Deciding FO-Rewritability of Ontology-Mediated Queries in Linear Temporal Logic

Vladislav Ryzhikov ✉
Department of Computer Science, Birkbeck, University of London, UK

Yury Savateev ✉
Department of Computer Science, Birkbeck, University of London, UK
HSE University, Moscow, Russia

Michael Zakharyaschev ✉
Department of Computer Science, Birkbeck, University of London, UK
HSE University, Moscow, Russia

Abstract

Our concern is the problem of determining the data complexity of answering an ontology-mediated query (OMQ) given in linear temporal logic *LTL* over $(\mathbb{Z}, <)$ and deciding whether it is rewritable to an $\mathsf{FO}(<)$-query, possibly with extra predicates. First, we observe that, in line with the circuit complexity and FO-definability of regular languages, OMQ answering in AC^0, ACC^0 and NC^1 coincides with $\mathsf{FO}(<, \equiv)$-rewritability using unary predicates $x \equiv 0 \,(\mathrm{mod}\; n)$, $\mathsf{FO}(<, \mathsf{MOD})$-rewritability, and $\mathsf{FO}(\mathsf{RPR})$-rewritability using relational primitive recursion, respectively. We then show that deciding $\mathsf{FO}(<)$-, $\mathsf{FO}(<, \equiv)$- and $\mathsf{FO}(<, \mathsf{MOD})$-rewritability of *LTL* OMQs is ExpSpace-complete, and that these problems become PSpace-complete for OMQs with a linear Horn ontology and an atomic query, and also a positive query in the cases of $\mathsf{FO}(<)$- and $\mathsf{FO}(<, \equiv)$-rewritability. Further, we consider $\mathsf{FO}(<)$-rewritability of OMQs with a binary-clause ontology and identify OMQ classes, for which deciding it is PSpace-, Π_2^p- and coNP-complete.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics

Keywords and phrases Linear temporal logic, ontology-mediated query, first-order rewritability

Digital Object Identifier 10.4230/LIPIcs.TIME.2021.10

Funding This work was supported by the UK EPSRC grant EP/S032282.

Acknowledgements The open access publication of this article was supported by the Alpen-Adria-Universität Klagenfurt, Austria.

1 Introduction

Motivation. The problem we consider in this paper originates in the area of ontology-based data access (OBDA) to temporal data. The aim of the OBDA paradigm [38, 51] and systems such as Mastro or Ontop[1] is to facilitate management and integration of possibly incomplete and heterogeneous data by providing the user with a view of the data through the lens of a description logic (DL) ontology. Thus, the user can think of the data as a "virtual knowledge graph" [52], $\mathcal{A}$, whose labels – unary and binary predicates supplied by an ontology, $\mathcal{O}$ – are the only thing to know when formulating queries, $\varkappa$. Ontology-mediated queries (OMQs) $\boldsymbol{q} = (\mathcal{O}, \varkappa)$ are supposed to be answered over $\mathcal{A}$ under the open world semantics (taking account of all models of $\mathcal{O}$ and $\mathcal{A}$), which can be prohibitively complex. So the key to practical OBDA is ensuring first-order rewritability of $\boldsymbol{q}$ (aka boundedness in the datalog literature [1]), which reduces open-world reasoning to evaluating an FO-formula over $\mathcal{A}$. The

[1] `https://www.obdasystems.com`, `https://ontopic.biz`

28th International Symposium on Temporal Representation and Reasoning (TIME 2021).
Editors: Carlo Combi, Johann Eder, and Mark Reynolds; Article No. 10; pp. 10:1–10:15
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

W3C standard ontology language *OWL 2 QL* for OBDA is based on the *DL-Lite* family of DL [3,17], which uniformly guarantees FO-rewritability of all OMQs with a conjunctive query. Other ontology languages with this feature include various dialects of tgds; see, e.g., [7,16,19]. However, by design such languages are rather inexpressive.

Theory and practice of OBDA have revived the interest to the problem of deciding whether an OMQ given in some expressive language is FO-rewritable, which was thoroughly investigated in the 1980–90s for datalog queries; see, e.g., [2,21,37,44,46]. The data complexity and rewritability of OMQs in various DLs and disjunctive datalog have become an active research area in the past decade [14,24,28,29,36], lying at the crossroads of logic, database theory, knowledge representation, circuit and descriptive complexity, and CSP.

There have been numerous attempts to extend ontology and query languages with constructors capable of representing events over temporal data; see [5, 35] for surveys and [15,49,50] for more recent developments. However, so far the focus has been on the uniform complexity of reasoning with arbitrary ontologies and queries in a given language rather than on understanding the data complexity and FO-rewritability of individual temporal OMQs. On the other hand, the non-uniform analysis of OMQs in DLs or datalog mentioned above is not applicable to standard temporal logics interpreted over linearly-ordered structures.

In this paper, we take a first step towards understanding the problem of FO-rewritability of OMQs over temporal data by focusing on the temporal dimension and considering OMQs given in linear temporal logic *LTL* interpreted over $(\mathbb{Z}, <)$.

