Towards a Data Complexity Classification of Ontology-Mediated Queries with Covering

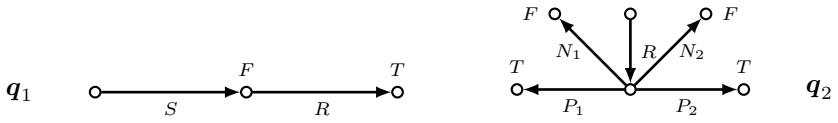
O. Gerasimova[1], S. Kikot[2], and M. Zakharyaschev[3]

[1] National Research University Higher School of Economics, Moscow, Russia
[2] University of Oxford, U.K.
[3] Birkbeck, University of London, U.K.

Abstract. We prove a number of new syntactic and semantic sufficient and necessary conditions for ontology-mediated queries (OMQs) with one covering axiom to be in the classes AC^0 and NL for data complexity, and to be L-, NL- or P-hard. We also give two new examples of very simple coNP-complete OMQs.

1 Introduction

The problem we are concerned with in this paper originated from the observation that the NPD FactPages ontology[4], used for testing ontology-based data access (OBDA) in industry [13, 12], contained covering axioms of the form $A \sqsubseteq B_1 \sqcup \cdots \sqcup B_n$. It has been known since Schaerf's paper [18] that answering ontology-mediated queries (OMQs) with a covering axiom can be coNP-hard for data complexity, and so such OMQs are not suitable for OBDA systems in general. In fact, two interesting OMQs with covering can be extracted from [18]: $\boldsymbol{Q}_1 = (\mathcal{C}ov_\top, \boldsymbol{q}_1)$ and $\boldsymbol{Q}_2 = (\mathcal{C}ov_\top, \boldsymbol{q}_2)$, where $\mathcal{C}ov_\top = \{\top \sqsubseteq F \sqcup T\}$ and the Boolean conjunctive queries (CQs) $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ are shown below.



Schaerf pointed out that answering these queries involves case analysis and showed that $\boldsymbol{Q}_2$ is coNP-hard for data complexity. It is readily seen that $\boldsymbol{Q}_1$ is NL-complete. The problem we started attacking in [9] was to find a complete syntactic or semantic classification of OMQs with covering according to their data complexity. The direct attack failed, and connections with other hard classification problems such as boundedness or linearisability of datalog programs (to be discussed in Section 3) indicate that a longer siege is needed. In this paper, we report on our recent findings.

We consider four ontologies with covering: $\mathcal{C}ov_\top$ as above, $\mathcal{C}ov_A = \{A \sqsubseteq F \sqcup T\}$, $\mathcal{C}ov_\top^\bot = \{\top \sqsubseteq F \sqcup T,\ \ F \sqcap T \sqsubseteq \bot\}$, $\mathcal{C}ov_A^\bot = \{A \sqsubseteq F \sqcup T,\ \ F \sqcap T \sqsubseteq \bot\}$, and we classify connected Boolean CQs $\boldsymbol{q}$ by the number of solitary occurrences of F and T in them, where a *solitary* F is any $F(x) \in \boldsymbol{q}$ with $T(x) \notin \boldsymbol{q}$, and symmetrically for T. We prove a number of new syntactic and semantic sufficient and necessary conditions for

[4] http://sws.ifi.uio.no/project/npd-v2/

such OMQs with at most one solitary F to be in the complexity classes AC^0 and NL, and to be L-, NL- or P-hard. We also give two new examples of very simple coNP-complete OMQs.

2 Preliminaries

In this paper, a *Boolean conjunctive query* (*CQ*) is any first-order (FO) sentence of the form $\boldsymbol{q} = \exists \boldsymbol{x}\, \varphi(\boldsymbol{x})$, where φ is a conjunction of unary or binary atoms $P(\boldsymbol{y})$ with $\boldsymbol{y} \subseteq \boldsymbol{x}$. We often regard CQs as *sets* of their atoms, depict them as labelled digraphs, and assume that all of our CQs are *connected*. By *answering* an *ontology-mediated query* (OMQ) $\boldsymbol{Q} = (\mathcal{T}, \boldsymbol{q})$ with a TBox $\mathcal{T}$ of the form defined above, we understand the problem of checking, given an ABox $\mathcal{A}$ (a finite set of unary or binary ground atoms), whether $\boldsymbol{q}$ holds in every model of $\mathcal{T} \cup \mathcal{A}$, or $\mathcal{T}, \mathcal{A} \models \boldsymbol{q}$ in symbols. For every $\boldsymbol{Q}$, this problem is clearly in coNP. It is in the complexity class AC^0 if there is an FO-sentence $\boldsymbol{q}'$, called an *FO-rewriting* of $\boldsymbol{Q}$, such that $\mathcal{T}, \mathcal{A} \models \boldsymbol{q}$ iff $\mathcal{A} \models \boldsymbol{q}'$, for any ABox $\mathcal{A}$.

A *datalog program*, Π, is a finite set of *rules* $\forall \boldsymbol{x}\, (\gamma_0 \leftarrow \gamma_1 \wedge \dots \wedge \gamma_m)$, where each γ_i is an atom $Q(\boldsymbol{y})$ with $\boldsymbol{y} \subseteq \boldsymbol{x}$. (As usual, we omit $\forall \boldsymbol{x}$.) The atom γ_0 is the *head* of the rule, and $\gamma_1, \dots, \gamma_m$ its *body*. All the variables in the head must occur in the body. The predicates in the head of rules are *IDB predicates*, the rest *EDB predicates* [1].

A *datalog query* is a pair (Π, G), where Π is a datalog program and G an 0-ary atom, the *goal*. The *answer* to (Π, G) over an ABox $\mathcal{A}$ is 'yes' if G holds in the FO-structure obtained by closing $\mathcal{A}$ under Π, in which case we write $\Pi, \mathcal{A} \models G$. A datalog query (Π, G) is a *datalog rewriting* of an OMQ $\boldsymbol{Q} = (\mathcal{T}, \boldsymbol{q})$ in case $\mathcal{T}, \mathcal{A} \models \boldsymbol{q}$ iff $\Pi, \mathcal{A} \models G$, for any ABox $\mathcal{A}$. The *answering problem* for (Π, G)—i.e., checking, given an ABox $\mathcal{A}$, whether $\Pi, \mathcal{A} \models G$—is clearly in P. Answering a datalog query with a *linear* program, whose rules have at most one IDB predicate in the body, can be done in NL. The NL upper bound also holds for datalog queries with a linear-stratified program defined as follows. A *stratified* program [1] is a sequence $\Pi = (\Pi_0, \dots, \Pi_n)$ of datalog programs, called the *strata* of Π, such that each predicate in Π can occur in the head of a rule only in one stratum Π_i and can occur in the body of a rule only in strata Π_j with $j \geq i$. If, additionally, the body of each rule in Π contains at most one occurrence of a head predicate from the same stratum, Π is called *linear-stratified*. Every linear-stratified program can be converted to an equivalent linear datalog program [2].

3 Connections in High Places

We begin by putting our little problem into the context of more general investigations of (i) boundedness and linearisability of datalog programs and (ii) the data complexity of answering OMQs with expressive ontologies.

