction, $\mathsf{dom}(\boldsymbol{s}\cup\boldsymbol{w})$ covers $\partial D'$, and so the domain of $(\boldsymbol{s}\cup\boldsymbol{w})\restriction\partial D'$ is $\partial D'$.

5 Bounded-Leaf CQs and Bounded-Depth TBoxes

We next consider OMQs with tree-shaped CQs in which both the depth of the ontology and the number of leaves in the CQ are bounded. Let $\mathcal{T}$ be a TBox of finite depth k, and let $\boldsymbol{q}(\boldsymbol{x})$ be a tree-shaped CQ with at most ℓ leaves. Fix one of the variables of $\boldsymbol{q}$ as root, and let M be the maximal distance to a leaf from the root. For $n \leq M$, let $\boldsymbol{z}^n$ denote the set of all variables of $\boldsymbol{q}$ at distance n from the root; clearly, $|\boldsymbol{z}^n| \leq \ell$. We call the $\boldsymbol{z}^n$ *slices* of $\boldsymbol{q}$ and observe that they satisfy the following: for every $R(u, v) \in \boldsymbol{q}$ with $u \neq v$, there exists $0 \leq n < M$ such that either $u \in \boldsymbol{z}^n$ and $v \in \boldsymbol{z}^{n+1}$ or $u \in \boldsymbol{z}^{n+1}$ and $v \in \boldsymbol{z}^n$. For $0 \leq n \leq M$, we denote by $\boldsymbol{q}_n(\boldsymbol{z}^n_\exists, \boldsymbol{x}^n)$ the query consisting of all atoms $S(\boldsymbol{u})$ of $\boldsymbol{q}$ such that $\boldsymbol{u} \subseteq \bigcup_{n \leq m \leq M} \boldsymbol{z}^m$, where $\boldsymbol{x}^n = \mathsf{var}(\boldsymbol{q}_n) \cap \boldsymbol{x}$ and $\boldsymbol{z}^n_\exists = \boldsymbol{z}^n \setminus \boldsymbol{x}$.

By *type of a slice* $\boldsymbol{z}^n$, we mean a total map $\boldsymbol{w}$ from $\boldsymbol{z}^n$ to $\mathbf{W}_\mathcal{T}$. Analogously to Section 4, we define what it means for a type (or pair of types) to be compatible with a slice (pair of adjacent slices). We call $\boldsymbol{w}$ *locally compatible* with $\boldsymbol{z}^n$ if for every $z \in \boldsymbol{z}^n$:

- if $z \in \mathsf{avar}(\boldsymbol{q})$, then $\boldsymbol{w}(z) = \varepsilon$;
- if $A(z) \in \boldsymbol{q}$, then either $\boldsymbol{w}(z) = \varepsilon$ or $\boldsymbol{w}(z) = wR$ with $\exists R^- \sqsubseteq_\mathcal{T} A$;
- if $R(z, z) \in \boldsymbol{q}$, then $\boldsymbol{w}(z) = \varepsilon$.

If $\boldsymbol{w}, \boldsymbol{s}$ are types for $\boldsymbol{z}^n$ and $\boldsymbol{z}^{n+1}$ respectively, then we call $(\boldsymbol{w}, \boldsymbol{s})$ *compatible* with $(\boldsymbol{z}^n, \boldsymbol{z}^{n+1})$ if $\boldsymbol{w}$ is locally compatible with $\boldsymbol{z}^n$, $\boldsymbol{s}$ is locally compatible with $\boldsymbol{z}^{n+1}$, and for every atom $R(z^n, z^{n+1}) \in \boldsymbol{q}$, one of the following holds: (*i*) $\boldsymbol{w}(z^n) = \boldsymbol{s}(z^{n+1}) = \varepsilon$, (*ii*) $\boldsymbol{s}(z^{n+1}) = \boldsymbol{w}(z^n)R'$ with $R' \sqsubseteq_\mathcal{T} R$, or (*iii*) $\boldsymbol{w}(z^n) = \boldsymbol{s}(z^{n+1})R'$ with $R' \sqsubseteq_\mathcal{T} R^-$.

Consider the NDL program $\Pi'_{\boldsymbol{q}}$ defined as follows. For every $0 \leq n < M$ and every pair of types $(\boldsymbol{w}, \boldsymbol{s})$ that is compatible with $(\boldsymbol{z}^n, \boldsymbol{z}^{n+1})$, we include the clause:

$$P^{\boldsymbol{w}}_n(\boldsymbol{z}^n_\exists, \boldsymbol{x}^n) \leftarrow \mathsf{At}^{\boldsymbol{w} \cup \boldsymbol{s}}(\boldsymbol{z}^n, \boldsymbol{z}^{n+1}) \wedge P^{\boldsymbol{s}}_{n+1}(\boldsymbol{z}^{n+1}_\exists, \boldsymbol{x}^{n+1}),$$

where $\boldsymbol{x}^n$ are the parameters of $P^{\boldsymbol{w}}_n$ and $\mathsf{At}^{\boldsymbol{w} \cup \boldsymbol{s}}(\boldsymbol{z}^n, \boldsymbol{z}^{n+1})$ is the conjunction of atoms (3), as defined in Section 4, for the union $\boldsymbol{w} \cup \boldsymbol{s}$ of types $\boldsymbol{w}$ and $\boldsymbol{s}$.

For every type $\boldsymbol{w}$ locally compatible with $\boldsymbol{z}^M$, we include the clause:

$$P^{\boldsymbol{w}}_M(\boldsymbol{z}^M_\exists, \boldsymbol{x}^M) \leftarrow \mathsf{At}^{\boldsymbol{w}}(\boldsymbol{z}^M).$$

(Recall that $\boldsymbol{z}^M$ is a disjoint union of $\boldsymbol{z}^M_\exists$ and $\boldsymbol{x}^M$.) We use G, with parameters $\boldsymbol{x}$, as the goal predicate and include $G(\boldsymbol{x}) \leftarrow P^{\boldsymbol{w}}_0(\boldsymbol{z}^0, \boldsymbol{x})$ for every predicate $P^{\boldsymbol{w}}_0(\boldsymbol{z}^0, \boldsymbol{x}^0)$ occurring in the head of one of the preceding clauses.

The following lemma (which is proved by induction) is the key step in showing that $(\Pi'_{\boldsymbol{q}}, G(\boldsymbol{x}))$ is a rewriting of $(\mathcal{T}, \boldsymbol{q})$ over H-complete ABoxes:

Lemma 11. *For any H-complete ABox $\mathcal{A}$, any $0 \leq n \leq M$, any predicate $P^{\boldsymbol{w}}_n$, any $\boldsymbol{b} \in \mathsf{ind}(\mathcal{A})^{|\boldsymbol{z}^n_\exists|}$ and any $\boldsymbol{a} \in \mathsf{ind}(\mathcal{A})^{|\boldsymbol{x}^n|}$, we have $\Pi'_{\boldsymbol{q}}, \mathcal{A} \models P^{\boldsymbol{w}}_n(\boldsymbol{b}, \boldsymbol{a})$ iff there is a homomorphism $h\colon \boldsymbol{q}_n \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that*

$$h(x) = \boldsymbol{a}(x), \text{ for } x \in \boldsymbol{x}^n, \quad \text{and} \quad h(z) = \boldsymbol{b}(z)\boldsymbol{w}(z), \text{ for } z \in \boldsymbol{z}^n_\exists. \tag{4}$$

It should be clear that $\Pi'_{\boldsymbol{q}}$ is a linear NDL program of width at most 2ℓ. Moreover, when ℓ and k are bounded by fixed constants, it takes only logarithmic space to store a type $\boldsymbol{w}$, which allows us to show that $\Pi'_{\boldsymbol{q}}$ can be computed by an L^{NL}-transducer. We can apply Lemma 2 to obtain an NDL rewriting for arbitrary ABoxes, and then use Theorem 1 to conclude that the resulting program can be evaluated in NL.

6 Bounded-Leaf CQs and Arbitrary TBoxes

For OMQs with bounded-leaf CQs and ontologies of unbounded depth, our rewriting utilises the notion of tree witness [15]. Let $\boldsymbol{Q}(\boldsymbol{x}) = (\mathcal{T}, \boldsymbol{q}(\boldsymbol{x}))$ with $\boldsymbol{q}(\boldsymbol{x}) = \exists \boldsymbol{y}\, \varphi(\boldsymbol{x}, \boldsymbol{y})$. For a pair $\mathtt{t} = (\mathtt{t}_\mathsf{r}, \mathtt{t}_\mathsf{i})$ of disjoint sets of variables in $\boldsymbol{q}$, with $\mathtt{t}_\mathsf{i} \subseteq \boldsymbol{y}$ and $\mathtt{t}_\mathsf{i} \neq \emptyset$, set

$$\boldsymbol{q}_\mathtt{t} \;=\; \big\{\, S(\boldsymbol{z}) \in \boldsymbol{q} \mid \boldsymbol{z} \subseteq \mathtt{t}_\mathsf{r} \cup \mathtt{t}_\mathsf{i} \text{ and } \boldsymbol{z} \not\subseteq \mathtt{t}_\mathsf{r} \,\big\}.$$

If $\boldsymbol{q}_\mathtt{t}$ is a minimal subset of $\boldsymbol{q}$ for which there is a homomorphism $h\colon \boldsymbol{q}_\mathtt{t} \to \mathcal{C}_\mathcal{T}^{A_R(a)}$ such that $\mathtt{t}_\mathsf{r} = h^{-1}(a)$ and $\boldsymbol{q}_\mathtt{t}$ contains every atom of $\boldsymbol{q}$ with at least one variable from $\mathtt{t}_\mathsf{i}$, then we call $\mathtt{t} = (\mathtt{t}_\mathsf{r}, \mathtt{t}_\mathsf{i})$ a *tree witness for $\boldsymbol{Q}$ generated by R*. Note that the same tree witness $\mathtt{t}$ can be generated by different roles R.

The logarithmic-depth NDL-rewriting for bounded-leaf queries and ontologies of unbounded depth is based upon the following observation [12].

Lemma 12. *Every tree T of size m has a node splitting it into subtrees of size $\leq \lceil m/2 \rceil$.*

We will use repeated applications of this lemma to decompose the input CQ into smaller and smaller subqueries. Formally, for every tree-shaped CQ $\boldsymbol{q}$, we use $v_{\boldsymbol{q}}$ to denote a vertex in the Gaifman graph $\mathcal{G}$ of $\boldsymbol{q}$ that satisfies the condition of Lemma 12. If $|\mathsf{var}(\boldsymbol{q})| = 2$ and $\boldsymbol{q}$ has at least one existential variable, we assume that $v_{\boldsymbol{q}}$ is existentially quantified. Then, for an OMQ $\boldsymbol{Q} = (\mathcal{T}, \boldsymbol{q}_0(\boldsymbol{x}))$, we define SQ as the smallest set of queries that contains $\boldsymbol{q}_0(\boldsymbol{x})$ and is such that, for every $\boldsymbol{q}(\boldsymbol{z}) \in \mathsf{SQ}$ with $\mathsf{var}(\boldsymbol{q}) \neq \boldsymbol{z}$, the following queries also belong to SQ:

- for every u_i adjacent to $v_{\boldsymbol{q}}$ in $\mathcal{G}$, the query $\boldsymbol{q}_i(\boldsymbol{z}_i)$ comprising all role atoms linking $v_{\boldsymbol{q}}$ and u_i, as well as all atoms whose variables cannot reach $v_{\boldsymbol{q}}$ in $\mathcal{G}$ without passing by u_i, and with $\boldsymbol{z}_i = \mathsf{var}(\boldsymbol{q}_i) \cap (\boldsymbol{z} \cup \{v_{\boldsymbol{q}}\})$;
- for every tree witness $\mathtt{t}$ for $(\mathcal{T}, \boldsymbol{q}(\boldsymbol{z}))$ with $\mathtt{t}_\mathsf{r} \neq \emptyset$ and $v_{\boldsymbol{q}} \in \mathtt{t}_\mathsf{i}$, the queries $\boldsymbol{q}_1^\mathtt{t}(\boldsymbol{z}_1^\mathtt{t}), \dots, \boldsymbol{q}_m^\mathtt{t}(\boldsymbol{z}_m^\mathtt{t})$ that correspond to the connected components of the set of atoms of $\boldsymbol{q}$ that are not in $\boldsymbol{q}_\mathtt{t}$, with $\boldsymbol{z}_i^\mathtt{t} = \mathsf{var}(\boldsymbol{q}_i^\mathtt{t}) \cap (\boldsymbol{z} \cup \mathtt{t}_\mathsf{r})$.