▶ **Example 1.** Let $\mathcal{O}$ be an *LTL* ontology with the following axioms (describing a system's behaviour and) containing the temporal operators $\Box_F/\Box_P$ (always in the future/past), $\Diamond_F/\Diamond_P$ (sometime in the future/past) and $\bigcirc_F/\bigcirc_P$ (the next/previous minute):

$$\Box_P\Box_F\big(\mathit{Malfunction} \to \Diamond_F \mathit{Fixed}\big), \tag{1}$$

$$\Box_P\Box_F\big(\mathit{Fixed} \to \bigcirc_F \mathit{InOperation}\big), \tag{2}$$

$$\Box_P\Box_F\big(\mathit{Malfunction} \land \bigcirc_P \mathit{Malfunction} \land \bigcirc_P^2 \mathit{Malfunction} \to \neg\bigcirc_F \mathit{InOperation}\big). \tag{3}$$

We query temporal data, say

$$\mathcal{A} = \{\mathit{Malfunction}(2), \mathit{Malfunction}(5), \mathit{Malfunction}(6), \mathit{Fixed}(6), \mathit{Malfunction}(7)\}$$

by means of *LTL*-formulas such as

$$\varkappa = \Diamond_P\Diamond_F\big(\mathit{Malfunction} \land \bigvee_{1\le i\le 5} \bigcirc_F^i(\mathit{Fixed} \land \bigvee_{1\le j\le 5} \neg\bigcirc_F^j \mathit{InOperation})\big)$$

asking whether there was a malfunction that was fixed in $\le$ 5m but within the next 5m the equipment went out of operation again. The certain answer to the OMQ $\boldsymbol{q} = (\mathcal{O}, \varkappa)$ over $\mathcal{A}$ is yes because $\varkappa$ is true in all models of $\mathcal{O}$ and $\mathcal{A}$. It is readily seen that the certain answer to $\boldsymbol{q}$ over any given data instance $\mathcal{A}'$ in the signature $\{\mathit{Malfunction}, \mathit{Fixed}\}$ can be computed by evaluating over $\mathcal{A}'$ the following $\mathsf{FO}(<)$-sentence, called an $\mathsf{FO}(<)$-rewriting of $\boldsymbol{q}$:

$$\exists x\, \big[\mathit{Malfunction}(x) \land \bigvee_{1\le i\le 5} \big(\mathit{Fixed}(x+i) \land \bigvee_{1\le j\le 5} \bigwedge_{0\le k\le 2} \mathit{Malfunction}(x+i+j-k)\big)\big].$$

Problem and related work. The problem we are interested in can be formulated in complexity-theoretic terms: given an *LTL* OMQ $\boldsymbol{q}$, determine the data complexity of answering $\boldsymbol{q}$ over any data instance $\mathcal{A}$ in a given signature Ξ. For simplicity's sake, let us assume that $\boldsymbol{q}$ is Boolean (with a yes/no answer). Then the data instances $\mathcal{A}$, over which the answer to $\boldsymbol{q}$ is yes, form a language $\boldsymbol{L}(\boldsymbol{q})$ over the alphabet 2^{Ξ}. In fact, using the automata-theoretic view of *LTL* [48], one can show that $\boldsymbol{L}(\boldsymbol{q})$ is regular, and so can be decided in NC^1 [8, 10]. The circuit and descriptive complexity of regular languages was investigated in [9, 43], which established an $\mathrm{AC}^0/\mathrm{ACC}^0/\mathrm{NC}^1$ trichotomy, gave algebraic characterisations of languages in these classes (implying that the trichotomy is decidable) and also in terms of extensions of FO. Namely, the languages $\boldsymbol{L}$ in AC^0 are definable by $\mathsf{FO}(<,\equiv)$-sentences with unary predicates $x \equiv 0 \,(\mathrm{mod}\ n)$; those in ACC^0 are definable by $\mathsf{FO}(<,\mathsf{MOD})$-sentences with quantifiers $\exists^n x\, \psi(x)$ checking whether the number of positions satisfying ψ is divisible by n; and all regular languages $\boldsymbol{L}$ are definable in $\mathsf{FO}(\mathsf{RPR})$ with relational primitive recursion [20].

Thus, our problem can be equivalently formulated in logic terms: given an *LTL* OMQ $\boldsymbol{q}$, decide whether $\boldsymbol{L}(\boldsymbol{q})$ is $\mathsf{FO}(<,\equiv)$- or $\mathsf{FO}(<,\mathsf{MOD})$-definable. In the OBDA context, we are also interested in $\mathsf{FO}(<)$-definability (without any extra predicates, quantifiers or recursion), which has been thoroughly investigated in both automata theory and logic; see, e.g., [23] and references therein. In particular, deciding $\mathsf{FO}(<)$-definability of regular languages given by a NFA can be done in PSpace [13, 41], with a matching lower bound established even for languages given by a minimal DFA [18]. These classical results have recently been extended by showing that deciding each of $\mathsf{FO}(<)$-, $\mathsf{FO}(<,\equiv)$-, and $\mathsf{FO}(<,\mathsf{MOD})$-definability of languages given by a two-way NFA can be done in PSpace, and that a matching lower bound holds for languages given by a minimal DFA [33]. Note also that, by Kamp's Theorem [30, 39], $\mathsf{FO}(<)$-rewritability reduces answering *LTL* OMQs to model checking *LTL*-formulas.