The decision problem whether a given datalog program is bounded has been a hot research topic in database theory since the late 1980s. Thus, it was shown that boundedness is undecidable already for linear datalog programs with binary IDB predicates [20] and single rule programs (aka *sirups*) [15]. On the other hand, deciding boundedness is 2ExpTime-complete for *monadic* datalog programs [7, 4] and PSpace-complete for linear monadic programs [7]; for linear sirups, it is even NP-complete [20].

The last two results are relevant for deciding FO-rewritability of OMQs $(\mathcal{C}ov_A, \boldsymbol{q})$, where $\boldsymbol{q}$ has a single solitary F and is called a 1-*CQ*. Indeed, suppose that $F(x)$ and $T(y_1), \dots, T(y_n)$ are all the solitary occurrences of F and T in $\boldsymbol{q}$. Let $\Pi_{\boldsymbol{q}}$ be a monadic datalog program with three rules

$$G \leftarrow F(x), \boldsymbol{q}', P(y_1), \dots, P(y_n), \tag{1}$$

$$P(x) \leftarrow T(x), \tag{2}$$

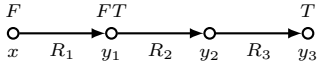
$$P(x) \leftarrow A(x), \boldsymbol{q}', P(y_1), \dots, P(y_n), \tag{3}$$

where $\boldsymbol{q}' = \boldsymbol{q} \setminus \{F(x), T(y_1), \dots, T(y_n)\}$ and P is a fresh predicate symbol that never occurs in our ABoxes. Then, for any ABox $\mathcal{A}$, we have $\mathcal{C}ov_A, \mathcal{A} \models \boldsymbol{q}$ iff $\Pi_{\boldsymbol{q}}, \mathcal{A} \models G$. Thus, FO-rewritability of $(\mathcal{C}ov_A, \boldsymbol{q})$ is clearly related to boundedness of the sirup (3).

The problem of linearising datalog programs, that is, transforming them into equivalent linear datalog programs, which are known to be in NL for data complexity, has also attracted much attention [16, 17, 21, 2] after the Ullman and van Gelder pioneering paper [19].

The success of the OBDA paradigm spurred considerable interest in the data complexity of answering individual OMQs with expressive ontologies. Thus, by establishing a remarkable connection to CSPs, it was shown in [5] that deciding FO-rewritability and datalog rewritability of OMQs with a $\mathcal{SHIU}$ ontology and a (Boolean) atomic query is NEXPTIME-complete. This result is obviously applicable to $(\mathcal{C}ov_A, \boldsymbol{q})$ with a *tree-shaped* $\boldsymbol{q}$. In [8], it was shown that checking rewritability of Boolean monadic *disjunctive* datalog programs into FO and monadic datalog can be done in, respectively, 2NEXPTIME and 3EXPTIME, which is applicable to all of our OMQs $(\mathcal{C}ov_A, \boldsymbol{q})$.

An AC^0/NL/P trichotomy for the data complexity of answering OMQs with an $\mathcal{EL}$ ontology and atomic query, which can be checked in EXPTIME, was established in [14]. This result is applicable to OMQs $(\mathcal{C}ov_A, \boldsymbol{q})$, in which $\boldsymbol{q}$ is an *F-tree* having a single solitary $F(x)$ such that the binary atoms in $\boldsymbol{q}$ form a ditree with root x. Indeed, denote by $\mathcal{T}_{\boldsymbol{Q}}$ the $\mathcal{EL}$ TBox with concept inclusions $F \sqcap C_{\boldsymbol{q}} \sqsubseteq G'$, $T \sqsubseteq P$ and $A \sqcap C_{\boldsymbol{q}} \sqsubseteq P$, where $C_{\boldsymbol{q}}$ is an $\mathcal{EL}$-concept representing $\boldsymbol{q} \setminus \{F(x)\}$ with P for T (so for $\boldsymbol{q}$ of the form



$C_{\boldsymbol{q}} = \exists R_1.(F \sqcap P \sqcap \exists R_2.\exists R_3.P)$). Then, for any ABox $\mathcal{A}$ that does not contain G', we have $\Pi_{\boldsymbol{Q}}, \mathcal{A} \models G$ iff $\mathcal{T}_{\boldsymbol{Q}}, \mathcal{A} \models \exists x\, G'(x)$.

Yet, despite all of these efforts and results (implying, in view of the recent positive solution to the Feder-Vardi conjecture [6, 22], that there is a P/CONP dichotomy for OMQs with a $\mathcal{SHIU}$ ontology and a (Boolean) atomic query, which is decidable in NEXPTIME), we are still lacking simple and transparent, in particular syntactic, conditions guaranteeing this or that data complexity or type of rewritability. Some results in this direction were obtained in [10, 11]. That a transparent classification of monadic sirups according to their data complexity has not been found so far and the close connection to CSPs indicate that this problem is extremely hard in general. Our aim below is to demonstrate that, for the OMQs with covering considered in this paper, syntactic conditions do exist but are difficult to find.

4 0-CQs

As observed in [9], if an OMQ $\boldsymbol{Q}$ (of the form defined above) is in AC^0, then $\boldsymbol{q}$ is an FO-rewriting of $\boldsymbol{Q}$. By a 0-*CQ* we mean any CQ that does not contain a solitary F. A *twin* in a CQ $\boldsymbol{q}$ is any pair $F(x), T(x) \in \boldsymbol{q}$. Here is an encouraging syntactic criterion of FO-rewritability for OMQs of the form $(\mathcal{C}ov_A^{\bot}, \boldsymbol{q})$:

Theorem 1. (i) *If $\boldsymbol{q}$ is a 0-CQ, then answering both $(\mathcal{C}ov_A^{\bot}, \boldsymbol{q})$ and $(\mathcal{C}ov_A, \boldsymbol{q})$ is in* AC^0*, with $\boldsymbol{q}$ being an FO-rewriting of these OMQs.*

(ii) *If $\boldsymbol{q}$ is not a 0-CQ and does not contain twins, then answering both $(\mathcal{C}ov_{\top}^{\bot}, \boldsymbol{q})$ and $(\mathcal{C}ov_{\top}, \boldsymbol{q})$ is* L*-hard.*

Proof. (i) follows from [9, Theorem 1]. (ii) The proof is by reduction to the reachability problem for undirected graphs, which is known to be L-complete; see, e.g., [3]. Denote by $\boldsymbol{q}'$ the CQ obtained from $\boldsymbol{q}$ by gluing together all the variables x with $F(x) \in \boldsymbol{q}$ and also all the variables y with $T(y) \in \boldsymbol{q}$. Thus, $\boldsymbol{q}'$ contains a single F-atom, $F(x)$, and a single T-atom, $T(y)$. Clearly, there is a homomorphism $h\colon \boldsymbol{q} \to \boldsymbol{q}'$. Let $\boldsymbol{q}'' = \boldsymbol{q}' \setminus \{F(x), T(y)\}$.