The NDL program $\Pi''_{\boldsymbol{Q}}$ uses IDB predicates $P_{\boldsymbol{q}}$, for $\boldsymbol{q}(\boldsymbol{z}) \in \mathsf{SQ}$, with arity $|\boldsymbol{z}|$ and parameters $\mathsf{var}(\boldsymbol{q}) \cap \boldsymbol{x}$. For each $\boldsymbol{q}(\boldsymbol{z}) \in \mathsf{SQ}$ with $\mathsf{var}(\boldsymbol{q}) = \boldsymbol{z}$, we include the clause $P_{\boldsymbol{q}}(\boldsymbol{z}) \leftarrow \boldsymbol{q}(\boldsymbol{z})$. For each $\boldsymbol{q}(\boldsymbol{z}) \in \mathsf{SQ}$ with $\mathsf{var}(\boldsymbol{q}) \neq \boldsymbol{z}$, we include the clause

$$P_{\boldsymbol{q}}(\boldsymbol{z}) \;\leftarrow\; \bigwedge_{A(v_{\boldsymbol{q}}) \in \boldsymbol{q}} A(v_{\boldsymbol{q}}) \;\wedge\; \bigwedge_{R(v_{\boldsymbol{q}}, v_{\boldsymbol{q}}) \in \boldsymbol{q}} R(v_{\boldsymbol{q}}, v_{\boldsymbol{q}}) \;\wedge\; \bigwedge_{1 \leq i \leq n} P_{\boldsymbol{q}_i}(\boldsymbol{z}_i),$$

where $\boldsymbol{q}_1(\boldsymbol{z}_1), \dots, \boldsymbol{q}_n(\boldsymbol{z}_n)$ are the subqueries induced by the neighbours of $v_{\boldsymbol{q}}$ in $\mathcal{G}$, and the following clause

$$P_{\boldsymbol{q}}(\boldsymbol{z}) \;\leftarrow\; \bigwedge_{u, u' \in \mathtt{t}_\mathsf{r}} (u = u') \;\wedge\; \bigwedge_{u \in \mathtt{t}_\mathsf{r}} A_R(u) \;\wedge\; \bigwedge_{1 \leq i \leq m} P_{\boldsymbol{q}_i^\mathtt{t}}(\boldsymbol{z}_i^\mathtt{t})$$

for every tree witness $\mathtt{t}$ for $(\mathcal{T}, \boldsymbol{q}(\boldsymbol{z}))$ with $\mathtt{t}_\mathsf{r} \neq \emptyset$ and $v_{\boldsymbol{q}} \in \mathtt{t}_\mathsf{i}$ and for every role R generating $\mathtt{t}$, where $\boldsymbol{q}_1^\mathtt{t}, \dots, \boldsymbol{q}_m^\mathtt{t}$ are the connected components of $\boldsymbol{q}$ without $\boldsymbol{q}_\mathtt{t}$. Finally, if $\boldsymbol{q}_0$ is Boolean, then we additionally include clauses $P_{\boldsymbol{q}_0} \leftarrow A(x)$ for all concept names A such that $\mathcal{T}, \{A(a)\} \models \boldsymbol{q}_0$.

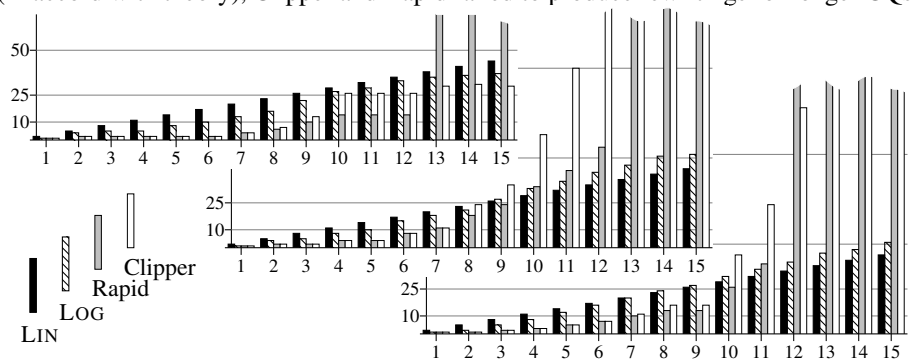
The program $\Pi''_{\boldsymbol{Q}}$ is inspired by a similar construction from [12]. By adapting results from the latter paper, we can show that $(\Pi''_{\boldsymbol{Q}}, P_{\boldsymbol{q}_0}(\boldsymbol{x}))$ is indeed a rewriting:

Lemma 13. *For any tree-shaped OMQ $\boldsymbol{Q}(\boldsymbol{x}) = (\mathcal{T}, \boldsymbol{q}_0(\boldsymbol{x}))$, any $\boldsymbol{q}(\boldsymbol{z}) \in \mathsf{SQ}$, any H-complete ABox $\mathcal{A}$, and any tuple $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{A})$, $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}}(\boldsymbol{a})$ iff there exists a homomorphism $h\colon \boldsymbol{q} \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that $h(\boldsymbol{z}) = \boldsymbol{a}$.*

Now fix $\ell > 1$, and consider the class of OMQs $\boldsymbol{Q}(\boldsymbol{x}) = (\mathcal{T}, \boldsymbol{q}(\boldsymbol{x}))$ with tree-shaped $\boldsymbol{q}(\boldsymbol{x})$ having at most ℓ leaves. The size of $\Pi''_{\boldsymbol{Q}}$ is polynomially bounded in $|\boldsymbol{Q}|$, since bounded-leaf CQs have polynomially many tree witnesses and also polynomially many tree-shaped subCQs. It is readily seen that the function ν defined by setting $\nu(P_{\boldsymbol{q}'}) = |\boldsymbol{q}'|$ is a weight function for $(\Pi''_{\boldsymbol{Q}}, P_{\boldsymbol{q}})$ such that $\nu(P_{\boldsymbol{q}}) \leq |\boldsymbol{Q}|$. Moreover, by Lemma 12, $\mathsf{d}(\Pi, G) \leq \log \nu(P_{\boldsymbol{q}}) + 1$. We can thus apply Corollary 6 to conclude that the obtained NDL-rewritings can be evaluated in LOGCFL. Finally, we note that since the number of leaves is bounded, it is in NL to decide whether a vertex satisfies the conditions of Lemma 12, and it is in LOGCFL to decide whether $\mathcal{T}, \{A(a)\} \models \boldsymbol{q}_0$ [3] or whether a (logspace) representation of a possible tree witness is indeed a tree witness. This allows us to show that $(\Pi''_{\boldsymbol{Q}}, P_{\boldsymbol{q}})$ can be generated by an $\mathsf{L}^{\mathsf{LOGCFL}}$-transducer.

7 Conclusions

As shown above, for three important classes of OMQs, NDL-rewritings can be constructed and evaluated by theoretically optimal NL and LOGCFL algorithms. To see whether these rewritings are viable in practice, we generated three sequences of OMQs with the ontology from Example 9 and linear CQs of up to 15 atoms as in Example 7. We compared our NL and LOGCFL rewritings from Secs. 5 and 4 (called LIN and LOG) with those produced by Clipper [8] and Rapid [6]. The barcharts below show the number of clauses in the rewritings over H-complete ABoxes. While LIN and LOG grow linearly (in accord with theory), Clipper and Rapid failed to produce rewritings for longer CQs.



We evaluated the rewritings over a few randomly generated ABoxes using off-the-shelf datalog engine RDFox [17]. The experiments (see the full version) show that our rewritings are usually executed faster than Clipper's and Rapid's when the number of answers is relatively small ($\lesssim 10^4$); for queries with $\gtrsim 10^6$ answers, the execution times are comparable. The version of RDFox we used did not seem to take advantage of the structure of the NL/LOGCFL rewritings, and it would be interesting to see whether their nonrecursiveness and parallelisability can be utilised to produce efficient execution plans.

Acknowledgements: The first author was supported by contract ANR-12-JS02-007-01, the fourth by the Russian Foundation for Basic Research and the grant MK-7312.2016.1.

A Proofs

Lemma 2. *For any fixed* $\mathsf{w} > 0$*, there is an* L^{NL}*-transducer that, given a linear NDL-rewriting of an OMQ* $\boldsymbol{Q}(\boldsymbol{x})$ *over H-complete ABoxes that is of width at most* w*, computes a linear NDL-rewriting of* $\boldsymbol{Q}(\boldsymbol{x})$ *over arbitrary ABoxes whose width is at most* $\mathsf{w}+1$.

Proof. Let $(\Pi, G(\boldsymbol{x}))$ be a linear NDL-rewriting of the OMQ $\boldsymbol{Q}(\boldsymbol{x}) = (\mathcal{T}, \boldsymbol{q}(\boldsymbol{x}))$ over H-complete ABoxes of width w. and we will replace every clause λ in Π by a set of clauses λ^* defined as follows. Suppose λ is of the form

$$H(\boldsymbol{z}) \leftarrow I \wedge \boldsymbol{EQ} \wedge E_1 \wedge \ldots \wedge E_n,$$

where I is the only IDB body atom in λ, $\boldsymbol{EQ}$ contains all equality body atoms, and $E_1, \ldots, E_n$ are the EDB body atoms not involving equality. For every atom E_i, we define a set $\upsilon(E_i)$ of atoms by taking

$$\begin{aligned} \upsilon(E_i) &= \big\{B(u) \mid B \sqsubseteq_{\mathcal{T}} A\big\} \cup \big\{R(u,u_i) \mid \exists R \sqsubseteq_{\mathcal{T}} A\big\}, && \text{if } E_i = A(u), \\ \upsilon(E_i) &= \big\{R'(u,v) \mid R' \sqsubseteq_{\mathcal{T}} R\big\}, && \text{if } E_i = R(u,v), \end{aligned}$$

where u_i is a fresh variable not occurring in λ; we assume $P^-(u,v)$ coincides with $P(v,u)$, for all role names P. Intuitively, $\upsilon(E_i)$ captures all atoms that imply E_i with respect to $\mathcal{T}$. Then λ^* consists of the following clauses:

$$\begin{aligned} H_0(\boldsymbol{z}_0) &\leftarrow I, \\ H_{i+1}(\boldsymbol{z}_i) &\leftarrow H_i(\boldsymbol{z}_i) \wedge E_i', && \text{for every } 1 \leq i \leq n \text{ and every } E_i' \in \upsilon(E_i), \\ H(\boldsymbol{z}) &\leftarrow H_{n+1}(\boldsymbol{z}_n) \wedge \boldsymbol{EQ}, \end{aligned}$$

where $\boldsymbol{z}_i$ is the restriction of $\boldsymbol{z}$ to variables occurring in I if $i = 0$ and in $H_i(\boldsymbol{z}_i)$ and E_i' except for u_i if $i > 0$ (note that $\boldsymbol{z}_n = \boldsymbol{z}$). Let Π' be the program obtained from Π by replacing each clause λ by the set of clauses λ^*. By construction, Π' is a linear NDL program and its width cannot exceed $\mathsf{w}+1$ (the possible increase of 1 is due to the replacement of concept atoms by role atoms).