$\mathsf{FO}(\mathsf{RPR})$-rewritability of all *LTL* OMQs was proved in [6], which also provided (uniform) rewritability results for various classes of *LTL* OMQs (to be defined below); see Table 2.

Our contribution. Let $\mathcal{L} \in \{\mathsf{FO}(<), \mathsf{FO}(<,\equiv), \mathsf{FO}(<,\mathsf{MOD})\}$. To investigate $\mathcal{L}$-rewritability of *LTL* OMQs $\boldsymbol{q} = (\mathcal{O}, \varkappa)$, we follow the classification of [6], according to which the axioms of every *LTL* ontology $\mathcal{O}$ are given in the clausal form

$$\Box_P \Box_F \big(C_1 \land \dots \land C_k \ \to\ C_{k+1} \lor \dots \lor C_{k+m}\big), \tag{4}$$

where the C_i are atoms, possibly prefixed by the temporal operators $\bigcirc_F$, $\bigcirc_P$, $\Box_F$, $\Box_P$. Given some $\boldsymbol{o} \in \{\Box, \bigcirc, \Box\bigcirc\}$ and $\boldsymbol{c} \in \{\mathit{bool}, \mathit{horn}, \mathit{krom}, \mathit{core}\}$, we denote by $\mathit{LTL}^{\boldsymbol{o}}_{\boldsymbol{c}}$ the fragment of *LTL* with clauses of the form (4), where the C_i can only use the (future and past) operators indicated in $\boldsymbol{o}$, and $m \le 1$ if $\boldsymbol{c} = \mathit{horn}$; $k+m \le 2$ if $\boldsymbol{c} = \mathit{krom}$; $k+m \le 2$ and $m \le 1$ if $\boldsymbol{c} = \mathit{core}$; and arbitrary k, m if $\boldsymbol{c} = \mathit{bool}$. If $\boldsymbol{o}$ is omitted, the C_i are atomic. An $\mathit{LTL}^{\boldsymbol{o}}_{\mathit{horn}}$-ontology $\mathcal{O}$ is linear if, in each of its axioms (4), at most one C_i, for $1 \le i \le k$, can occur on the right-hand side of an axiom in $\mathcal{O}$ (is an IDB predicate, in datalog parlance). We distinguish between arbitrary $\mathit{LTL}^{\boldsymbol{o}}_{\boldsymbol{c}}$ OMQs $\boldsymbol{q} = (\mathcal{O}, \varkappa)$, where $\mathcal{O}$ is any $\mathit{LTL}^{\boldsymbol{o}}_{\boldsymbol{c}}$ ontology and $\varkappa$ any *LTL*-formula with $\bigcirc$-, $\Box$- and $\Diamond$-operators; positive OMQs (OMPQs), where $\varkappa$ is $\to,\neg$-free; existential OMPQs (OMPEQs) with $\Box$-free $\varkappa$; and atomic OMQs (OMAQs) with atomic $\varkappa$.

The main result of this paper is the tight complexity bounds on deciding $\mathcal{L}$-rewritability (and so data complexity) of *LTL* OMQs in various classes defined above, which are summarised in Table 1. The ExpSpace upper bound in the first stripe is shown using the $\mathcal{L}$-definability criteria recently obtained in [33] and exponential-size NFAs for *LTL* akin to those in [47]; in the proof of the matching lower bound, an exponential-size automaton is encoded in a polynomial-size ontology. If the ontology in an $\mathit{LTL}^{\bigcirc}_{\mathit{horn}}$ OMAQ is linear, we show that its language (yes-data instances) can be captured by a 2NFA with polynomially many states, which allows us to reduce the complexity of deciding $\mathcal{L}$-rewritability to PSpace. However, for linear $\mathit{LTL}^{\bigcirc}_{\mathit{horn}}$ OMPQs (with more expressive queries $\varkappa$), the existence of polynomial-state

Table 1 Complexity of deciding FO-rewritability of *LTL* OMQs.

class of OMQs	FO(<)	FO(<, ≡), AC^0	FO(<, MOD), ACC^0
$LTL^{\bigcirc}_{horn}$ OMAQs LTL_{krom} OMPEQs $LTL^{\Box\bigcirc}_{bool}$ OMQs	ExpSpace	ExpSpace	ExpSpace
linear $LTL^{\bigcirc}_{horn}$ OMAQ linear $LTL^{\bigcirc}_{horn}$ OMPQs	PSpace	PSpace	PSpace ?
$LTL^{\bigcirc}_{krom}$ OMAQs $LTL^{\bigcirc}_{core}$ OMPEQs $LTL^{\bigcirc}_{core}$ OMPQs	coNP Π^p_2 PSpace	all in AC^0 [6]	–