Suppose we are given an undirected graph $G = (V, E)$ and two vertices $s, t \in V$. We regard G as a directed graph such that $(u, v) \in E$ iff $(v, u) \in E$, for any $u, v \in V$. Now, we encode G by means of an ABox $\mathcal{A}_G$ that is obtained from G as follows. For every edge $e = (u, v) \in E$, let $\boldsymbol{q}''_e$ be the set of atoms in $\boldsymbol{q}''$ with x renamed to u, y to v and all other variables z to z_e. Then $\mathcal{A}_G$ comprises all such $\boldsymbol{q}''_e$, for $e \in E$, as well as $F(s)$ and $T(t)$. We show that $s \to_G t$ iff $\mathcal{C}ov_{\top}, \mathcal{A}_G \models \boldsymbol{q}$.

Suppose that $s \to_G t$, i.e., there exists a path $s = v_0, \dots, v_n = t$ in G with $e_i = (v_i, v_{i+1}) \in E$, for $i < n$. Consider an arbitrary model $\mathcal{I}$ of $\mathcal{C}ov_{\top}$ and $\mathcal{A}_G$. Since $\mathcal{I} \models \mathcal{C}ov_{\top}$, and $F(s)$ and $T(t)$ are in $\mathcal{A}_G$, we can find some $i < n$ such that $\mathcal{I} \models F(v_i)$ and $\mathcal{I} \models T(v_{i+1})$. As $\boldsymbol{q}''_{e_i}$ is an isomorphic copy of $\boldsymbol{q}''$, we obtain $\mathcal{I} \models \boldsymbol{q}'$, and so $\mathcal{I} \models \boldsymbol{q}$. Conversely, suppose $s \not\to_G t$. Define an interpretation $\mathcal{I}$, extending the ABox $\mathcal{A}_G$, by setting $F^{\mathcal{I}}$ to be the set of objects in $\mathcal{A}_G$ that are reachable from s and $T^{\mathcal{I}}$ its complement. Clearly, $\mathcal{I}$ is a model of $\mathcal{C}ov_{\top}$. By the construction, the elements of the connected component of $\mathcal{I}$ containing s cannot be instances of T, while the remaining elements of $\mathcal{I}$ are not be instances of F. Since $\boldsymbol{q}$ is connected, it follows that $\mathcal{I} \not\models \boldsymbol{q}$.

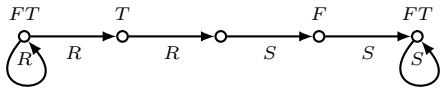
Corollary 1. *An OMQ $(\mathcal{C}ov_A^{\bot}, \boldsymbol{q})$ is in* AC^0 *iff $\boldsymbol{q}$ is a 0-CQ, which can be decided in linear time.*

If twins can occur in CQs (that is, F and T are not necessarily disjoint), the picture becomes more complex. On one hand, we have the following criterion for OMQs $(\mathcal{C}ov_A, \boldsymbol{q})$ with a *path CQ* $\boldsymbol{q}$ whose variables $x_0, \dots, x_n$ in $\boldsymbol{q}$ are ordered so that the binary atoms in $\boldsymbol{q}$ form a chain $R_1(x_0, x_1), \dots, R_n(x_{n-1}, x_n)$.

Theorem 2 ([9]). *An OMQ $(\mathcal{C}ov_A, \boldsymbol{q})$ with a path CQ $\boldsymbol{q}$ is in* AC^0 *iff $\boldsymbol{q}$ is a 0-CQ. If $\boldsymbol{q}$ contains both solitary F and T, then $(\mathcal{C}ov_A, \boldsymbol{q})$ is* NL*-hard.*

On the other hand, this AC^0/NL criterion collapses for path CQs with loops:

Proposition 1. *The OMQ $(\mathcal{C}ov_A, \boldsymbol{q})$, where $\boldsymbol{q}$ is shown below, is in* AC^0.



Proof. Follows from the sufficient condition of Theorem 4 (i).

Note that the CQ $\boldsymbol{q}$ above is *minimal* (not equivalent to any of its proper sub-CQs). Note also that, if a minimal 1-CQ $\boldsymbol{q}$ contains both a solitary F and a solitary T, then FO-rewritability of $(\mathcal{C}ov_A, \boldsymbol{q})$ implies that $\boldsymbol{q}$ contains at least one twin (FT) and at least one y with $T(y) \notin \boldsymbol{q}$ and $F(y) \notin \boldsymbol{q}$ (which can be shown using Theorem 4 (i)).

5 1-CQs

1-CQs have exactly one solitary F. As observed in Section 3, we have the following:

Theorem 3. (i) *Answering any OMQ* $(\mathcal{C}ov_A, \boldsymbol{q})$ *with a* 1*-CQ* $\boldsymbol{q}$ *can be done in* P.

(ii) *Answering any OMQ* $(\mathcal{C}ov_A, \boldsymbol{q})$ *with an* F*-tree* $\boldsymbol{q}$ *is either in* AC^0 *or* NL*-complete or* P*-complete. The trichotomy can be decided in* ExpTime.

Theorem 3 (ii) was proved by a reduction to the AC^0/NL/P-trichotomy of [14]. It is to be noted, however, that applying the algorithm from [14] in our case is tricky because the input ontology $\mathcal{T}_{\boldsymbol{Q}}$ must first be converted to a normal form. As a result, we do not obtain transparent syntactic criteria on the shape of $\boldsymbol{q}$ that would guarantee that the OMQ $(\mathcal{C}ov_A, \boldsymbol{q})$ belongs to the desired complexity class (see examples below).

We now give a semantic sufficient condition for an OMQ with a 1-CQ to lie in NL. This condition uses ideas and constructions from [7, 14]. Let $\boldsymbol{Q} = (\mathcal{C}ov_A, \boldsymbol{q})$ be an OMQ with a 1-CQ $\boldsymbol{q}$ having a solitary $F(x)$. Define by induction a class $\mathfrak{K}_{\boldsymbol{Q}}$ of ABoxes that will be called *cactuses for* $\boldsymbol{Q}$. We start by setting $\mathfrak{K}_{\boldsymbol{Q}} := \{\boldsymbol{q}\}$, regarding $\boldsymbol{q}$ as an ABox, and then recursively apply to $\mathfrak{K}_{\boldsymbol{Q}}$ the following two rules:

(bud) if $T(y) \in \mathcal{A} \in \mathfrak{K}_{\boldsymbol{Q}}$ with solitary $T(y)$, then we add to $\mathfrak{K}_{\boldsymbol{Q}}$ the ABox obtained by replacing $T(y)$ in $\mathcal{A}$ with $(\boldsymbol{q} \setminus F(x)) \cup \{A(x)\}$, in which x is renamed to y and all of the other variables are given *fresh* names;

(prune) if $\mathcal{C}ov_A, \mathcal{A}' \models \boldsymbol{Q}$, where $\mathcal{A}' = \mathcal{A} \setminus \{T(y)\}$ and $T(y)$ is solitary, we add to $\mathfrak{K}_{\boldsymbol{Q}}$ the ABox obtained by removing $T(y)$ from $\mathcal{A} \in \mathfrak{K}_{\boldsymbol{Q}}$.