We now argue that $(\Pi', G(\boldsymbol{x}))$ is a rewriting of $\boldsymbol{Q}(\boldsymbol{x})$ over arbitrary ABoxes. It is easily verified that $(\Pi', G(\boldsymbol{x}))$ is equivalent to $(\Pi'', G(\boldsymbol{x}))$, where Π'' is obtained from Π by replacing each clause $H(\boldsymbol{z}) \leftarrow I \wedge \boldsymbol{EQ} \wedge E_1 \wedge \ldots \wedge E_n$ by the (possibly exponentially larger) set of clauses

$$\big\{H(\boldsymbol{z}) \leftarrow I \wedge \boldsymbol{EQ} \wedge E_1' \wedge \ldots \wedge E_n' \mid E_i' \in \upsilon(E_i), 1 \leq i \leq n\big\}.$$

It thus suffices to show that $(\Pi'', G(\boldsymbol{x}))$ is a rewriting of $\boldsymbol{Q}(\boldsymbol{x})$ over arbitrary ABoxes.

First suppose that $\mathcal{T}, \mathcal{A} \models \boldsymbol{q}(\boldsymbol{a})$, where $\mathcal{A}$ is an arbitrary ABox. Let $\mathcal{A}'$ be the H-complete ABox obtained from $\mathcal{A}$ by adding the assertions:

- $P(a,b)$ whenever $R(a,b) \in \mathcal{A}$ and $R \sqsubseteq_{\mathcal{T}} P$;
- $A(a)$ whenever $B(a) \in \mathcal{A}$ (with B a basic concept) and $B \sqsubseteq_{\mathcal{T}} A$.

Clearly, $\mathcal{T}, \mathcal{A}' \models \boldsymbol{q}(\boldsymbol{a})$, so we must have $\Pi, \mathcal{A}' \models G(\boldsymbol{a})$. A simple inductive argument (on the order of derivation of ground atoms) shows that whenever a clause

$H(\boldsymbol{z}) \leftarrow I \wedge \boldsymbol{EQ} \wedge E_1 \wedge \ldots \wedge E_n$ is applied using a substitution $\boldsymbol{c}$ for the variables in the body to derive $H(\boldsymbol{c}(\boldsymbol{z}))$ using Π, we can find a corresponding clause $H(\boldsymbol{z}) \leftarrow I \wedge \boldsymbol{EQ} \wedge E'_1 \wedge \ldots \wedge E'_n$ and a substitution $\boldsymbol{c}'$ extending $\boldsymbol{c}$ (on the fresh variables u_i) that allows us to derive $H(\boldsymbol{c}'(\boldsymbol{z}))$ using Π''. Indeed, if $E_i = A(u)$, then $A(\boldsymbol{c}(u)) \in \mathcal{A}'$, so there must exist either a concept assertion $A'(\boldsymbol{c}(u)) \in \mathcal{A}$ such that $A' \sqsubseteq_{\mathcal{T}} A$ or a role assertion $R(a, b) \in \mathcal{A}$ such that $\exists R \sqsubseteq_{\mathcal{T}} A$. Similarly, if $E_i = R(u, v)$, then there must exist a role assertion $R'(u, v) \in \mathcal{A}$ such that $R' \sqsubseteq_{\mathcal{T}} R$. It then suffices to choose a clause $H(\boldsymbol{z}) \leftarrow I \wedge \boldsymbol{EQ} \wedge E'_1 \wedge \ldots \wedge E'_n$ with atoms E'_i whose form matches that of the assertion in $\mathcal{A}$ corresponding to E_i.

For the converse direction, it suffices to observe that $\Pi \subseteq \Pi''$.

To complete the proof, we note that it is in NL to decide whether an atom belongs to $v(E_i)$, and thus we can construct the program Π' by means of an L^{NL}-transducer. ❑

Lemma 10. *For any ABox $\mathcal{A}$, any $D \in \mathsf{sub}(T)$, any type $\boldsymbol{w}$ with $\mathsf{dom}(\boldsymbol{w}) = \partial D$, any $\boldsymbol{b} \in \mathsf{ind}(\mathcal{A})^{|\partial D|}$ and $\boldsymbol{a} \in \mathsf{ind}(\mathcal{A})^{|\boldsymbol{x}_D|}$, we have $\Pi_{\boldsymbol{Q}}, \mathcal{A} \models G_D^{\boldsymbol{w}}(\boldsymbol{b}, \boldsymbol{a})$ iff there is a homomorphism $h\colon \boldsymbol{q}_D \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that*

$$h(x) = \boldsymbol{a}(x), \text{ for } x \in \boldsymbol{x}_D, \quad \text{and} \quad h(v) = \boldsymbol{b}(v)\boldsymbol{w}(v), \text{ for } v \in \partial D. \tag{5}$$

Proof. ($\Rightarrow$) The proof is by induction on $\prec$. For the base of induction, let D be of size 1. By the definition of $\Pi_{\boldsymbol{Q}}$, there exists a type $\boldsymbol{s}$ such that $\mathsf{dom}(\boldsymbol{s}) = \lambda(\sigma(D))$ and $\boldsymbol{w}$ agrees with $\boldsymbol{s}$ on ∂D and a respective tuple $\boldsymbol{c} \in \mathsf{ind}(\mathcal{A})^{|\lambda(\sigma(D))|}$ such that $\boldsymbol{c}(v) = \boldsymbol{b}(v)$, for all $v \in \partial D$, and $\boldsymbol{c}(x) = \boldsymbol{a}(x)$, for all $x \in \boldsymbol{x}_D$, and $\Pi_{\boldsymbol{Q}}, \mathcal{A} \models \mathsf{At}^{\boldsymbol{s}}(\boldsymbol{c})$. Then, for any atom $S(\boldsymbol{v}) \in \boldsymbol{q}_D$, we have $\boldsymbol{v} \subseteq \lambda(\sigma(D))$, whence $\mathcal{C}_{\mathcal{T},\mathcal{A}} \models S(h(\boldsymbol{v}))$ as $\boldsymbol{w}$ agrees with $\boldsymbol{s}$ on ∂D.

For the inductive step, suppose that $\Pi_{\boldsymbol{Q}}, \mathcal{A} \models G_D^{\boldsymbol{w}}(\boldsymbol{b}, \boldsymbol{a})$. By the definition of $\Pi_{\boldsymbol{Q}}$, there exists a type $\boldsymbol{s}$ such that $\mathsf{dom}(\boldsymbol{s}) = \lambda(\sigma(D))$ and $\boldsymbol{w}$ agrees with $\boldsymbol{s}$ on their common domain and a respective tuple $\boldsymbol{c} \in \mathsf{ind}(\mathcal{A})^{|\lambda(\sigma(D))|}$ such that $\boldsymbol{c}(v) = \boldsymbol{b}(v)$, for all $v \in \partial D$, and $\boldsymbol{c}(x) = \boldsymbol{a}(x)$, for all $x \in \boldsymbol{x}_D$, and

$$\Pi_{\boldsymbol{Q}}, \mathcal{A} \models \mathsf{At}^{\boldsymbol{s}}(\boldsymbol{c}) \wedge \bigwedge\nolimits_{D' \prec D} G_{D'}^{(\boldsymbol{s} \cup \boldsymbol{w}) \restriction \partial D'}(\boldsymbol{b}_{D'}, \boldsymbol{a}_{D'}),$$

where $\boldsymbol{b}_{D'}$ and $\boldsymbol{a}_{D'}$ are the restrictions of $\boldsymbol{b} \cup \boldsymbol{c}$ to $\partial D'$ and of $\boldsymbol{a}$ to $\boldsymbol{x}_{D'}$, respectively. By the induction hypothesis, for any $D' \prec D$, there is a homomorphism $h_{D'}\colon \boldsymbol{q}_{D'} \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that (5) is satisfied.

Let us show that the $h_{D'}$ agree on common variables. Suppose that v is shared by $\boldsymbol{q}_{D'}$ and $\boldsymbol{q}_{D''}$ for $D' \prec D$ and $D'' \prec D$. By the definition of tree decomposition, for every $v \in V$, the nodes $\{t \mid v \in \lambda(t)\}$ induce a connected subtree of T, and so $v \in \lambda(\sigma(D)) \cap \lambda(t') \cap \lambda(t'')$, where t' and t'' are the unique neighbours of $\sigma(D)$ lying in D' and D'', respectively. Since $\boldsymbol{w}' = (\boldsymbol{w} \cup \boldsymbol{s}) \restriction \partial D'$ and $\boldsymbol{w}'' = (\boldsymbol{w} \cup \boldsymbol{s}) \restriction \partial D''$ are the restrictions of $\boldsymbol{w} \cup \boldsymbol{s}$, we have $\boldsymbol{w}'(v) = \boldsymbol{w}''(v)$. This implies that $h_{D'}(v) = \boldsymbol{c}(v)\boldsymbol{w}'(v) = \boldsymbol{c}(v)\boldsymbol{w}''(v) = h_{D''}(v)$.

Now we define h on every v in $\boldsymbol{q}_D$ by taking

$$h(v) = \begin{cases} h_{D'}(v) & \text{if } v \in \lambda(t), \text{ for } t \in D' \text{ and } D' \prec D, \\ \boldsymbol{c}(v) \cdot (\boldsymbol{w} \cup \boldsymbol{s})(v), & \text{if } v \in \lambda(\sigma(D)). \end{cases}$$

If follows that h is well defined, h satisfies (5) and that h is a homomorphism from $\boldsymbol{q}_D$ to $\mathcal{C}_{\mathcal{T},\mathcal{A}}$. Indeed, take an atom $S(\boldsymbol{v}) \in \boldsymbol{q}_D$. Then either $\boldsymbol{v} \subseteq \lambda(\sigma(D))$, in which case $\mathcal{C}_{\mathcal{T},\mathcal{A}} \models S(h(\boldsymbol{v}))$ since $\boldsymbol{w}$ is compatible with $\sigma(D)$ and $\Pi_{\boldsymbol{Q}}, \mathcal{A} \models \mathsf{At}^{\boldsymbol{s}}(\boldsymbol{c})$, or $S(\boldsymbol{v}) \in \boldsymbol{q}_{D'}$ for some $D' \prec D$, in which case we use the fact that h extends a homomorphism $h_{D'}$.