2NFAs remains open; instead, we show how the structure of the canonical (minimal) models for $LTL^{\bigcirc}_{horn}$-ontologies can be utilised to yield a PSpace algorithm. In the third stripe of the table, we deal with binary-clause ontologies. The coNP-completeness of deciding FO-rewritability of $LTL^{\bigcirc}_{krom}$ OMAQs is established using unary NFAs and results from [42]. The Π^p_2-completeness for $LTL^{\bigcirc}_{core}$ OMPEQs (without $\vee$ in ontologies but with $\wedge$, $\vee$, $\Diamond$ in queries) and the PSpace-completeness for $LTL^{\bigcirc}_{core}$ OMPQs (admitting $\Box$ in queries, too) can be explained by the fact that the combined complexity of answering such OMPEQs and OMPQs is NP-hard rather than tractable as in the previous case.

All omitted details and proofs are provided in the full draft of the paper [40].

2 Preliminaries

Temporal ontology-mediated queries. In our setting, the alphabet of *LTL* comprises a set of *atomic concepts* A_i, $i < \omega$. *Basic temporal concepts*, C, are defined by the grammar $C ::= A_i \mid \Box_F C \mid \Box_P C \mid \bigcirc_F C \mid \bigcirc_P C$. A *temporal ontology*, $\mathcal{O}$, is a finite set of *axioms* in normal form (4) with $\Box_P\Box_F$ omitted. An $LTL^{\boldsymbol{o}}_{\boldsymbol{c}}$ *ontology-mediated query* (OMQ) is a pair $\boldsymbol{q} = (\mathcal{O}, \varkappa)$, where $\mathcal{O}$ is an $LTL^{\boldsymbol{o}}_{\boldsymbol{c}}$ ontology (defined above) and $\varkappa$ a *temporal concept* built from atoms A_i using the Booleans and temporal operators $\bigcirc_F$, $\Box_F$, $\Diamond_F$ and their past-time counterparts $\bigcirc_P$, $\Box_P$, $\Diamond_P$. The set of atomic concepts occurring in $\boldsymbol{q}$ is denoted by $\mathsf{sig}(\boldsymbol{q})$.

A *data instance* – *ABox* in description logic parlance – is a finite set $\mathcal{A}$ of atoms $A_i(\ell)$, for $\ell \in \mathbb{Z}$, together with a finite interval $\mathsf{tem}(\mathcal{A}) = [m, n] \subseteq \mathbb{Z}$, the *active domain* of $\mathcal{A}$, such that $m \leq \ell \leq n$, for all $A_i(\ell) \in \mathcal{A}$. If $\mathcal{A} = \emptyset$, then $\mathsf{tem}(\mathcal{A})$ may also be $\emptyset$. Otherwise, we assume (without loss of generality) that $m = 0$. If $\mathsf{tem}(\mathcal{A})$ is not specified explicitly, it is assumed to be either empty or $[0, n]$, where n is the maximal timestamp in $\mathcal{A}$. By a *signature*, Ξ, we mean any finite set of atomic concepts. An ABox $\mathcal{A}$ is a Ξ-*ABox* if $A_i(\ell) \in \mathcal{A}$ implies $A_i \in \Xi$.

A *temporal interpretation* is a structure of the form $\mathcal{I} = (\mathbb{Z}, A_0^{\mathcal{I}}, A_1^{\mathcal{I}}, \dots)$ with $A_i^{\mathcal{I}} \subseteq \mathbb{Z}$, for every $i < \omega$. The *extension* $\varkappa^{\mathcal{I}}$ of a temporal concept $\varkappa$ in $\mathcal{I}$ is defined inductively as usual in *LTL* under the "strict semantics" [22, 27]: $(\bigcirc_F \varkappa)^{\mathcal{I}} = \{ n \in \mathbb{Z} \mid n + 1 \in \varkappa^{\mathcal{I}} \}$, $(\Box_F \varkappa)^{\mathcal{I}} = \{ n \in \mathbb{Z} \mid k \in \varkappa^{\mathcal{I}} \text{ for all } k > n \}$, $(\Diamond_F \varkappa)^{\mathcal{I}} = \{ n \in \mathbb{Z} \mid \text{there is } k > n \text{ with } k \in \varkappa^{\mathcal{I}} \}$, and symmetrically for the past-time operators. We say that an axiom (4) is *true* in $\mathcal{I}$ if $C_1^{\mathcal{I}} \cap \dots \cap C_k^{\mathcal{I}} \subseteq C_{k+1}^{\mathcal{I}} \cup \dots \cup C_{k+m}^{\mathcal{I}}$. An interpretation $\mathcal{I}$ is a *model* of $\mathcal{O}$ if all axioms of $\mathcal{O}$ are true in $\mathcal{I}$; it is a *model* of $\mathcal{A}$ if $A_i(\ell) \in \mathcal{A}$ implies $\ell \in A_i^{\mathcal{I}}$.