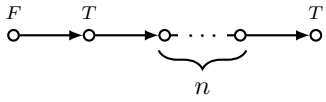
It is readily seen that, for any ABox $\mathcal{A}'$, we have $\mathcal{C}ov_A, \mathcal{A}' \models \boldsymbol{Q}$ iff there exist $\mathcal{A} \in \mathfrak{K}_{\boldsymbol{Q}}$ and a homomorphism $h\colon \mathcal{A} \to \mathcal{A}'$. Denote by $\mathfrak{K}^{\dagger}_{\boldsymbol{Q}}$ the set of minimal cactuses in $\mathfrak{K}_{\boldsymbol{Q}}$ (that have no proper sub-cactuses in $\mathfrak{K}_{\boldsymbol{Q}}$).

For a cactus $\mathcal{C} \in \mathfrak{K}_{\boldsymbol{Q}}$, we refer to the copies of (maximal subsets of) $\boldsymbol{q}$ that comprise $\mathcal{C}$ as *segments*. The *skeleton* $\mathcal{C}^s$ of $\mathcal{C}$ is the ditree whose nodes are the segments $\mathfrak{s}$ of $\mathcal{C}$ and edges $(\mathfrak{s}, \mathfrak{s}')$ mean that $\mathfrak{s}'$ was attached to $\mathfrak{s}$ by budding. The atoms $T(y) \in \mathfrak{s}$ are called the *buds* of $\mathfrak{s}$. The *rank* $\boldsymbol{r}(\mathfrak{s})$ of $\mathfrak{s}$ is defined by induction: if $\mathfrak{s}$ is a leaf, then $\boldsymbol{r}(\mathfrak{s}) = 0$; for non-leaf $\mathfrak{s}$, we compute the maximal rank m of its children and then set

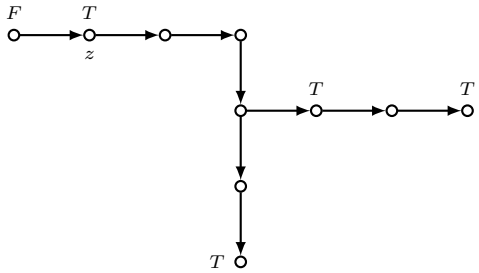
$$\boldsymbol{r}(\mathfrak{s}) = \begin{cases} m+1, & \text{if } \mathfrak{s} \text{ has} \geq 2 \text{ children of rank } m; \\ m, & \text{otherwise.} \end{cases}$$

The *width* of $\mathcal{C}$ and $\mathcal{C}^s$ is the rank of the root in $\mathcal{C}^s$. We say that $\mathfrak{K}^\dagger_{\boldsymbol{Q}}$ is of *width* k if it contains a cactus of width k but no cactus of greater width. The *depth* of $\mathcal{C}$ and $\mathcal{C}^s$ is the number of edges in the longest branch in $\mathcal{C}^s$.

We illustrate the definition by an example. Denote by $\boldsymbol{q}_{TnT}$, for $n \geq 0$, the 1-CQ shown below, where all the binary predicates are R and the n variables without labels do not occur in F- or T-atoms:



Example 1. Let $\boldsymbol{Q} = (\mathcal{C}ov_\top, \boldsymbol{q}_{T1T})$. In the picture below, we show a cactus $\mathcal{C}$ obtained by applying **(bud)** twice to $\boldsymbol{q}_{T1T}$ (with $A = \top$ omitted):



One can check that $\mathcal{C}ov_\top, \mathcal{C} \setminus \{T(z)\} \models \boldsymbol{q}_{T1T}$, and so an application of **(prune)** will remove $T(z)$ from $\mathcal{C}$. Using this observation, one can show that $\mathfrak{K}^\dagger_{\boldsymbol{Q}}$ is of width 1. On the other hand, if $\boldsymbol{Q} = (\mathcal{C}ov_A, \boldsymbol{q}_{T1T})$ then $\mathfrak{K}^\dagger_{\boldsymbol{Q}}$ is of unbounded width as follows from Theorem 6 below.

Theorem 4. *Let $\boldsymbol{Q} = (\mathcal{C}ov_A, \boldsymbol{q})$ be an OMQ with a 1-CQ $\boldsymbol{q}$. Then*

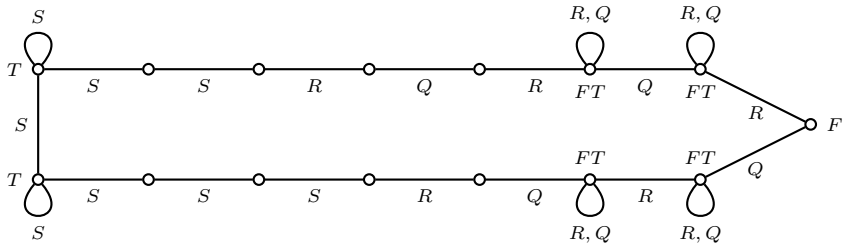
(i) $\boldsymbol{Q}$ is in AC^0 *iff for every $\mathcal{C} \in \mathfrak{K}^\dagger_{\boldsymbol{Q}}$, there is a homomorphism $h\colon \boldsymbol{q} \to \mathcal{C}$;*

(ii) $\boldsymbol{Q}$ is rewritable in linear datalog, and so is in NL*, if $\mathfrak{K}^\dagger_{\boldsymbol{Q}}$ is of bounded width.*

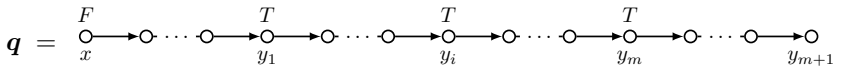
Proof. The proof of (i) is straightforward. To prove (ii), we represent the ABoxes in $\mathfrak{K}_{\boldsymbol{Q}}$ as words in a tree alphabet and construct a non-deterministic finite tree automaton $\mathfrak{A}_{\boldsymbol{Q}}$ such that $\mathfrak{K}^\dagger_{\boldsymbol{Q}} \subseteq L(\mathfrak{A}_{\boldsymbol{Q}}) \subseteq \mathfrak{K}_{\boldsymbol{Q}}$. Then we show that, for $\mathfrak{K}^\dagger_{\boldsymbol{Q}}$ of finite width, the automaton $\mathfrak{A}_{\boldsymbol{Q}}$ can be transformed into a monadic linear-stratified datalog rewriting of $\boldsymbol{Q}$. It can further be converted into a linear datalog rewriting at the expense of increasing the arity if IDBs in the program [2].