($\Leftarrow$) The proof is by induction on $\prec$. Fix D and $\boldsymbol{w}$ such that $|\boldsymbol{w}| = |\partial D|$. Take $\boldsymbol{b} \in \mathsf{ind}(\mathcal{A})^{|\partial D|}$, $\boldsymbol{a} \in \mathsf{ind}(\mathcal{A})^{|\boldsymbol{x}_D|}$, and a homomorphism $h\colon \boldsymbol{q}_D \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ satisfying (5). Define a type $\boldsymbol{s}$ and a tuple $\boldsymbol{c} \in \mathsf{ind}(\mathcal{A})^{|\lambda(\sigma(D))|}$ by taking, for all $v \in \lambda(\sigma(D))$,

$$\boldsymbol{s}(v) = w \quad \text{and} \quad \boldsymbol{c}(v) = a, \qquad \text{if } h(v) = aw, \text{ for } a \in \mathsf{ind}(\mathcal{A}).$$

By definition, $\mathsf{dom}(\boldsymbol{s}) = \lambda(\sigma(D))$ and, by (5), $\boldsymbol{s}$ and $\boldsymbol{w}$ agree on the common domain. For the inductive step, for each $D' \prec D$, let $h_{D'}$ be the restriction of h to $\boldsymbol{q}_{D'}$ and let $\boldsymbol{b}_{D'}$ and and $\boldsymbol{a}_{D'}$ be the restrictions of $\boldsymbol{b} \cup \boldsymbol{c}$ to $\partial D'$ and of $\boldsymbol{a}$ to $\boldsymbol{x}_{D'}$, respectively. By the inductive hypothesis, $\Pi_{\boldsymbol{Q}}, \mathcal{A} \models G_{D'}^{\boldsymbol{w}'}(\boldsymbol{b}_{D'}, \boldsymbol{a}_{D'})$. (This argument is not needed for the basis of induction.) Since h is a homomorphism, we have $\Pi_{\boldsymbol{Q}}, \mathcal{A} \models \mathsf{At}^{\boldsymbol{s}}(\boldsymbol{c})$, whence, $\Pi_{\boldsymbol{Q}}, \mathcal{A} \models G_D^{\boldsymbol{w}}(\boldsymbol{b}, \boldsymbol{a})$. ❑

Lemma 11. *For any H-complete ABox $\mathcal{A}$, any $0 \leq n \leq M$, any predicate $P_n^{\boldsymbol{w}}$, any $\boldsymbol{b} \in \mathsf{ind}(\mathcal{A})^{|\boldsymbol{z}_\exists^n|}$ and any $\boldsymbol{a} \in \mathsf{ind}(\mathcal{A})^{|\boldsymbol{x}^n|}$, we have $\Pi'_{\boldsymbol{Q}}, \mathcal{A} \models P_n^{\boldsymbol{w}}(\boldsymbol{b}, \boldsymbol{a})$ iff there is a homomorphism $h\colon \boldsymbol{q}_n \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that*

$$h(x) = \boldsymbol{a}(x), \text{ for } x \in \boldsymbol{x}^n, \quad \text{and} \quad h(z) = \boldsymbol{b}(z)\boldsymbol{w}(z), \text{ for } z \in \boldsymbol{z}_\exists^n. \tag{4}$$

Proof. The proof is by induction on n. For the base case ($n = M$), first suppose that we have $\Pi'_{\boldsymbol{Q}}, \mathcal{A} \models P_M^{\boldsymbol{w}}(\boldsymbol{b}, \boldsymbol{a})$. The only rule in $\Pi'_{\boldsymbol{Q}}$ with head predicate $P_M^{\boldsymbol{w}}$ is $P_M^{\boldsymbol{w}}(\boldsymbol{z}_\exists^M, \boldsymbol{x}^M) \leftarrow \mathsf{At}^{\boldsymbol{w}}(\boldsymbol{z}^M)$ with $\boldsymbol{z}^M = \boldsymbol{z}_\exists^M \uplus \boldsymbol{x}^M$, which is equivalent to

$$P_M^{\boldsymbol{w}}(\boldsymbol{z}_\exists^M, \boldsymbol{x}^M) \leftarrow \bigwedge_{z \in \boldsymbol{z}^M} \Big(\bigwedge_{\substack{A(z) \in \boldsymbol{q} \\ \boldsymbol{w}(z) = \varepsilon}} A(z) \;\wedge \bigwedge_{\substack{R(z,z) \in \boldsymbol{q} \\ \boldsymbol{w}(z) = \varepsilon}} R(z,z) \;\wedge \bigwedge_{\boldsymbol{w}(z) = Sw'} A_S(z) \Big). \tag{6}$$

So the body of this rule must be satisfied when $\boldsymbol{b}$ and $\boldsymbol{a}$ are substituted for $\boldsymbol{z}_\exists^M$ and $\boldsymbol{x}^M$ respectively. Moreover, by local compatibility of $\boldsymbol{w}$ with $\boldsymbol{z}^M$, we know that $\boldsymbol{w}(x) = \varepsilon$ for every $x \in \boldsymbol{x}^M$. It follows that

- $A(\boldsymbol{a}(x)) \in \mathcal{A}$ for every $A(x) \in \boldsymbol{q}$ such that $x \in \boldsymbol{x}^M$;
- $A(\boldsymbol{b}(z)) \in \mathcal{A}$ for every $A(z) \in \boldsymbol{q}$ such that $z \in \boldsymbol{z}_\exists^M$ and $\boldsymbol{w}(z) = \varepsilon$;
- $R(\boldsymbol{a}(x), \boldsymbol{a}(x)) \in \mathcal{A}$ for every $R(x,x) \in \boldsymbol{q}$ such that $x \in \boldsymbol{x}^M$;
- $R(\boldsymbol{b}(z), \boldsymbol{b}(z)) \in \mathcal{A}$ for every $R(z,z) \in \boldsymbol{q}$ such that $z \in \boldsymbol{z}_\exists^M$ and $\boldsymbol{w}(z) = \varepsilon$;
- $A_S(z) \in \mathcal{A}$ for every $z \in \boldsymbol{z}^M$ with $\boldsymbol{w}(z) = Sw'$.

Now let h^M be the unique mapping from $\boldsymbol{z}^M$ to $\Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ satisfying (4). First note that h^M is well-defined, since by the last item, whenever $\boldsymbol{w}(z) = Sw'$, we have $A_S(z) \in \mathcal{A}$ and $Sw' \in \mathbf{W}_{\mathcal{T}}$, so $\boldsymbol{b}(z)Sw'$ belongs to $\Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$. To show that h^M is a homomorphism of $\boldsymbol{q}_M$ into $\mathcal{C}_{\mathcal{T},\mathcal{A}}$, first recall that the atoms of $\boldsymbol{q}_M$ are of two types: $A(z)$ or $R(z,z)$, with $z \in \boldsymbol{z}^M$. Take some $A(z) \in \boldsymbol{q}_M$. If $\boldsymbol{w}(z) = \varepsilon$, then we immediately obtain either

$A(h^M(z)) = A(\boldsymbol{a}(z)) \in \mathcal{A}$ or $A(h^M(z)) = A(\boldsymbol{b}(z)) \in \mathcal{A}$, depending on whether $z \in \boldsymbol{z}^M_\exists$ or in $\boldsymbol{x}^M$. Otherwise, if $\boldsymbol{w}(z) \neq \varepsilon$, then the local compatibility of $\boldsymbol{w}$ with $\boldsymbol{z}^M$ means that the final letter R in $\boldsymbol{w}(z)$ is such that $\exists R^- \sqsubseteq_\mathcal{T} A$, hence $h^M(z) = \boldsymbol{b}(z)\boldsymbol{w}(z) \in A^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$. Finally, suppose that $R(z,z) \in \boldsymbol{q}$. The local compatibility of $\boldsymbol{w}$ with $\boldsymbol{z}^M$ ensures that $\boldsymbol{w}(z) = \varepsilon$, and thus we have either $R(\boldsymbol{a}(z), \boldsymbol{a}(z)) \in \mathcal{A}$ or $R(\boldsymbol{b}(z), \boldsymbol{b}(z)) \in \mathcal{A}$, depending again on whether $z \in \boldsymbol{z}^M_\exists$ or $z \in \boldsymbol{x}^M$.

For the other direction of the base case, suppose that the mapping h^M given by (4) defines a homomorphism from $\boldsymbol{q}_M$ into $\mathcal{C}_{\mathcal{T},\mathcal{A}}$. We therefore have:

- $\boldsymbol{a}(x) \in A^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ for every $A(x) \in \boldsymbol{q}$ with $x \in \boldsymbol{x}^M$;
- $\boldsymbol{b}(z)\boldsymbol{w}(z) \in A^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ for every $A(z) \in \boldsymbol{q}$ with $z \in \boldsymbol{z}^M_\exists$;
- $(\boldsymbol{a}(x), \boldsymbol{a}(x)) \in R^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ for every $R(x,x) \in \boldsymbol{q}$ such that $x \in \boldsymbol{x}^M$;
- $(\boldsymbol{b}(z), \boldsymbol{b}(z)) \in R^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ for every $R(z,z) \in \boldsymbol{q}$ such that $z \in \boldsymbol{z}^M_\exists$;
- $\mathcal{T}, \mathcal{A} \models \exists S(\boldsymbol{b}(z))$ for every $z \in \boldsymbol{z}^M_\exists$ with $\boldsymbol{w}(z) = Sw'$ (for otherwise $\boldsymbol{b}(z)\boldsymbol{w}(z)$ would not belong to the domain of $\mathcal{C}_{\mathcal{T},\mathcal{A}}$).

The first two items, together with H-completeness of the ABox $\mathcal{A}$, ensure that all atoms in $\{A(z) \mid A(z) \in \boldsymbol{q}, z \in \boldsymbol{z}^M, \boldsymbol{w}(z) = \varepsilon\}$ are present in $\mathcal{A}$ when $\boldsymbol{b}$ and $\boldsymbol{a}$ substituted for $\boldsymbol{z}^M_\exists$ and $\boldsymbol{x}^M$ respectively. The third and fourth items, again together with H-completeness, ensure the presence of the atoms in $\{R(z,z) \mid R(z,z) \in \boldsymbol{q}, z \in \boldsymbol{z}^M, \boldsymbol{w}(z) = \varepsilon\}$. Finally, the fifth item plus H-completeness ensures that $\mathcal{A}$ contains all atoms in $\{A_S(z) \mid z \in \boldsymbol{z}^M, \boldsymbol{w}(z) = Sw'\}$. It follows that the body of the unique rule for $P^{\boldsymbol{w}}_M$ is satisfied when $\boldsymbol{b}$ and $\boldsymbol{a}$ are substituted for $\boldsymbol{z}^M_\exists$ and $\boldsymbol{x}^M$ respectively, and thus $\Pi'_{\boldsymbol{Q}}, \mathcal{A} \models P^{\boldsymbol{w}}_M(\boldsymbol{b}, \boldsymbol{a})$.

For the induction step, assume that the statement has been shown to hold for all $n \leq k+1 \leq M$, and let us show that it holds when $n = k$. For the first direction, suppose $\Pi'_{\boldsymbol{Q}}, \mathcal{A} \models P^{\boldsymbol{w}}_k(\boldsymbol{b}, \boldsymbol{a})$. It follows that there exists a pair of types $(\boldsymbol{w}, \boldsymbol{s})$ compatible with $(\boldsymbol{z}^k, \boldsymbol{z}^{k+1})$ and an assignment $\boldsymbol{c}$ of individuals from $\mathcal{A}$ to the variables in $\boldsymbol{z}^k \cup \boldsymbol{z}^{k+1}$ such that $\boldsymbol{c}(x) = \boldsymbol{a}(x)$ for all $x \in (\boldsymbol{z}^k \cup \boldsymbol{z}^{k+1}) \cap \boldsymbol{x}$, and $\boldsymbol{c}(z) = \boldsymbol{b}(z)$ for all $z \in \boldsymbol{z}^k_\exists$, and such that every atom in the body of the clause

$$P^{\boldsymbol{w}}_k(\boldsymbol{z}^k_\exists, \boldsymbol{x}^k) \leftarrow \mathsf{At}^{\boldsymbol{w}\cup\boldsymbol{s}}(\boldsymbol{z}^k, \boldsymbol{z}^{k+1}) \wedge P^{\boldsymbol{s}}_{k+1}(\boldsymbol{z}^{k+1}_\exists, \boldsymbol{x}^{k+1})$$

is entailed from $\Pi'_{\boldsymbol{Q}}, \mathcal{A}$ when the individuals in $\boldsymbol{c}$ are substituted for $\boldsymbol{z}^k \cup \boldsymbol{z}^{k+1}$. We recall that $\mathsf{At}^{\boldsymbol{w}\cup\boldsymbol{s}}(\boldsymbol{z}^k, \boldsymbol{z}^{k+1})$ is the conjunction of the following atoms, for $z, z' \in \boldsymbol{z}^k \cup \boldsymbol{z}^{k+1}$:

- $A(z)$, if $A(z) \in \boldsymbol{q}$ and $(\boldsymbol{w} \cup \boldsymbol{s})(z) = \varepsilon$,
- $R(z,z')$, if $R(z,z') \in \boldsymbol{q}$ and $(\boldsymbol{w} \cup \boldsymbol{s})(z) = (\boldsymbol{w} \cup \boldsymbol{s})(z') = \varepsilon$,
- $z = z'$, if $R(z,z') \in \boldsymbol{q}$ and either $(\boldsymbol{w} \cup \boldsymbol{s})(z) \neq \varepsilon$ or $(\boldsymbol{w} \cup \boldsymbol{s})(z') \neq \varepsilon$,
- $A_S(z)$, if $(\boldsymbol{w} \cup \boldsymbol{s})(z)$ is of the form Sw'.