We can treat $\boldsymbol{q}$ as a *Boolean* OMQ, which returns yes/no, or as a *specific* OMQ, which returns timestamps from the ABox in question assigned to the free variable, say x, in the standard FO-translation of $\varkappa$. In the latter case, we write $\boldsymbol{q}(x) = (\mathcal{O}, \varkappa(x))$. More precisely, a *certain answer* to a Boolean OMQ $\boldsymbol{q} = (\mathcal{O}, \varkappa)$ over an ABox $\mathcal{A}$ is yes if, for every model

Table 2 Rewritability of *LTL* OMQs [6].

	OMAQs		OMPQs	
$\boldsymbol{c}$	$LTL_{\boldsymbol{c}}^{\Box}$	$LTL_{\boldsymbol{c}}^{\bigcirc}$ and $LTL_{\boldsymbol{c}}^{\Box\bigcirc}$	$LTL_{\boldsymbol{c}}^{\Box}$	$LTL_{\boldsymbol{c}}^{\bigcirc}$ and $LTL_{\boldsymbol{c}}^{\Box\bigcirc}$
bool		$\mathsf{FO}(\mathsf{RPR})$	$\mathsf{FO}(\mathsf{RPR})$	
krom	$\mathsf{FO}(<)$	$\mathsf{FO}(<,\equiv)$		$\mathrm{FO(RPR)}$
horn		$\mathsf{FO}(\mathsf{RPR})$	$\mathsf{FO}(<)$	
core		$\mathsf{FO}(<,\equiv)$		$\mathsf{FO}(<,\equiv)$

$\mathcal{I}$ of $\mathcal{O}$ and $\mathcal{A}$, there is $k \in \mathbb{Z}$ such that $k \in \varkappa^{\mathcal{I}}$, in which case we write $(\mathcal{O}, \mathcal{A}) \models \exists x \varkappa(x)$. We write $(\mathcal{O}, \mathcal{A}) \models \varkappa(k)$, for $k \in \mathbb{Z}$, if $k \in \varkappa^{\mathcal{I}}$ in all models $\mathcal{I}$ of $\mathcal{O}$ and $\mathcal{A}$. A *certain answer* to a specific OMQ $\boldsymbol{q}(x) = (\mathcal{O}, \varkappa(x))$ over $\mathcal{A}$ is any $k \in \mathsf{tem}(\mathcal{A})$ with $(\mathcal{O}, \mathcal{A}) \models \varkappa(k)$. By the *evaluation* (or *answering*) *problem* for $\boldsymbol{q}$ or $\boldsymbol{q}(x)$ we understand the decision problem "$(\mathcal{O}, \mathcal{A}) \models^? \exists x \varkappa(x)$" or "$(\mathcal{O}, \mathcal{A}) \models^? \varkappa(k)$" with input $\mathcal{A}$ or, respectively, $\mathcal{A}$ and $k \in \mathsf{tem}(\mathcal{A})$.

▶ **Example 2.**

(i) Suppose $\mathcal{O}_1 = \{A \to \Box_{\!F} B,\ \Box_{\!F} B \to C\}$ and $\boldsymbol{q}_1 = (\mathcal{O}_1, C \wedge D)$. The certain answer to $\boldsymbol{q}_1$ over $\mathcal{A}_1 = \{D(0), B(1), A(1)\}$ is yes, and no over $\mathcal{A}_2 = \{D(0), A(1)\}$. The only answer to $\boldsymbol{q}_1(x) = \big(\mathcal{O}_1, (C \wedge D)(x)\big)$ over $\mathcal{A}_1$ is 0.

(ii) Let $\mathcal{O}_2 = \{\, \bigcirc_{\!P} A \to B,\ \bigcirc_{\!P} B \to A,\ A \wedge B \to \bot \,\}$. The certain answer to $\boldsymbol{q}_2 = (\mathcal{O}_2, C)$ over $\mathcal{A}_1 = \{A(0)\}$ is no, and yes over $\mathcal{A}_2 = \{A(0), A(1)\}$. There are no certain answers to $\boldsymbol{q}_2(x) = (\mathcal{O}_1, C(x))$ over $\mathcal{A}_1$, while over $\mathcal{A}_2$ the answers are 0 and 1.

(iii) Consider now the ontology $\mathcal{O}_3 = \{\bigcirc_{\!P} B_k \wedge A_0 \to B_k,\ \bigcirc_{\!P} B_{1-k} \wedge A_1 \to B_k \mid k = 0, 1\}$. For any word $\boldsymbol{e} = e_1 \dots e_n \in \{0,1\}^n$, let $\mathcal{A}_{\boldsymbol{e}} = \{B_0(0)\} \cup \{A_{e_i}(i) \mid 0 < i \le n\} \cup \{E(n)\}$. The answer to $\boldsymbol{q}_3 = (\mathcal{O}_3, B_0 \wedge E)$ over the ABox $\mathcal{A}_{\boldsymbol{e}}$ is yes iff the number of 1s in $\boldsymbol{e}$ is even.