It is worth noting that, for $\boldsymbol{Q} = (\mathcal{C}ov_\top, \boldsymbol{q})$ with $\boldsymbol{q}$ from Proposition 1, $\mathfrak{K}^\dagger_{\boldsymbol{Q}}$ consists of $\boldsymbol{q}$ and the cactus of depth 1, in which the only solitary T is removed by **(prune)**. Clearly, there is a homomorphism from $\boldsymbol{q}$ into this cactus, and so $\boldsymbol{Q}$ is FO-rewritable. However, for the 1-CQ $\boldsymbol{q}$ in the picture below (where all edges are bidirectional), $(\mathcal{C}ov_\top, \boldsymbol{q})$ is not FO-rewritable, but there is a homomorphism from $\boldsymbol{q}$ to both cactuses of depth 1. We do not know whether, in general, there is an upper bound $N_{\boldsymbol{q}}$ such that the existence of homomorphisms $h\colon \boldsymbol{q} \to \mathcal{C}$, for all $\mathcal{C} \in \mathfrak{K}_{\boldsymbol{Q}}$ of depth $N_{\boldsymbol{q}}$, would ensure FO-rewritability of $(\mathcal{C}ov_A, \boldsymbol{q})$. For 1-CQs $\boldsymbol{q}$ with a single solitary T, one can take $N_{\boldsymbol{q}} = |\boldsymbol{q}| + 1$. Neither do we know the exact complexity of deciding FO-rewritability of OMQs with 1-CQs.

As mentioned in Section 3, this problem is reducible to the boundedness problem for monadic datalog programs, which is known to be in 2EXPTIME.



Theorem 4 (ii) allows us to obtain a sufficient condition for linear-datalog rewritability of OMQs $(\mathcal{C}ov_A, \boldsymbol{q})$ with an F-path CQ $\boldsymbol{q}$, that is, a path CQ with a single solitary F at its root. We represent such a $\boldsymbol{q}$ as shown in the picture below, which indicates *all* the solitary occurrences of F and T:



We require the following sub-CQs of $\boldsymbol{q}$:

- $\boldsymbol{q}_i$ is the suffix of $\boldsymbol{q}$ that starts at y_i, but without $T(y_i)$, for $1 \leq i \leq m$;
- $\boldsymbol{q}_i^*$ is the prefix of $\boldsymbol{q}$ that ends at y_i, but without $F(x)$ and $T(y_i)$, for $1 \leq i \leq m$;
- $\boldsymbol{q}_{m+1}^*$ is $\boldsymbol{q}$ without $F(x)$,

and write $f_i \colon \boldsymbol{q}_i \twoheadrightarrow \boldsymbol{q}$ if f_i is a homomorphism from $\boldsymbol{q}_i$ into $\boldsymbol{q}$ with $f_i(y_i) = x$.

Theorem 5. *If for each $1 \leq i \leq m$ there exist $f_i \colon \boldsymbol{q}_i \twoheadrightarrow \boldsymbol{q}$, then $(\mathcal{C}ov_A, \boldsymbol{q})$ is rewritable into a linear datalog program, and so is* NL*-complete.*

For F-path CQs $\boldsymbol{q}$ without twins, we extend Theorem 5 to a NL/P dichotomy (provided that NL $\neq$ P). Given such a CQ $\boldsymbol{q}$, we denote by $N_{\boldsymbol{q}}$ the set of the numbers indicating the length of the path from x to each of the y_i, $i = 1, \dots, m+1$.

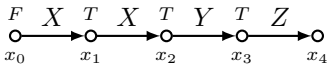
Theorem 6. *Let $\boldsymbol{Q} = (\mathcal{C}ov_A, \boldsymbol{q})$ be an OMQ where $\boldsymbol{q}$ is an F-path CQ without twins having a single binary relation. The following are equivalent unless* NL $=$ P:

(i) $\boldsymbol{Q}$ *is* NL*-complete*;
(ii) $\{0\} \cup N_{\boldsymbol{q}}$ *is an arithmetic progression*;
(iii) *there exist $f_i \colon \boldsymbol{q}_i \twoheadrightarrow \boldsymbol{q}$ for every $i = 1, \dots, m$.*

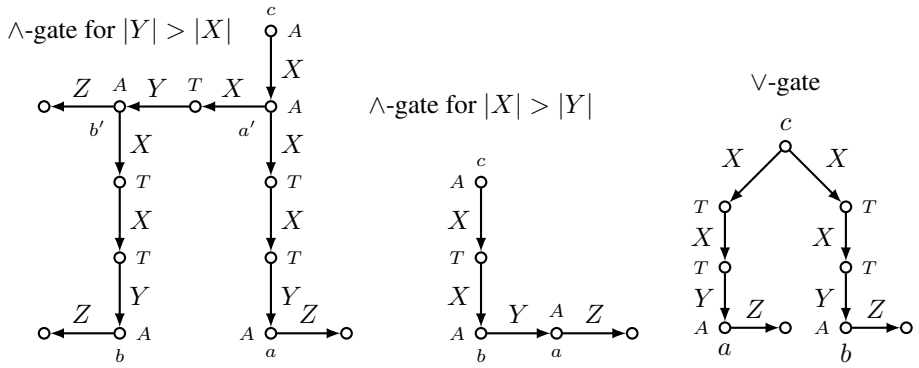
If these conditions do not hold, then $\boldsymbol{Q}$ is P*-complete.*

Proof. It is readily seen, using Theorem 5, that $(ii) \Rightarrow (iii) \Rightarrow (i)$. We prove the implication $(i) \Rightarrow (ii)$. Suppose $\{0\} \cup N_{\boldsymbol{q}}$ is not an arithmetic progression. We show that $\boldsymbol{Q}$ is P-hard by reduction of the monotone circuit evaluation problem. Let X be the sub-CQ of $\boldsymbol{q}$ between $F(x)$ and the first T but without the F and T atoms, and let $|X|$ be the number of atoms in X. Let Y be the first sub-CQ between neighbouring T-atoms without these T-atoms such that $|Y| \neq |X|$, n the number of X-sub-CQs before Y, and let Z be the rest of the query without the first T atom; see the picture below where

$n = 2$.



We distinguish between two cases: $|Y| > |X|$ and $|Y| < |X|$. Depending on the case, we use the following gadgets for $\wedge$-gates, which are shown below for $n = 2$ (it should be clear how to modify the construction for $n > 2$); the $\vee$-gate gadget is the same for both cases.



Given any monotone Boolean circuit C and its input α, we construct an ABox $\mathcal{A}_{C,\alpha}$ by replacing $\wedge$- and $\vee$-gates with the inputs a and b and output c by the gadgets above, placing T on the input gates evaluated to 1 under α and F on the output gate. We claim that $\mathcal{C}ov_A, \mathcal{A}_{C,\alpha} \models \boldsymbol{q}$ iff $C(\alpha) = 1$.

($\Leftarrow$) It is not hard to show by induction that whenever a non-input gate g in C outputs 1 under α, then the sub-ABox of $\mathcal{A}_{C,\alpha}$ generated by g and with F placed on g validates $\boldsymbol{Q}$. For example, suppose $T(a)$, $T(b)$ and $F(c)$ hold in the left-hand side gadget. Then either $F(b')$ or $T(b')$. In the former case, $\boldsymbol{q}$ is satisfied; so assume $T(b')$. We have either $F(a')$ or $T(a')$. Now, in either case, $\boldsymbol{q}$ is satisfied, as required.