In particular, we have $\Pi'_{\boldsymbol{Q}}, \mathcal{A} \models P^{\boldsymbol{s}}_{k+1}(\boldsymbol{c}(\boldsymbol{z}^{k+1}_\exists), \boldsymbol{c}(\boldsymbol{x}^{k+1}))$. By the induction hypothesis, there exists a homomorphism $h^{k+1}\colon \boldsymbol{q}_{k+1} \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that $h^{k+1}(u) = \boldsymbol{c}(u)\boldsymbol{s}(u)$ for every $u \in \boldsymbol{z}^{k+1}_\exists \cup \boldsymbol{x}^{k+1}$. Define a mapping h^k from $\mathsf{var}(\boldsymbol{q}_k)$ to $\Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ by setting $h^k(u) = h^{k+1}(u)$ for every variable $u \in \mathsf{var}(\boldsymbol{q}_{k+1})$, setting $h^k(x) = \boldsymbol{a}(x)$ for every $x \in \boldsymbol{z}^k \cap \boldsymbol{x}$, and setting $h^k(z) = \boldsymbol{b}(z)\boldsymbol{w}(z)$ for every $z \in \boldsymbol{z}^k$. Using the same argument

as was used in the base case, we can show that h^k is well-defined. For atoms from $\boldsymbol{q}_k$ involving only variables from $\boldsymbol{q}_{k+1}$, we can use the induction hypothesis to conclude that they are satisfied under h^k, and for atoms only involving variables from $\boldsymbol{z}^k$, we can argue as in the base case. It thus remains to handle role atoms that contain one variable from $\boldsymbol{z}^k$ and one variable from $\boldsymbol{z}^{k+1}$. Consider such an atom $R(z, z') \in \boldsymbol{q}_k$, for $z \in \boldsymbol{z}^k$ and $z' \in \boldsymbol{z}^{k+1}$. If $\boldsymbol{w}(z) = \boldsymbol{s}(z') = \varepsilon$, then the atom $R(z, z')$ appears in the body of the clause we are considering. It follows that $\Pi'_{\boldsymbol{Q}}, \mathcal{A} \models R(\boldsymbol{c}(z), \boldsymbol{c}(z'))$, hence $(\boldsymbol{c}(z), \boldsymbol{c}(z')) \in R^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$. It then suffices to note that $\boldsymbol{c}$ agrees with $\boldsymbol{a}$ and $\boldsymbol{b}$ on the variables in $\boldsymbol{z}^k$. Next suppose that either $\boldsymbol{w}(z) \neq \varepsilon$ or $\boldsymbol{s}(z') \neq \varepsilon$. It follows that the clause body contains $z = z'$, hence $\boldsymbol{c}(z) = \boldsymbol{c}(z')$. As $(\boldsymbol{w}, \boldsymbol{s})$ is compatible with $(\boldsymbol{z}^k, \boldsymbol{z}^{k+1})$, one of the following must hold: (a) $\boldsymbol{s}(z') = \boldsymbol{w}(z)R'$ with $R' \sqsubseteq_{\mathcal{T}} R$, or (b) $\boldsymbol{w}(z) = \boldsymbol{s}(z')R'$ with $R' \sqsubseteq_{\mathcal{T}} R^-$. We give the argument in the case where $z \in \boldsymbol{z}^k_\exists$ (the argument is entirely similar if $z \in \boldsymbol{x}^k$). If (a) holds, then

$$(h^k(z), h^k(z')) = (\boldsymbol{b}(z)\boldsymbol{w}(z), \boldsymbol{c}(z')\boldsymbol{s}(z')) = (\boldsymbol{b}(z)\boldsymbol{w}(z), \boldsymbol{c}(z')\boldsymbol{w}(z)R') \in R^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$$

since $R' \sqsubseteq_{\mathcal{T}} R$ and $\boldsymbol{c}(z') = \boldsymbol{c}(z) = \boldsymbol{b}(z)$. If (b) holds, then

$$(h^k(z), h^k(z')) = (\boldsymbol{b}(z)\boldsymbol{w}(z), \boldsymbol{c}(z')\boldsymbol{s}(z')) = (\boldsymbol{b}(z)\boldsymbol{s}(z')R', \boldsymbol{c}(z')\boldsymbol{s}(z')) \in R^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$$

since $R' \sqsubseteq_{\mathcal{T}} R^-$.

For the converse direction of the induction step, let $\boldsymbol{w}$ be a type that is locally compatible with $\boldsymbol{z}^k$, let $\boldsymbol{a} \in \mathsf{ind}(\mathcal{A})^{|\boldsymbol{x}^k|}$ and $\boldsymbol{b} \in \mathsf{ind}(\mathcal{A})^{|\boldsymbol{z}^k_\exists|}$, and let $h^k \colon \boldsymbol{q}_k \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ be a homomorphism satisfying

$$h^k(x) = \boldsymbol{a}(x), \text{ for } x \in \boldsymbol{x}^k, \quad \text{and} \quad h^k(z) = \boldsymbol{b}(z)\boldsymbol{w}(z), \text{ for } z \in \boldsymbol{z}^k_\exists. \tag{7}$$

We let $\boldsymbol{c}$ for $\boldsymbol{z}^{k+1}$ be defined by setting $\boldsymbol{c}(z)$ equal to the unique individual c such that $h(z)$ is of the form cw (for some $w \in \mathbf{W}_{\mathcal{T}}$), and let $\boldsymbol{s}$ be the unique type for $\boldsymbol{z}^{k+1}$ satisfying $h(z) = \boldsymbol{c}(z)\boldsymbol{s}(z)$ for every $z \in \boldsymbol{z}^{k+1}$; in other words, we obtain $\boldsymbol{s}(z)$ from $h(z)$ by omitting the initial individual name $\boldsymbol{c}(z)$. Note that since $\boldsymbol{x}^{k+1} \subseteq \boldsymbol{x}^k$, we have $\boldsymbol{a}(x) = \boldsymbol{c}(x)$ for every $x \in \boldsymbol{x}^{k+1}$. It follows from the fact that h^k is a homomorphism that $\boldsymbol{s}$ is locally compatible with $\boldsymbol{z}^{k+1}$ and that, for every role atom $R(z, z') \in \boldsymbol{q}_k$ with $z \in \boldsymbol{z}^k$ and $z' \in \boldsymbol{z}^{k+1}$, one of the following holds: (*i*) $\boldsymbol{w}(z) = \boldsymbol{s}(z') = \varepsilon$, (*ii*) $\boldsymbol{s}(z') = \boldsymbol{w}(z)R'$ with $R' \sqsubseteq_{\mathcal{T}} R$, or (*iii*) $\boldsymbol{w}(z) = \boldsymbol{s}(z')R'$ with $R' \sqsubseteq_{\mathcal{T}} R^-$. Thus, the pair of types $(\boldsymbol{w}, \boldsymbol{s})$ is compatible with $(\boldsymbol{z}^k, \boldsymbol{z}^{k+1})$, and so the following rule appears in $\Pi'_{\boldsymbol{Q}}$:

$$P^{\boldsymbol{w}}_k(\boldsymbol{z}^k_\exists, \boldsymbol{x}^k) \leftarrow \mathsf{At}^{\boldsymbol{w}\cup\boldsymbol{s}}(\boldsymbol{z}^k, \boldsymbol{z}^{k+1}) \wedge P^{\boldsymbol{s}}_{k+1}(\boldsymbol{z}^{k+1}_\exists, \boldsymbol{x}^{k+1}),$$

where we recall that $\mathsf{At}^{\boldsymbol{w}\cup\boldsymbol{s}}(\boldsymbol{z}^k, \boldsymbol{z}^{k+1})$ is the conjunction of the following atoms, for $z, z' \in \boldsymbol{z}^k \cup \boldsymbol{z}^{k+1}$:

- $A(z)$, if $A(z) \in \boldsymbol{q}$ and $(\boldsymbol{w} \cup \boldsymbol{s})(z) = \varepsilon$,
- $R(z, z')$, if $R(z, z') \in \boldsymbol{q}$ and $(\boldsymbol{w} \cup \boldsymbol{s})(z) = (\boldsymbol{w} \cup \boldsymbol{s})(z') = \varepsilon$,
- $z = z'$, if $R(z, z') \in \boldsymbol{q}$ and either $(\boldsymbol{w} \cup \boldsymbol{s})(z) \neq \varepsilon$ or $(\boldsymbol{w} \cup \boldsymbol{s})(z') \neq \varepsilon$,
- $A_S(z)$, if $(\boldsymbol{w} \cup \boldsymbol{s})(z)$ is of the form Sw'.

It follows from Equation (7) and the fact that h^k is a homomorphism that each of the ground atoms obtained by taking an atom from $\mathsf{At}^{\boldsymbol{w}\cup\boldsymbol{s}}(\boldsymbol{z}^k, \boldsymbol{z}^{k+1})$ and substituting $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ for $\boldsymbol{x}^k$, $\boldsymbol{z}^k_\exists$ and $\boldsymbol{z}^{k+1}$, respectively, is present in $\mathcal{A}$. By applying the induction hypothesis to the predicate $P^{\boldsymbol{s}}_{k+1}$ and the homomorphism $h^{k+1}\colon \boldsymbol{q}_{k+1} \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ obtained by restricting h^k to $\mathsf{var}(\boldsymbol{q}_{k+1})$, we obtain that $\Pi'_{\boldsymbol{Q}}, \mathcal{A} \models P^{\boldsymbol{s}}_{k+1}(\boldsymbol{c}(\boldsymbol{z}^{k+1}_\exists), \boldsymbol{a}(\boldsymbol{x}^{k+1}))$. Since for the considered substitution, all body atoms are entailed, we can conclude that $\Pi'_{\boldsymbol{Q}}, \mathcal{A} \models P^{\boldsymbol{w}}_k(\boldsymbol{b}, \boldsymbol{a})$. ❑

Lemma 13. *For any tree-shaped OMQ $\boldsymbol{Q}(\boldsymbol{x}) = (\mathcal{T}, \boldsymbol{q}_0(\boldsymbol{x}))$, any $\boldsymbol{q}(\boldsymbol{z}) \in \mathsf{SQ}$, any H-complete ABox $\mathcal{A}$, and any tuple $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{A})$, $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}}(\boldsymbol{a})$ iff there exists a homomorphism $h\colon \boldsymbol{q} \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that $h(\boldsymbol{z}) = \boldsymbol{a}$.*

Proof. An inspection of the definition of the set SQ shows that every $\boldsymbol{q}(\boldsymbol{z}) \in \mathsf{SQ}$ is a tree-shaped query having at least one answer variable, with the possible exception of the original query $\boldsymbol{q}_0(\boldsymbol{x})$, which may be Boolean.