▶ Remark 3. As follows from [4, 25], if arbitrary (boxed) *LTL*-formulas are used as axioms of an ontology $\mathcal{O}$, then one can construct an $LTL_{bool}^{\Box\bigcirc}$ ontology $\mathcal{O}'$ that is a model conservative extension of $\mathcal{O}$. For example, let $\mathcal{O}'$ be the result of replacing (1) in $\mathcal{O}$ from Example 1 by $\mathit{Malfunction} \wedge \Box_{\!F} X \to \bot$ and $\top \to X \vee \mathit{Fixed}$, for a fresh X. Then $\boldsymbol{q} = (\mathcal{O}, \varkappa)$ is equivalent to $\boldsymbol{q}' = (\mathcal{O}', \varkappa)$ in the sense that $\boldsymbol{q}$ and $\boldsymbol{q}'$ have the same certain answers over any $\mathsf{sig}(\boldsymbol{q})$-ABox.

Let $\mathcal{L} \in \{\mathsf{FO}(<), \mathsf{FO}(<,\equiv), \mathsf{FO}(<,\mathsf{MOD}), \mathsf{FO}(\mathsf{RPR})\}$. A Boolean OMQ $\boldsymbol{q}$ is $\mathcal{L}$-*rewritable over* Ξ-*ABoxes* if there is an $\mathcal{L}$-sentence $\boldsymbol{Q}$ such that, for any Ξ-ABox $\mathcal{A}$, the certain answer to $\boldsymbol{q}$ over $\mathcal{A}$ is yes iff $\mathfrak{S}_{\mathcal{A}} \models \boldsymbol{Q}$. Here, $\mathfrak{S}_{\mathcal{A}}$ is a structure with domain $\mathsf{tem}(\mathcal{A})$ ordered by $<$, in which $\mathfrak{S}_{\mathcal{A}} \models A_i(\ell)$ iff $A_i(\ell) \in \mathcal{A}$. A specific OMQ $\boldsymbol{q}(x)$ is $\mathcal{L}$-*rewritable over* Ξ-*ABoxes* if there is an $\mathcal{L}$-formula $\boldsymbol{Q}(x)$ with one free variable x such that, for any Ξ-ABox $\mathcal{A}$, k is a certain answer to $\boldsymbol{q}(x)$ over $\mathcal{A}$ iff $\mathfrak{S}_{\mathcal{A}} \models \boldsymbol{Q}(k)$. The sentence $\boldsymbol{Q}$ and the formula $\boldsymbol{Q}(x)$ are called $\mathcal{L}$-*rewritings* of the OMQs $\boldsymbol{q}$ and $\boldsymbol{q}(x)$, respectively. All $LTL_{bool}^{\Box\bigcirc}$ (Boolean and specific) OMQs are $\mathsf{FO}(\mathsf{RPR})$-rewritable. The *syntactic* classification of *LTL* OMQs by their rewritability type, obtained in [6], is shown in Table 2. It follows, e.g., that all $LTL_{core}^{\Box\bigcirc}$ OMPQs are $\mathsf{FO}(<,\equiv_{\mathbb{N}})$-rewritable, with some of them being not $\mathsf{FO}(<)$-rewritable. It is to be noted that $\mathsf{FO}(<,\mathsf{MOD})$-rewritable OMQs such as $\boldsymbol{q}_3$ in Example 2 and 4 are not captured by these syntactic classes.

▶ **Example 4.**

(i) An $\mathsf{FO}(<)$-rewriting of $\boldsymbol{q}_1(x)$ over arbitrary ABoxes is

$$\boldsymbol{Q}_1(x) = D(x) \land [C(x) \lor \exists y\,(A(y) \land \forall z\,((x < z \le y) \to B(z)))],$$

$\exists x\,\boldsymbol{Q}_1(x)$ is an $\mathsf{FO}(<)$-rewriting of $\boldsymbol{q}_1$.

(ii) An $\mathsf{FO}(<,\equiv)$-rewriting of $\boldsymbol{q}_2(x)$ is

$$\begin{aligned}\boldsymbol{Q}_2(x) = {}& C(x) \lor \exists x,y\,[(A(x) \land A(y) \land \mathsf{odd}(x,y)) \lor \\ & (B(x) \land B(y) \land \mathsf{odd}(x,y)) \lor (A(x) \land B(y) \land \neg\mathsf{odd}(x,y))],\end{aligned}$$

where $\mathsf{odd}(x,y) = \big(x \equiv 0\,(\mathrm{mod}\ 2) \leftrightarrow y \not\equiv 0\,(\mathrm{mod}\ 2)\big)$ implies that $|x-y|$ is odd, and an $\mathsf{FO}(<,\equiv)$-rewriting of $\boldsymbol{q}_2$ is $\exists x\,\boldsymbol{Q}_2(x)$. Recall that odd is not expressible in $\mathsf{FO}(<)$ [34].