($\Rightarrow$) Suppose $C(\alpha) = 0$. We construct a model $\mathcal{I}$ of $\mathcal{C}ov_A$ based on $\mathcal{A}_{C,\alpha}$ by placing T or F to g and g' (if available) depending on the value of g in C under α. It can be checked that $\mathcal{I} \not\models \boldsymbol{q}$. In particular, if $|Y| > |X|$, c is the conjunction of b and a, $c = b = 0$ and $a = 1$, then if x_0 goes to c, x_1 goes to a', then x_2 must go to the point below a', and so x_3 must go somewhere between a and the point above a, which is impossible. If $a = 0, b = 1$ and x_0 goes to a' and x_1 goes to the central T, then x_2 must go somewhere between the central T and b', which is also impossible. If $|Y| < |X|$, $c = b \wedge a$, $b = d \vee e$, $a = c = 0$, $b = 1$, then if x_0 goes to c, x_2 goes to b, then x_3 cannot go to a as $a = 0$, and so it must go into some X-segment of the gate $b = d \vee e$ going out of b, which is again impossible.

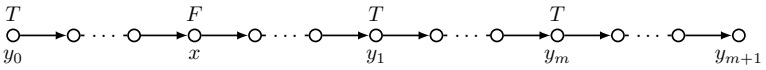
Note that the proof of P-hardness in Theorem 6 does not go through for $A = \top$. Thus, for $(\mathcal{C}ov_\top, \boldsymbol{q}_{T1T})$, we are in the framework of Example 1 and, by Theorem 4 (ii), this OMQ is in NL. In fact, we have the following NL/P dichotomy for the OMQs of the form $(\mathcal{C}ov_\top, \boldsymbol{q}_{TnT})$.

Theorem 7. (i) *The OMQ* $(\mathcal{C}ov_\top, \boldsymbol{q}_{T1T})$ *is* NL*-complete* (*whereas* $(\mathcal{C}ov_A, \boldsymbol{q}_{T1T})$ *is* P*-complete*).

(ii) *The OMQs* $(\mathcal{C}ov_\top, \boldsymbol{q}_{TnT})$ *(and* $(\mathcal{C}ov_A, \boldsymbol{q}_{TnT})$*), for* $n \geq 2$*, are* P*-complete.*

Proof. The proof is similar to that of P-hardness in Theorem 6.

We now apply Theorem 4 (ii) to the class of TF*-path CQs* of the form

$\boldsymbol{q}_{TF} \;=$ 

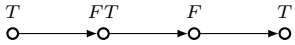
where the $T(y_i)$ and $F(x)$ are all the solitary occurrences of T and F in $\boldsymbol{q}_{TF}$. We represent this CQ as

$$\boldsymbol{q}_{TF} = \{T(y_0)\} \cup \boldsymbol{q}_0 \cup \boldsymbol{q},$$

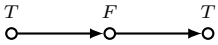
where $\boldsymbol{q}_0$ is the sub-CQ of $\boldsymbol{q}_{TF}$ between y_0 and x with $T(y_0)$ removed and $\boldsymbol{q}$ is the same as in Theorem 5 (and $\boldsymbol{q}^*_{m+1}$ is $\boldsymbol{q}$ without $F(x)$).

Theorem 8. *If* $\boldsymbol{q}$ *satisfies the condition of Theorem 5 and there is a homomorphism* $h\colon \boldsymbol{q}^*_{m+1} \to \boldsymbol{q}_0$ *such that* $h(x) = y_0$*, then answering* $(\mathcal{C}ov_A, \boldsymbol{q}_{TF})$ *is* NL*-complete.*

For example, the OMQ $(\mathcal{C}ov_A, \boldsymbol{q})$ with $\boldsymbol{q}$ shown below is NL-complete:



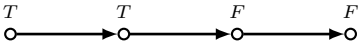
On the other hand, as shown in [9], $(\mathcal{C}ov_A, \boldsymbol{q})$ with $\boldsymbol{q}$ of the form



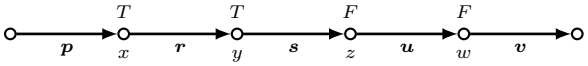
is P-complete; in fact, it follows from the proof that $(\mathcal{C}ov_\top, \boldsymbol{q})$ is P-complete, too.

6 2-CQs

A *2-CQ* has at least two solitary F and at least two solitary T. For example, as shown in [9], answering $(\mathcal{C}ov_A, \boldsymbol{q})$ with the following CQ $\boldsymbol{q}$ is coNP-complete:



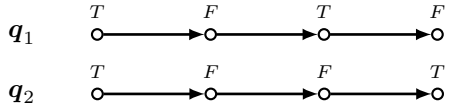
The proof can be generalised to the class of *2-2-CQs*, which are path 2-CQs where all the F are located after all the T, and every occurrence of T or F is solitary. We represent any given 2-2-CQ $\boldsymbol{q}$ as shown below



where $\boldsymbol{p}$, $\boldsymbol{r}$, $\boldsymbol{u}$ and $\boldsymbol{v}$ do not contain F and T, while $\boldsymbol{s}$ may contain solitary occurrences of both T and F (in other words, the T shown in the picture are the first two occurrences of T in $\boldsymbol{q}$ and the F are the last two occurrences of F in $\boldsymbol{q}$). Denote by $\boldsymbol{q}_{\boldsymbol{r}}$ the suffix of $\boldsymbol{q}$ that starts from x but without $T(x)$; similarly, $\boldsymbol{q}_{\boldsymbol{u}}$ is the suffix of $\boldsymbol{q}$ starting from z but without $F(z)$. Denote by $\boldsymbol{q}^-_{\boldsymbol{r}}$ the prefix of $\boldsymbol{q}$ that ends at y but without $T(y)$; similarly, $\boldsymbol{q}^-_{\boldsymbol{u}}$ is the prefix of $\boldsymbol{q}$ ending at w but without $F(w)$. Using the construction from [9], one can show the following:

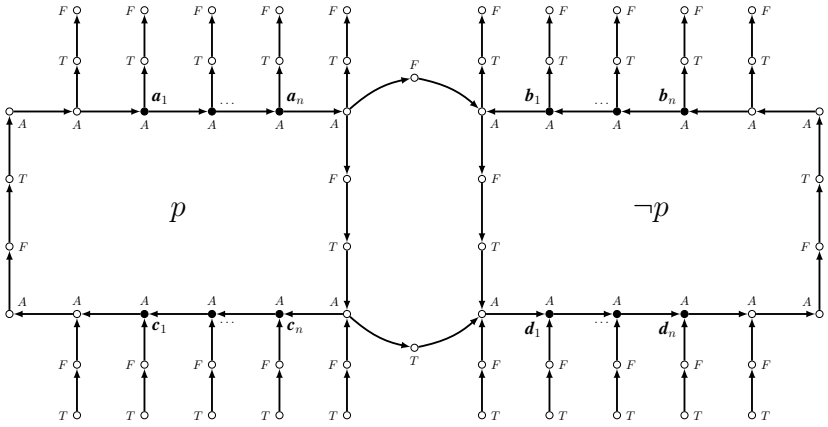
Theorem 9. *Any OMQ* $(\mathcal{C}ov_A, \boldsymbol{q})$ *with a 2-2-CQ* $\boldsymbol{q}$ *is* coNP*-complete provided the following conditions are satisfied:* (i) *there is no homomorphism* $h_1 \colon \boldsymbol{q}_{\boldsymbol{u}} \to \boldsymbol{q}_{\boldsymbol{r}}$ *with* $h_1(z) = x$*, and* (ii) *there is no homomorphism* $h_2 \colon \boldsymbol{q}_{\boldsymbol{r}}^- \to \boldsymbol{q}_{\boldsymbol{u}}^-$ *with* $h_2(y) = w$*.*