Just as we did for subtrees in Section 4, we associate a binary relation on the queries in SQ by setting $\boldsymbol{q}'(\boldsymbol{z}') \prec \boldsymbol{q}(\boldsymbol{z})$ whenever $\boldsymbol{q}'(\boldsymbol{z}')$ was introduced when applying one of the two decomposition conditions on p. 10 to $\boldsymbol{q}(\boldsymbol{z})$. The proof is by induction on the subqueries in SQ, according to $\prec$. We will start by establishing the statement for all queries in SQ other than $\boldsymbol{q}_0(\boldsymbol{x})$, and afterwards, we will complete the proof by giving an argument for $\boldsymbol{q}_0(\boldsymbol{x})$.

For the basis of induction, take some $\boldsymbol{q}(\boldsymbol{z}) \in \mathsf{SQ}$ that is minimal in the ordering induced by $\prec$, which means that $\mathsf{var}(\boldsymbol{q}) = \boldsymbol{z}$. Indeed, if there is an existentially quantified variable, then the first decomposition rule will give rise to a 'smaller' query (in particular, if $|\mathsf{var}(\boldsymbol{q})| = 2$, then although the 'smaller' query may have the same atoms, the selected existential variable will become an answer variable). For the first direction, suppose that $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}}(a)$. By definition, $P_{\boldsymbol{q}}(z) \leftarrow \boldsymbol{q}(z)$ is the only clause with head predicate $P_{\boldsymbol{q}}$. Thus, all atoms in the ground CQ $\boldsymbol{q}(\boldsymbol{a})$ are present in $\mathcal{A}$, and hence the desired homomorphism exists. For the converse direction, suppose there is a homomorphism $h\colon \boldsymbol{q}(\boldsymbol{z}) \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that $h(\boldsymbol{z}) = \boldsymbol{a}$. It follows that every atom in the ground CQ $\boldsymbol{q}(\boldsymbol{a})$ is entailed from $\mathcal{T}, \mathcal{A}$. H-completeness of $\mathcal{A}$ ensures that all of the ground atoms in $\boldsymbol{q}(\boldsymbol{a})$ are present in $\mathcal{A}$, and thus we can apply the clause $P_{\boldsymbol{q}}(\boldsymbol{z}) \leftarrow \boldsymbol{q}(\boldsymbol{z})$ to derive $P_{\boldsymbol{q}}(\boldsymbol{a})$.

For the induction step, consider $\boldsymbol{q}(\boldsymbol{z}) \in \mathsf{SQ}$ with $\mathsf{var}(\boldsymbol{q}) \neq \boldsymbol{z}$ and suppose that the claim holds for all $\boldsymbol{q}'(\boldsymbol{z}') \in \mathsf{SQ}$ with $\boldsymbol{q}'(\boldsymbol{z}') \prec \boldsymbol{q}(\boldsymbol{z})$. For the first direction, let $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}}(\boldsymbol{a})$. There are two cases, depending on which type of clause was used to derive $P_{\boldsymbol{q}}(\boldsymbol{a})$.

– Case 1: $P_{\boldsymbol{q}}(\boldsymbol{a})$ was derived by an application of the following clause:

$$P_{\boldsymbol{q}}(\boldsymbol{z}) \quad \leftarrow \bigwedge_{A(v_{\boldsymbol{q}})\in\boldsymbol{q}} A(v_{\boldsymbol{q}}) \quad \wedge \bigwedge_{R(v_{\boldsymbol{q}},v_{\boldsymbol{q}})\in\boldsymbol{q}} R(v_{\boldsymbol{q}}, v_{\boldsymbol{q}}) \quad \wedge \bigwedge_{1\le i\le n} P_{\boldsymbol{q}_i}(\boldsymbol{z}_i),$$

where $\boldsymbol{q}_1(\boldsymbol{z}_1), \ldots, \boldsymbol{q}_n(\boldsymbol{z}_n)$ are the subqueries induced by the neighbours of $v_{\boldsymbol{q}}$ in the Gaifman graph $\mathcal{G}$ of $\boldsymbol{q}$. Then there exists a substitution $\boldsymbol{c}$ for the variables in the body of this rule that coincides with $\boldsymbol{a}$ on $\boldsymbol{z}$ and is such that the ground atoms obtained by applying $\boldsymbol{c}$ to the variables in the body are all entailed from

$\Pi''_{\boldsymbol{Q}}, \mathcal{A}$. In particular, $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}_i}(\boldsymbol{c}(\boldsymbol{z}_i))$ for every $1 \leq i \leq n$. We can apply the induction hypothesis to the $\boldsymbol{q}_i(\boldsymbol{z}_i)$ to obtain homomorphisms $h_i \colon \boldsymbol{q}_i \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that $h_i(\boldsymbol{z}_i) = \boldsymbol{c}(\boldsymbol{z}_i)$. Let h be the mapping from $\mathsf{var}(\boldsymbol{q})$ to $\Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ defined by taking $h(v) = h_i(v)$, for $v \in \mathsf{var}(\boldsymbol{q}_i)$. Note that h is well-defined since $\mathsf{var}(\boldsymbol{q}) = \bigcup_{i=1}^{n} \mathsf{var}(\boldsymbol{q}_i)$, and the $\boldsymbol{q}_i$ have no variable in common other than $v_{\boldsymbol{q}}$, which is sent to $\boldsymbol{c}(v_{\boldsymbol{q}})$ by every h_i. To see why h is a homomorphism from $\boldsymbol{q}$ to $\mathcal{C}_{\mathcal{T},\mathcal{A}}$, observe that $\boldsymbol{q} = \bigcup_{i=1}^{n} \boldsymbol{q}_i \cup \{A(v_{\boldsymbol{q}}) \in \boldsymbol{q}\} \cup \{R(v_{\boldsymbol{q}}, v_{\boldsymbol{q}}) \in \boldsymbol{q}\}$. By the definition of h, all atoms in $\bigcup_{i=1}^{n} \boldsymbol{q}_i$ hold under h. If $A(v_{\boldsymbol{q}}) \in \boldsymbol{q}$, then $A(\boldsymbol{c}(v_{\boldsymbol{q}}))$ is entailed from $\Pi''_{\boldsymbol{Q}}, \mathcal{A}$, and hence is present in $\mathcal{A}$. Similarly, we can show that for every $R(v_{\boldsymbol{q}}, v_{\boldsymbol{q}}) \in \boldsymbol{q}$, the ground atom $R(\boldsymbol{c}(v_{\boldsymbol{q}}), \boldsymbol{c}(v_{\boldsymbol{q}}))$ belongs to $\mathcal{A}$. It follows that all of these atoms hold in $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ under h. Finally, we recall that $\boldsymbol{c}$ coincides with $\boldsymbol{a}$ on $\boldsymbol{z}$, so we have $h(\boldsymbol{z}) = \boldsymbol{a}$, as required.

– Case 2: $P_{\boldsymbol{q}}(\boldsymbol{a})$ was derived by an application of the following clause, for a tree witness t for $(\mathcal{T}, \boldsymbol{q}(\boldsymbol{z}))$ with $\mathsf{t}_\mathsf{r} \neq \emptyset$ and $v_{\boldsymbol{q}} \in \mathsf{t}_\mathsf{i}$ and role R generating t:

$$P_{\boldsymbol{q}}(\boldsymbol{z}) \quad \leftarrow \bigwedge_{u,u' \in \mathsf{t}_\mathsf{r}} (u = u') \quad \wedge \bigwedge_{u \in \mathsf{t}_\mathsf{r}} A_R(u) \quad \wedge \bigwedge_{1 \leq i \leq m} P_{\boldsymbol{q}_i^\mathsf{t}}(\boldsymbol{z}_i^\mathsf{t}),$$

where $\boldsymbol{q}_1^\mathsf{t}, \ldots, \boldsymbol{q}_m^\mathsf{t}$ are the connected components of $\boldsymbol{q}$ without $\boldsymbol{q}_\mathsf{t}$. There must exist a substitution $\boldsymbol{c}$ for the variables in the body of this rule that coincides with $\boldsymbol{a}$ on $\boldsymbol{z}$ and is such that the ground atoms obtained by applying $\boldsymbol{c}$ to the variables in the body are all entailed from $\Pi''_{\boldsymbol{Q}}, \mathcal{A}$. In particular, for every $1 \leq i \leq m$, we have $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}_i^\mathsf{t}}(\boldsymbol{c}(\boldsymbol{z}_i^\mathsf{t}))$. We can apply the induction hypothesis to the $\boldsymbol{q}_i^\mathsf{t}(\boldsymbol{z}_i^\mathsf{t})$ to find homomorphisms $h_1, \ldots, h_m$ of $\boldsymbol{q}_1^\mathsf{t}, \ldots, \boldsymbol{q}_m^\mathsf{t}$ into $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that $h_i(\boldsymbol{z}_i^\mathsf{t}) = \boldsymbol{c}(\boldsymbol{z}_i^\mathsf{t})$. Since t is a tree witness for $(\mathcal{T}, \boldsymbol{q}(\boldsymbol{z}))$ generated by R, there exists a homomorphism h_t of $\boldsymbol{q}_\mathsf{t}$ into $\mathcal{C}_{\mathcal{T}}^{A_R(a)}$ with $\mathsf{t}_\mathsf{r} = h_\mathsf{t}^{-1}(a)$ and such that $h_\mathsf{t}(v)$ begins by aR for every $v \in \mathsf{t}_\mathsf{i}$. Now pick some $u_0 \in \mathsf{t}_\mathsf{r}$ (recall that $\mathsf{t}_\mathsf{r} \neq \emptyset$). Then $A_R(u_0)$ is an atom in the clause body, and so $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models A_R(\boldsymbol{c}(u_0))$, which means that $A_R(\boldsymbol{c}(u_0))$ must appear in $\mathcal{A}$. It follows that for every element in $\mathcal{C}_{\mathcal{T}}^{A_R(a)}$ of the form aRw, there exists a corresponding element $\boldsymbol{c}(u_0)Rw$ in $\Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$. We now define a mapping h from $\mathsf{var}(\boldsymbol{q})$ to $\Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ as follows:

$$h(v) = \begin{cases} h_i(v), & \text{for every } v \in \mathsf{var}(\boldsymbol{q}_i^\mathsf{t}), \\ \boldsymbol{c}(u_0)Rw, & \text{if } v \in \mathsf{t}_\mathsf{i} \text{ and } h_\mathsf{t}(v) = aRw, \\ \boldsymbol{c}(u_0) & \text{if } v \in \mathsf{t}_\mathsf{r}. \end{cases}$$