(iii) The OMQ $\boldsymbol{q}_3$ is not rewritable to an FO-formula with any numeric predicates as PARITY is not in AC^0 [26]; the following sentence is an $\mathsf{FO}(<,\mathsf{MOD})$-rewriting of $\boldsymbol{q}_3$:

$$\begin{aligned}\boldsymbol{Q}_3 = \exists x,y\,\big[&E(x) \land (y \le x) \land \forall z\,\big((y < z \le x) \to A_0(z) \lor A_1(z)\big) \land \\ &\big((B_0(y) \land \exists^2 z\,((y < z \le x) \land A_1(z))) \lor (B_1(y) \land \neg\exists^2 z\,((y < z \le x) \land A_1(z)))\big)\big].\end{aligned}$$

In this paper, our aim is to understand how (complex it is) to decide the optimal type of FO-rewritability for a given *LTL* OMQ $\boldsymbol{q}$ over Ξ-ABoxes. Although all of our results hold for both Boolean and specific OMQs, here we only focus on the former; detailed proofs for the latter can be found in the full draft. We begin by observing an intimate connection between $\mathcal{L}$-rewritability of OMQs and $\mathcal{L}$-definability of certain regular languages.

Automata, languages, and OMQs. A *two-way nondeterministic finite automaton* is a quintuple $\mathfrak{A} = (Q, \Sigma, \delta, Q_0, F)$ that consists of an alphabet Σ, a finite set of states Q with a subset $Q_0 \neq \emptyset$ of initial states and a subset F of accepting states, and a transition function $\delta\colon Q \times \Sigma \to 2^{Q \times \{-1,0,1\}}$ indicating the next state and whether the head should move left (-1), right (1), or stay put (0). If $Q_0 = \{q_0\}$ and $|\delta(q,a)| = 1$, for all $q \in Q$ and $a \in \Sigma$, then $\mathfrak{A}$ is *deterministic*, in which case we write $\mathfrak{A} = (Q, \Sigma, \delta, q_0, F)$. If $\delta(q,a) \subseteq Q \times \{1\}$, for all $q \in Q$ and $a \in \Sigma$, then $\mathfrak{A}$ is a *one-way* automaton, and we write $\delta\colon Q \times \Sigma \to 2^Q$. As usual, DFA and NFA refer to one-way deterministic and non-deterministic finite automata, respectively, while 2DFA and 2NFA to the corresponding two-way automata. Given a 2NFA $\mathfrak{A}$, we write $q \to_{a,d} q'$ if $(q',d) \in \delta(q,a)$; given an NFA $\mathfrak{A}$, we write $q \to_a q'$ if $q' \in \delta(q,a)$. A *run* of a 2NFA $\mathfrak{A}$ is a word in $(Q \times \mathbb{N})^*$. A run $(q_0,i_0),\dots,(q_m,i_m)$ is a *run of* $\mathfrak{A}$ *on a word* $w = a_0 \dots a_n \in \Sigma^*$ if $q_0 \in Q_0$, $i_0 = 0$ and there exist $d_0,\dots,d_{m-1} \in \{-1,0,1\}$ such that $q_j \to_{a_j,d_j} q_{j+1}$ and $i_{j+1} = i_j + d_j$ for all j, $0 \le j < m$. The run is *accepting* if $q_m \in F$, $i_m = n+1$. $\mathfrak{A}$ *accepts* $w \in \Sigma^*$ if there is an accepting run of $\mathfrak{A}$ on w; the language $\boldsymbol{L}(\mathfrak{A})$ of $\mathfrak{A}$ is the set of all words accepted by $\mathfrak{A}$.

Given an NFA $\mathfrak{A}$, states $q,q' \in Q$, and $w = a_0 \dots a_n \in \Sigma^*$, we write $q \to_w q'$ if either $w = \varepsilon$ and $q' = q$ or there is a run of $\mathfrak{A}$ on w that starts with $(q_0,0)$ and ends with $(q',n+1)$. We say that a state $q \in Q$ is *reachable* if $q' \to_w q$, for some $q' \in Q_0$ and $w \in \Sigma^*$. Given a DFA $\mathfrak{A} = (Q, \Sigma, \delta, q_0, F)$, for any word $w \in \Sigma^*$, we define a function $\delta_w\colon Q \to Q$ by taking $\delta_w(q) = q'$ iff $q \to_w q'$.