Unfortunately, the proof does not generalise to the other two path 2-CQs with four variables, whose complexity has so far been open:



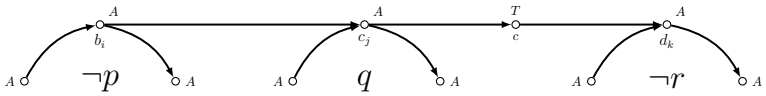
Theorem 10. *The OMQs* $\boldsymbol{Q}_1 = (\mathcal{C}ov_A, \boldsymbol{q}_1)$ *and* $\boldsymbol{Q}_2 = (\mathcal{C}ov_A, \boldsymbol{q}_2)$ *are* coNP*-complete.*

Proof. We prove coNP-hardness of $\boldsymbol{Q}_1$ by reduction of the NP-complete 3SAT. Given a 3CNF ψ, we construct an ABox $\mathcal{A}_\psi$ as follows. First, for every variable p occurring in ψ, we take the following p*-gadget*, where n is the number of clauses in ψ:



The key property of the p-gadget is that, for any model $\mathcal{I}$ of $\mathcal{C}ov_A$ based on this gadget, if $\mathcal{I} \not\models \boldsymbol{q}_1$ then either the A-points on left-hand side of the gadget are all in $T^\mathcal{I}$ and the A-points on the right-hand side are all in $F^\mathcal{I}$, or the other way round. We refer to the a_i and b_i as $p^\uparrow$- and $\neg p^\uparrow$*-contacts*, and to the c_i and d_i as $p^\downarrow$- and $\neg p^\downarrow$*-contacts*, respectively.

Now, for every clause $c = (l_1 \vee l_2 \vee l_3)$ in ψ, we add to the constructed gadgets for the variables in ψ the atoms $R(u^c_{\neg l_1}, v^c_{l_2})$, $R(v^c_{l_2}, c)$, $T(c)$, $R(c, w^c_{l_3})$, where c is a new individual, $u^c_{\neg l_1}$ a fresh $\neg l_1^\uparrow$-contact, $v^c_{l_2}$ a fresh $l_2^\downarrow$-contact, and $w^c_{l_3}$ a fresh $l_3^\downarrow$-contact. For example, for the clause $c = (p \vee q \vee \neg r)$, we obtain the fragment below:



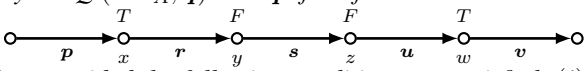
The resulting ABox $\mathcal{A}_\psi$ is such that ψ is satisfiable iff $\mathcal{C}ov_A, \mathcal{A}_\psi \not\models \boldsymbol{q}_1$.

For the OMQ $\boldsymbol{Q}_2$, we use (simplified) p-gadgets presented in Theorem 11 and connect them similarly to the picture above with T replaced by F at c.

We now sketch two generalisation of Theorem 10.

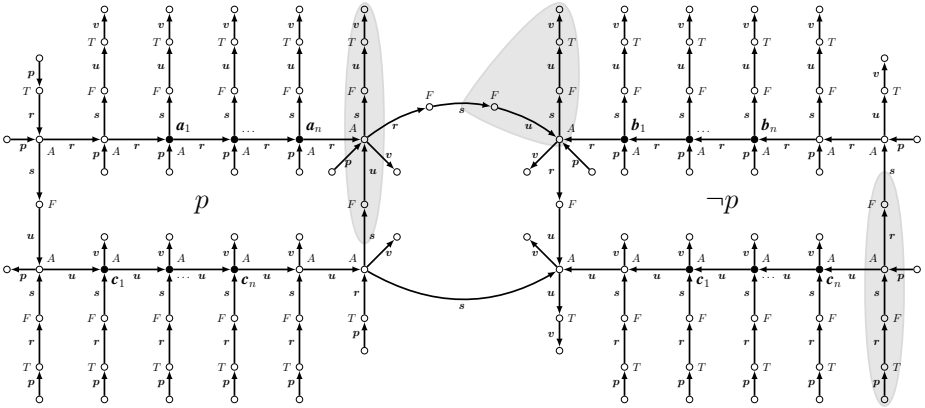
In Theorem 11 and 12, we assume that $\boldsymbol{p}$ and $\boldsymbol{v}$ do not contain F and T, while $\boldsymbol{r}$ and $\boldsymbol{u}$ may only contain solitary occurrences of T ($F \notin \boldsymbol{r}, \boldsymbol{u}$), and $\boldsymbol{s}$ only solitary occurrences of F ($T \notin \boldsymbol{s}$).

Theorem 11. *Any OMQ* $(\mathcal{C}ov_A, \boldsymbol{q})$ *with* $\boldsymbol{q}$ *of the form*

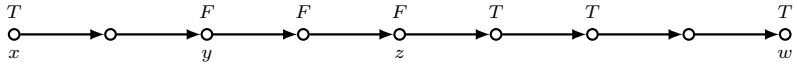


is coNP*-complete provided the following conditions are satisfied:* (i) *there is no homomorphism* $h_1 \colon \boldsymbol{r_t} \to \boldsymbol{u}$ *with* $h_1(y) = w$*, and* (ii) *there is no homomorphism* $h_2 \colon \boldsymbol{u_t} \to \boldsymbol{r}$ *with* $h_2(z) = x$*, where* $\boldsymbol{r}$ *is the sub-CQ of* $\boldsymbol{q}$ *between* x *and* y *without* $T(x)$, $F(y)$*, and similarly for* $\boldsymbol{u}$, $\boldsymbol{r_t}$ *is* $\boldsymbol{r}$ *with* $T(x)$ *and* $\boldsymbol{u_t}$ *is* $\boldsymbol{u}$ *with* $T(w)$.

The proof uses the following p-gadget:

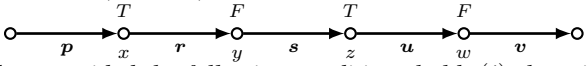


If one of the two conditions of this theorem is not satisfied then the given gadget will not work; see the shaded parts. For example, the complexity of the OMQ with the CQ below is still unknown:



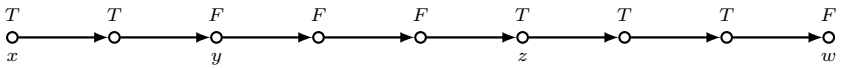
In Theorem 12, we use $\boldsymbol{r}^{ext} = \boldsymbol{r}(x, y) \wedge T(y) \wedge \boldsymbol{s}_1(y, y_1) \wedge F(y_1)$, where $\boldsymbol{s_1}$ is the part of $\boldsymbol{s}$ such that $\boldsymbol{s}(y, z) = \boldsymbol{s}_1(y, y_1) \wedge F(y_1) \wedge \boldsymbol{s}_2(y_1, z)$ and $\boldsymbol{s}_1(y, y_1)$ does not contain any occurrences of F. In other words, the variable y_1 corresponds to the first appearance of F in $\boldsymbol{s}$, where $\boldsymbol{s}$ is the sub-CQ of $\boldsymbol{q}$ between y and z without $F(y), T(z)$.