Every variable in $\mathsf{var}(\boldsymbol{q})$ occurs in $\mathsf{t}_\mathsf{r} \cup \mathsf{t}_\mathsf{i}$ or in exactly one of the $\boldsymbol{q}_i^\mathsf{t}$, and so is assigned a unique value by h. Note that although $\mathsf{t}_\mathsf{r} \cap \mathsf{var}(\boldsymbol{q}_i^\mathsf{t})$ is not necessarily empty, due to the equality atoms, we have $h(v) = h(v')$, for all $v, v' \in \mathsf{t}_\mathsf{r}$, and so the function is well-defined. We claim that h is a homomorphism from $\boldsymbol{q}$ into $\mathcal{C}_{\mathcal{T},\mathcal{A}}$. Clearly, the atoms occurring in some $\boldsymbol{q}_i^\mathsf{t}$ are preserved under h. Now consider some $A(v)$ with $v \in \mathsf{t}_\mathsf{i}$. Then $h(v) = \boldsymbol{c}(u_0)Rw$, where $h_\mathsf{t}(v) = aRw$. Since h_t is a homomorphism, we know that w ends with a role S such that $\exists S^- \sqsubseteq_\mathcal{T} A$. It follows that $h(v)$ also ends with S, and thus $h(v) \in A^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$. Next, consider a role atom $S(v, v')$, where at least one of v and v' belongs to t_i. As h_t is a homomorphism, either $h_\mathsf{t}(v') = h_\mathsf{t}(v)S'$ with $S' \sqsubseteq_\mathcal{T} S$, or $h_\mathsf{t}(v) = h_\mathsf{t}(v')S'$ with $S' \sqsubseteq_\mathcal{T} S^-$, for

some S'. We also know that $\boldsymbol{c}(u) = \boldsymbol{c}(u_0)$ for all $u \in \mathsf{t_r}$, hence $h(u) = h(u_0)$ for all $u \in \mathsf{t_r}$. It follows that either $h(v') = h(v)S'$ with $S' \sqsubseteq_{\mathcal{T}} S$, or $h(v) = h(v')S'$ with $S' \sqsubseteq_{\mathcal{T}} S^-$, and so $S(v, v')$ is preserved under h. Finally, since $\boldsymbol{c}$ coincides with $\boldsymbol{a}$ on $\boldsymbol{z}$, we have $h(\boldsymbol{z}) = \boldsymbol{a}$.

For the converse direction of the induction step, suppose that h is a homomorphism of $\boldsymbol{q}$ into $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that $h(\boldsymbol{z}) = \boldsymbol{a}$. There are two cases to consider, depending on where h maps the 'splitting' variable $v_{\boldsymbol{q}}$.

- Case 1: $h(v_{\boldsymbol{q}}) \in \mathsf{ind}(\mathcal{A})$. In this case, let $\boldsymbol{q}_1(\boldsymbol{z}_1), \dots, \boldsymbol{q}_n(\boldsymbol{z}_n)$ be the subqueries of $\boldsymbol{q}(\boldsymbol{z})$ induced by the neighbours of $v_{\boldsymbol{q}}$ in $\mathcal{G}$. Recall that $\boldsymbol{z}_i$ consists of $v_{\boldsymbol{q}}$ and the variables in $\mathsf{var}(\boldsymbol{q}_i) \cap \boldsymbol{z}$. By restricting h to $\mathsf{var}(\boldsymbol{q}_i)$, we obtain, for each $1 \leq i \leq n$, a homomorphism of $\boldsymbol{q}_i(\boldsymbol{z}_i)$ into $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ that maps $v_{\boldsymbol{q}}$ to $h(v_{\boldsymbol{q}})$ and $\mathsf{var}(\boldsymbol{q}_i) \cap \boldsymbol{z}$ to $\boldsymbol{a}(\mathsf{var}(\boldsymbol{q}_i) \cap \boldsymbol{z})$. Consider $\boldsymbol{a}^*$ defined by taking $\boldsymbol{a}^*(z) = \boldsymbol{a}(z)$ for every $z \in \mathsf{var}(\boldsymbol{q}_i) \cap \boldsymbol{z}$ and $\boldsymbol{a}^*(v_{\boldsymbol{q}}) = h(v_{\boldsymbol{q}})$. By the induction hypothesis, for every $1 \leq i \leq n$, we have $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}_i}(\boldsymbol{a}^*(\boldsymbol{z}_i))$. Next, since h is a homomorphism, we must have $h(v_{\boldsymbol{q}}) \in A^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ whenever $A(v_{\boldsymbol{q}}) \in \boldsymbol{q}$ and $(h(v_{\boldsymbol{q}}), h(v_{\boldsymbol{q}})) \in R^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ whenever $R(v_{\boldsymbol{q}}, v_{\boldsymbol{q}}) \in \boldsymbol{q}$. Since $\mathcal{A}$ is H-complete, $A(h(v_{\boldsymbol{q}})) \in \mathcal{A}$ for every $A(v_{\boldsymbol{q}}) \in \boldsymbol{q}$ and $R(h(v_{\boldsymbol{q}}), h(v_{\boldsymbol{q}}))$ for every $R(v_{\boldsymbol{q}}, v_{\boldsymbol{q}}) \in \boldsymbol{q}$. We have thus shown that, under the substitution $\boldsymbol{a}^*$, every atom in the body of the clause

$$P_{\boldsymbol{q}}(\boldsymbol{z}) \quad \leftarrow \bigwedge_{A(v_{\boldsymbol{q}}) \in \boldsymbol{q}} A(v_{\boldsymbol{q}}) \quad \wedge \bigwedge_{R(v_{\boldsymbol{q}}, v_{\boldsymbol{q}}) \in \boldsymbol{q}} R(v_{\boldsymbol{q}}, v_{\boldsymbol{q}}) \quad \wedge \bigwedge_{1 \leq i \leq n} P_{\boldsymbol{q}_i}(\boldsymbol{z}_i)$$

is entailed from $\Pi''_{\boldsymbol{Q}}, \mathcal{A}$. It follows that we must also have $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}}(\boldsymbol{a})$.

- Case 2: $h(v_{\boldsymbol{q}}) \notin \mathsf{ind}(\mathcal{A})$. Then $h(v_{\boldsymbol{q}})$ is of the form bRw. Let V be the smallest subset of $\mathsf{var}(\boldsymbol{q})$ that contains $v_{\boldsymbol{q}}$ and satisfies the following closure property:
  - if $v \in V$, $h(v) \notin \mathsf{ind}(\mathcal{A})$ and $\boldsymbol{q}$ contains an atom with v and v', then $v' \in V$.

  Let V' consist of all variables in V such that $h(v) \notin \mathsf{ind}(\mathcal{A})$. We observe that $h(v)$ begins by bR for every $v \in V'$ and $h(v) = b$ for every $v \in V \setminus V'$. Define $\boldsymbol{q}_V$ as the CQ comprising all atoms in $\boldsymbol{q}$ whose variables are in V and which contain at least one variable from V'; the answer variables of $\boldsymbol{q}_V$ are $V \setminus V'$. By replacing the initial b by a in the mapping h, we obtain a homomorphism h_V of $\boldsymbol{q}_V$ into $\mathcal{C}_{\mathcal{T}}^{A_R(a)}$ with $V \setminus V' = h_V^{-1}(a)$. It follows that $\mathsf{t} = (\mathsf{t_r}, \mathsf{t_i})$ with $\mathsf{t_r} = V \setminus V'$ and $\mathsf{t_i} = V'$ is a tree witness for $(\mathcal{T}, \boldsymbol{q}(\boldsymbol{z}))$ generated by R (and $\boldsymbol{q}_{\mathsf{t}} = \boldsymbol{q}_V$). Moreover, $\mathsf{t_r} \neq \emptyset$ because $\boldsymbol{q}$ has at least one answer variable. This means that the program $\Pi''_{\boldsymbol{Q}}$ contains the following clause

$$P_{\boldsymbol{q}}(\boldsymbol{z}) \quad \leftarrow \bigwedge_{u, u' \in \mathsf{t_r}} (u = u') \quad \wedge \bigwedge_{u \in \mathsf{t_r}} A_R(u) \quad \wedge \bigwedge_{1 \leq i \leq m} P_{\boldsymbol{q}^{\mathsf{t}}_i}(\boldsymbol{z}^{\mathsf{t}}_i),$$

where $\boldsymbol{q}^{\mathsf{t}}_1, \dots, \boldsymbol{q}^{\mathsf{t}}_m$ are the connected components of $\boldsymbol{q}$ without $\boldsymbol{q}_{\mathsf{t}}$. Recall that the query $\boldsymbol{q}^{\mathsf{t}}_i$ has answer variables $\boldsymbol{z}^{\mathsf{t}}_i = \mathsf{var}(\boldsymbol{q}^{\mathsf{t}}_i) \cap (\boldsymbol{z} \cup \mathsf{t_r})$. Let $\boldsymbol{a}^*$ be the substitution for $\boldsymbol{z} \cup \mathsf{t_r}$ such that $\boldsymbol{a}^*(z) = \boldsymbol{a}(z)$ for $z \in \boldsymbol{z}$ and $\boldsymbol{a}^*(v) = h(v)$ for $v \in \mathsf{t_r}$. Then, for every $1 \leq i \leq m$, there exists a homomorphism h_i from $\boldsymbol{q}^{\mathsf{t}}_i$ to $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that $h_i(z) = \boldsymbol{a}^*(z)$ for every $z \in \boldsymbol{z}^{\mathsf{t}}_i$. By the induction hypothesis, $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}^{\mathsf{t}}_i}(\boldsymbol{a}^*(\boldsymbol{z}^{\mathsf{t}}_i))$. Next, since $h(v) = b$ for every $v \in \mathsf{t_r}$, we have

$\boldsymbol{a}^*(u) = \boldsymbol{a}^*(u')$ for every $u, u' \in \mathsf{t_r}$. Moreover, the presence of the element bR in $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ means that $\mathcal{T}, \mathcal{A} \models A_R(b)$. Since $\mathcal{A}$ is H-complete, we have $A_R(b) \in \mathcal{A}$. It follows that under the substitution $\boldsymbol{a}^*$, all atoms in the body of the clause under consideration are entailed by $\Pi''_{\boldsymbol{Q}}, \mathcal{A}$. Thus, we must also have $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}}(\boldsymbol{a})$.

We have thus shown the lemma for all queries SQ other than $\boldsymbol{q}_0(\boldsymbol{x})$. Let us now turn to $\boldsymbol{q}_0(\boldsymbol{x})$. For the first direction, suppose $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}_0}(\boldsymbol{a})$. There are four cases, depending on which type of clause was used to derive $P_{\boldsymbol{q}}(\boldsymbol{a})$. We skip the first three cases, which are identical to those considered in the base case and induction step, and focus instead on the case in which $P_{\boldsymbol{q}_0}(\boldsymbol{a})$ was derived using a clause of the form $P_{\boldsymbol{q}_0} \leftarrow A(x)$ with A a concept name such that $\mathcal{T}, \{A(a)\} \models \boldsymbol{q}_0$. In this case, there must exist some $b \in \mathsf{ind}(\mathcal{A})$ such that $\mathcal{T}, \mathcal{A} \models A(b)$. By H-completeness of $\mathcal{A}$, we obtain $A(b) \in \mathcal{A}$. Since $\mathcal{T}, \{A(a)\} \models \boldsymbol{q}_0$, we get $\mathcal{T}, \mathcal{A} \models \boldsymbol{q}_0$, which implies the existence of a homomorphism from $\boldsymbol{q}_0$ into $\mathcal{C}_{\mathcal{T},\mathcal{A}}$.