Theorem 12. *Any OMQ* $(\mathcal{C}ov_A, \boldsymbol{q})$ *with* $\boldsymbol{q}$ *of the form*



is coNP*-complete provided the following conditions hold:* (i) *there is no homomorphism* $g_1 \colon \boldsymbol{r_t} \to \boldsymbol{u}$ *with* $g_1(y) = w$*, and* (ii) *there is no homomorphism* $g_2 \colon \boldsymbol{u} \to \boldsymbol{r}^{ext}$ *with* $g_2(z) = x$ *and* $g_2(w) = y_1$.

The CQ below does not satisfy (ii), and its complexity is unknown:



It is also unknown whether there are OMQs with 2CQs in P.

Acknowledgements. The work of O. Gerasimova and M. Zakharyaschev was carried out at the National Research University Higher School of Economics and supported by the Russian Science Foundation under grant 17-11-01294.

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley (1995)
2. Afrati, F.N., Gergatsoulis, M., Toni, F.: Linearisability on datalog programs. Theor. Comput. Sci. 308(1-3), 199–226 (2003), https://doi.org/10.1016/S0304-3975(02)00730-2
3. Arora, S., Barak, B.: Computational Complexity: A Modern Approach. Cambridge University Press, New York, NY, USA, 1st edn. (2009)
4. Benedikt, M., ten Cate, B., Colcombet, T., Vanden Boom, M.: The complexity of boundedness for guarded logics. In: 30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015. pp. 293–304. IEEE Computer Society (2015), https://doi.org/10.1109/LICS.2015.36
5. Bienvenu, M., ten Cate, B., Lutz, C., Wolter, F.: Ontology-based data access: A study through disjunctive datalog, CSP, and MMSNP. ACM Transactions on Database Systems 39(4), 33:1–44 (2014)
6. Bulatov, A.A.: A dichotomy theorem for nonuniform csps. In: Umans, C. (ed.) 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017. pp. 319–330. IEEE Computer Society (2017), https://doi.org/10.1109/FOCS.2017.37
7. Cosmadakis, S.S., Gaifman, H., Kanellakis, P.C., Vardi, M.Y.: Decidable optimization problems for database logic programs (preliminary report). In: STOC. pp. 477–490 (1988)
8. Feier, C., Kuusisto, A., Lutz, C.: Rewritability in monadic disjunctive datalog, mmsnp, and expressive description logics (invited talk). In: Benedikt, M., Orsi, G. (eds.) 20th International Conference on Database Theory, ICDT 2017, March 21-24, 2017, Venice, Italy. LIPIcs, vol. 68, pp. 1:1–1:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2017), https://doi.org/10.4230/LIPIcs.ICDT.2017.1
9. Gerasimova, O., Kikot, S., Podolskii, V., Zakharyaschev, M.: On the data complexity of ontology-mediated queries with a covering axiom. In: Proceedings of the 30th International Workshop on Description Logics (2017)
10. Hernich, A., Lutz, C., Ozaki, A., Wolter, F.: Schema.org as a description logic. In: Calvanese, D., Konev, B. (eds.) Proceedings of the 28th International Workshop on Description Logics, Athens,Greece, June 7-10, 2015. CEUR Workshop Proceedings, vol. 1350. CEUR-WS.org (2015), http://ceur-ws.org/Vol-1350/paper-24.pdf
11. Kaminski, M., Nenov, Y., Grau, B.C.: Datalog rewritability of disjunctive datalog programs and non-Horn ontologies. Artif. Intell. 236, 90–118 (2016), http://dx.doi.org/10.1016/j.artint.2016.03.006
12. Kharlamov, E., Hovland, D., Skjæveland, M.G., Bilidas, D., Jiménez-Ruiz, E., Xiao, G., Soylu, A., Lanti, D., Rezk, M., Zheleznyakov, D., Giese, M., Lie, H., Ioannidis, Y.E., Kotidis, Y., Koubarakis, M., Waaler, A.: Ontology based data access in statoil. J. Web Sem. 44, 3–36 (2017), https://doi.org/10.1016/j.websem.2017.05.005
13. Lanti, D., Rezk, M., Xiao, G., Calvanese, D.: The NPD benchmark: Reality check for OBDA systems. In: Alonso, G., Geerts, F., Popa, L., Barceló, P., Teubner, J., Ugarte, M., den Bussche, J.V., Paredaens, J. (eds.) Proceedings of the 18th International Conference on Extending Database Technology, EDBT 2015, Brussels, Belgium, March 23-27, 2015. pp. 617–628. OpenProceedings.org (2015), https://doi.org/10.5441/002/edbt.2015.62

14. Lutz, C., Sabellek, L.: Ontology-mediated querying with the description logic EL: trichotomy and linear datalog rewritability. In: Sierra, C. (ed.) Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017. pp. 1181–1187. ijcai.org (2017), https://doi.org/10.24963/ijcai.2017/164
15. Marcinkowski, J.: DATALOG sirups uniform boundedness is undecidable. In: Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science, New Brunswick, New Jersey, USA, July 27-30, 1996. pp. 13–24. IEEE Computer Society (1996), https://doi.org/10.1109/LICS.1996.561299
16. Ramakrishnan, R., Sagiv, Y., Ullman, J.D., Vardi, M.Y.: Proof-tree transformation theorems and their applications. In: Proceedings of the eighth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems. pp. 172–181. ACM (1989)
17. Saraiya, Y.P.: Linearizing nonlinear recursions in polynomial time. In: Silberschatz, A. (ed.) Proceedings of the Eighth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, March 29-31, 1989, Philadelphia, Pennsylvania, USA. pp. 182–189. ACM Press (1989), http://doi.acm.org/10.1145/73721.73740
18. Schaerf, A.: On the complexity of the instance checking problem in concept languages with existential quantification. J. of Intelligent Information Systems 2, 265–278 (1993)
19. Ullman, J.D., Gelder, A.V.: Parallel complexity of logical query programs. Algorithmica 3, 5–42 (1988), https://doi.org/10.1007/BF01762108
20. Vardi, M.Y.: Decidability and undecidability results for boundedness of linear recursive queries. In: Edmondson-Yurkanan, C., Yannakakis, M. (eds.) Proceedings of the Seventh ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, March 21-23, 1988, Austin, Texas, USA. pp. 341–351. ACM (1988), http://doi.acm.org/10.1145/308386.308470
21. Zhang, W., Yu, C.T., Troy, D.: Necessary and sufficient conditions to linearize double recursive programs in logic databases. ACM Trans. Database Syst. 15(3), 459–482 (1990), http://doi.acm.org/10.1145/88636.89237
22. Zhuk, D.: A proof of CSP dichotomy conjecture. In: Umans, C. (ed.) 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017. pp. 331–342. IEEE Computer Society (2017), https://doi.org/10.1109/FOCS.2017.38