For the converse direction, suppose that there is a homomorphism $h\colon \boldsymbol{q}_0 \to \mathcal{C}_{\mathcal{T},\mathcal{A}}$ such that $h(\boldsymbol{x}) = \boldsymbol{a}$. We focus on the case in which $\boldsymbol{q}_0$ is Boolean ($\boldsymbol{x} = \emptyset$) and none of the variables in $\boldsymbol{q}_0$ is mapped to an ABox individual (the other cases can be handled exactly as in the induction basis and induction step). In this case, there must exist an individual b and role R such that $h(z)$ begins by bR for every $z \in \mathsf{var}(\boldsymbol{q})$. It follows that $\mathcal{T}, \{A_R(a)\} \models \boldsymbol{q}_0$, since the mapping h' defined by setting $h'(z) = aRw$ whenever $h(z) = bRw$ is a homomorphism from $\boldsymbol{q}$ to $\mathcal{C}_{\mathcal{T},\{A_R(a)\}}$. It follows that $\Pi''_{\boldsymbol{Q}}$ contains the clause $P_{\boldsymbol{q}_0} \leftarrow A_R(x)$. Since bR occurs in $\Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$, we have $\mathcal{T}, \mathcal{A} \models A_R(b)$. By H-completeness of $\mathcal{A}$, $A_R(b) \in \mathcal{A}$, and so by applying the clause $P_{\boldsymbol{q}_0} \leftarrow A_R(x)$, we obtain $\Pi''_{\boldsymbol{Q}}, \mathcal{A} \models P_{\boldsymbol{q}}(\boldsymbol{a})$. ❑

B Experiments

B.1 Computing rewritings

We computed four types of rewritings for linear queries similar to those in Example 7 and a fixed ontology from Example 9. We denote the rewriting from Section 4 by LOG (because it is of logarithmic depth), and from Section 5 by LIN (because it is of linear depth). Other two rewritings were obtained by running executables of Rapid [6] and Clipper [8] with a 5 minute timeout on a desktop machine. We considered the following three sequences of letters R and S:

$$RRSRSRSRRSRRSSR, \qquad \text{(Sequence 1)}$$
$$SRRRRRSRSRRRRRR, \qquad \text{(Sequence 2)}$$
$$SRRSSRSRSRRSRRSS. \qquad \text{(Sequence 3)}$$

For each of the three sequences, we consider the line-shaped queries with 1–15 atoms formed by their prefixes. Table 1 present the sizes of different types of rewritings.

B.2 Datasets

We used Erdös-Rènyi random graphs with independent parameters V (number of vertices), p (probability of an R-edge) and q (probability of concepts A and B at a given

Table 1. The size (number of clauses) of different types of rewritings for the three sequences of queries (– indicates timeout after 5 minutes)

no. of atoms	Sequence 1 $RRSRSRSRRSRRSSR$				Sequence 2 $SRRRRRSRSRRRRRR$				Sequence 3 $SRRSSRSRSRRSRRSS$			
	Rapid	Clipper	LIN	LOG	Rapid	Clipper	LIN	LOG	Rapid	Clipper	LIN	LOG
1	1	1	2	1	1	1	2	1	1	1	2	1
2	1	1	5	2	2	2	5	4	2	2	5	4
3	2	2	8	5	2	2	8	5	2	2	8	5
4	3	3	11	8	2	2	11	6	4	4	11	8
5	5	5	14	12	2	2	14	8	4	4	14	10
6	7	7	17	16	2	2	17	10	8	8	17	15
7	10	11	20	20	4	4	20	13	11	11	20	18
8	13	16	23	24	6	7	23	16	18	24	23	21
9	13	16	26	27	10	13	26	22	24	35	26	27
10	26	44	29	32	14	26	29	27	34	63	29	33
11	39	72	32	36	14	26	32	29	43	100	32	37
12	39	126	35	40	14	26	35	33	56	302	35	42
13	–	241	38	45	–	30	38	35	–	–	38	46
14	–	–	41	47	–	31	41	36	–	–	41	51
15	–	–	44	51	–	30	44	37	–	–	44	52

Table 2. Generated datasets

dataset	V	p	q	avg. degree of vertices	no. of atoms
4.ttl	1 000	0.050	0.050	50	61 498
5.ttl	5 000	0.002	0.004	10	64 157
6.ttl	10 000	0.002	0.004	20	256 804
8.ttl	20 000	0.002	0.010	40	1 027 028

vertex). Note that we intentionally did not introduce any S-edges. The last parameter, the average degree of a vertex, is $V \cdot p$. Table 2 summarises the parameters of the datasets.

B.3 Evaluating rewritings

We evaluated all obtained rewritings for the sequence $RRSRSRSRRSRRSSR$ on the datasets in Section B.2 using RDFox triplestore [17]. The materialisation time and other relevant statistics are given in Table 3.

Table 3. Evaluating rewritings on RDFox

data-set	query size	evaluation time (sec)				no. of answers	no. of generated tuples			
		Rapid	Clipper	LIN	LOG		Rapid	Clipper	LIN	LOG
4.ttl	7	0.271	0.242	0.008	0.243	2 956	2 956	2 956	3 246	125 361
	8	0.412	0.377	0.084	0.904	212 213	212 213	212 213	302 221	1 659 409
	9	3.117	3.337	3.376	2.941	998 945	998 945	998 945	2 927 979	2 684 359
	10	1.079	1.102	0.012	0.607	8 374	8 374	10 760	12 573	1 178 714
	11	2.246	1.984	0.385	0.945	436 000	436 000	436 000	836 876	1 618 743
	12	13.693	30.032	8.129	6.867	999 998	999 998	1 000 000	5 311 314	4 439 352
	13	–	6.810	0.027	0.616	20 985	–	24 839	38 200	553 821
	14	–	–	0.013	0.358	0	–	–	48	312 723
	15	–	–	0.032	0.394	2 000	–	–	70 277	376 313
5.ttl	7	0.089	0.080	0.008	0.078	427	427	427	613	68 546
	8	0.136	0.125	0.029	0.434	8 778	8 778	8 778	76 202	1 085 362
	9	0.202	0.254	0.369	0.554	105 853	105 853	105 853	1 020 363	1 190 249
	10	0.174	0.204	0.011	0.461	11	11	438	506	943 097
	11	0.192	0.259	0.036	0.473	651	651	9 396	74 922	944 210
	12	0.244	0.699	0.396	1.034	8 058	8 058	113 179	1 004 735	1 940 300
	13	–	0.629	0.015	0.244	0	–	438	502	209 915
	14	–	–	0.014	0.153	0	–	–	31	200 962
	15	–	–	0.032	0.172	0	–	–	64 543	265 087
6.ttl	7	0.631	0.581	0.035	0.756	1 217	1 217	1 217	1 499	296 711
	8	0.925	0.876	0.159	4.377	67 022	67 022	67 022	335 578	7 546 184
	9	1.949	2.275	4.063	5.251	1 678 668	1 678 668	1 678 668	8 613 829	9 225 201
	10	1.24	1.377	0.049	4.731	60	60	1 277	1 389	6 936 178
	11	1.403	1.798	0.249	4.846	11 498	11 498	77 811	341 459	6 949 160
	12	1.697	5.413	4.355	10.128	305 640	305 640	1 951 654	8 780 232	15 626 926
	13	–	4.382	0.082	1.762	0	–	1 277	1 377	917 117
	14	–	–	0.063	1.115	0	–	–	47	850 309
	15	–	–	0.177	1.011	0	–	–	257 974	1 107 065
8.ttl	7	6.614	6.277	0.243	8.586	13 103	13 103	13 103	14 625	1 665 376
	8	11.441	10.923	1.880	54.813	1 286 991	1 286 991	1 286 991	2 432 629	56 098 445
	9	46.704	50.668	76.169	102.055	58 753 514	58 753 514	58 753 514	114 973 160	114 837 395
	10	14.348	15.503	0.375	43.347	19 966	19 966	33 014	35 359	52 103 362
	11	19.593	20.907	2.843	44.410	1 872 159	1 872 159	3 051 184	4 397 556	53 986 724
	12	71.354	182.499	172.822	237.478	79 939 048	79 939 048	120 229 590	199 083 489	242 500 074
	13	–	54.497	0.562	22.345	22 474	–	53 717	58 826	5 686 759
	14	–	–	0.550	12.462	0	–	–	253	4 356 739
	15	–	–	1.211	11.315	12 165	–	–	1 064 542	5 395 902

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley (1995)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press (2003)
3. Bienvenu, M., Kikot, S., Podolskii, V.V.: Tree-like queries in OWL 2 QL: succinctness and complexity results. In: Proc. of the 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2015). pp. 317–328. ACM (2015)
4. Calì, A., Gottlob, G., Lukasiewicz, T.: A general datalog-based framework for tractable query answering over ontologies. Journal of Web Semantics 14, 57–83 (2012)
5. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: the *DL-Lite* family. Journal of Automated Reasoning 39(3), 385–429 (2007)
6. Chortaras, A., Trivela, D., Stamou, G.: Optimized query rewriting for OWL 2 QL. In: Proc. of CADE-23. LNCS, vol. 6803, pp. 192–206. Springer (2011)
7. Cook, S.A.: Characterizations of pushdown machines in terms of time-bounded computers. Journal of the ACM 18(1), 4–18 (1971)
8. Eiter, T., Ortiz, M., Šimkus, M., Tran, T.K., Xiao, G.: Query rewriting for Horn-SHIQ plus rules. In: Proc. of the 26th AAAI Conf. on Artificial Intelligence (AAAI 2012). pp. 726–733. AAAI (2012)
9. Gottlob, G., Kikot, S., Kontchakov, R., Podolskii, V.V., Schwentick, T., Zakharyaschev, M.: The price of query rewriting in ontology-based data access. Artificial Intelligence 213, 42–59 (2014)
10. Gottlob, G., Leone, N., Scarcello, F.: Computing LOGCFL certificates. In: Proc. of the 26th Int. Colloquium on Automata, Languages and Programming (ICALP-99). LNCS, vol. 1644, pp. 361–371. Springer (1999)
11. Huffman, D.A.: A method for the construction of minimum-redundancy codes. Proc. of the Institute of Radio Engineers 40(9), 1098–1101 (1952)
12. Kikot, S., Kontchakov, R., Podolskii, V., Zakharyaschev, M.: On the succinctness of query rewriting over shallow ontologies. In: Proc. of the 29th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2014). ACM (2014)
13. Kikot, S., Kontchakov, R., Podolskii, V.V., Zakharyaschev, M.: Exponential lower bounds and separation for query rewriting. In: Proc. of the 39th Int. Colloquium on Automata, Languages and Programming (ICALP 2012). LNCS, vol. 7392, pp. 263–274. Springer (2012)
14. Kikot, S., Kontchakov, R., Zakharyaschev, M.: On (in)tractability of OBDA with OWL 2 QL. In: Proc. of the 24th Int. Workshop on Description Logics (DL 2011). vol. 745, pp. 224–234. CEUR-WS (2011)
15. Kikot, S., Kontchakov, R., Zakharyaschev, M.: Conjunctive query answering with OWL 2 QL. In: Proc. of the 13th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2012). pp. 275–285. AAAI (2012)
16. Kontchakov, R., Lutz, C., Toman, D., Wolter, F., Zakharyaschev, M.: The combined approach to query answering in DL-Lite. In: Proc. of the 12th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2010). AAAI Press (2010)
17. Nenov, Y., Piro, R., Motik, B., Horrocks, I., Wu, Z., Banerjee, J.: RDFox: A highly-scalable RDF store. In: Proc. of the 14th Int. Semantic Web Conf. (ISWC 2015), Part II. LNCS, vol. 9367, pp. 3–20. Springer (2015)
18. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking data to ontologies. Journal on Data Semantics X, 133–173 (2008)

19. Sudborough, I.H.: On the tape complexity of deterministic context-free languages. Journal of the ACM 25(3), 405–414 (1978)
20. Venkateswaran, H.: Properties that characterize LOGCFL. Journal of Computer and System Sciences 43(2), 380–404 (1991)