A language $\boldsymbol{L}$ over an alphabet Σ is $\mathcal{L}$-*definable* if there is an $\mathcal{L}$-sentence φ in the signature Σ, whose symbols are treated as unary predicates, such that, for any $w \in \Sigma^*$, we have $w = a_0 \dots a_n \in \boldsymbol{L}$ iff $\mathfrak{S}_w \models \varphi$, where $\mathfrak{S}_w$ is a structure with domain $\{0,\dots,n\}$, in which $\mathfrak{S}_w \models a(i)$ iff $a = a_i$, for $i \le n$.

For any OMQ $\boldsymbol{q}$ and $\Xi \subseteq \mathsf{sig}(\boldsymbol{q})$, we regard $\Sigma_\Xi = 2^\Xi$ as an *alphabet*. Any Ξ-ABox $\mathcal{A}$ can be given as a Σ_Ξ-word $w_\mathcal{A} = a_0 \dots a_n$ with $a_i = \{A \mid A(i) \in \mathcal{A}\}$. Conversely, any Σ_Ξ-word $w = a_0 \dots a_n$ gives the ABox $\mathcal{A}_w$ with $\mathsf{tem}(\mathcal{A}_w) = [0, n]$ and $A(i) \in \mathcal{A}_w$ iff $A \in a_i$. The word $\emptyset$ corresponds to $\mathcal{A}_\emptyset = \emptyset$ with $\mathsf{tem}(\mathcal{A}_\emptyset) = [0, 0]$. The *language* $\boldsymbol{L}_\Xi(\boldsymbol{q})$ is defined to be the set of Σ_Ξ-words $w_\mathcal{A}$ with a yes-answer to $\boldsymbol{q}$ over $\mathcal{A}$.

▶ **Proposition 5.** *The language $\boldsymbol{L}_\Xi(\boldsymbol{q})$ is regular. For $\mathcal{L} \in \{\mathsf{FO}(<), \mathsf{FO}(<, \equiv), \mathsf{FO}(<, \mathsf{MOD})\}$, the OMQ $\boldsymbol{q}$ is $\mathcal{L}$-rewritable over Ξ-ABoxes iff $\boldsymbol{L}_\Xi(\boldsymbol{q})$ is $\mathcal{L}$-definable.*

Proof. Let $\mathsf{sub}_{\boldsymbol{q}}$ be the set of temporal concepts in $\boldsymbol{q}$ and their negations. A *type* is any maximal subset $\tau \subseteq \mathsf{sub}_{\boldsymbol{q}}$ consistent with $\mathcal{O}$. Let $\boldsymbol{T}$ be the set of all types. Define an NFA $\mathfrak{A}$ over Σ_Ξ with $\boldsymbol{L}(\mathfrak{A}) = \Sigma_\Xi^* \setminus \boldsymbol{L}_\Xi(\boldsymbol{q})$. Its states are $Q_{\neg\varkappa} = \{\tau \in \boldsymbol{T} \mid \neg\varkappa \in \tau\}$. The transition relation $\to_a$, for $a \in \Sigma_\Xi$, is defined by taking $\tau_1 \to_a \tau_2$ if the following conditions hold: (a) $a \subseteq \tau_2$, (b) $\bigcirc_{\!\scriptscriptstyle F} C \in \tau_1$ iff $C \in \tau_2$, (c) $\Box_{\scriptscriptstyle F} C \in \tau_1$ iff $C \in \tau_2$ and $\Box_{\scriptscriptstyle F} C \in \tau_2$, (d) $\Diamond_{\scriptscriptstyle F} C \in \tau_1$ iff $C \in \tau_2$ or $\Diamond_{\scriptscriptstyle F} C \in \tau_2$, and symmetrically for $\bigcirc_{\!\scriptscriptstyle P}, \Box_{\scriptscriptstyle P}, \Diamond_{\scriptscriptstyle P}$. The initial (accepting) states are those $\tau \in Q_{\neg\varkappa}$, for which $\tau \cup \{\Box_{\scriptscriptstyle P} \neg\varkappa\}$ (respectively, $\tau \cup \{\Box_{\scriptscriptstyle F} \neg\varkappa\}$) is consistent with $\mathcal{O}$. Then $w \in \boldsymbol{L}(\mathfrak{A})$ iff $(\mathcal{O}, \mathcal{A}_w) \not\models \exists x\, \varkappa(x)$, for any $w \in \Sigma_\Xi^*$. The number of states in $\mathfrak{A}$ is $2^{O(|\boldsymbol{q}|)}$ and $\mathfrak{A}$ can be constructed using space polynomial in $|\boldsymbol{q}|$ as *LTL*-satisfiability is in PSPACE. ◀

Thus, we can reformulate the evaluation problem for an *LTL* OMQ $\boldsymbol{q}$ over Ξ-ABoxes as the *word problem* for the regular language $\boldsymbol{L}_\Xi(\boldsymbol{q})$.

3 Deciding FO-rewritability of *LTL* OMQs

In this section, we establish the complexity of recognising the rewritability type of an arbitrary $LTL_{bool}^{\Box\bigcirc}$ OMQ.

▶ **Theorem 6.** *For any $\mathcal{L} \in \{\mathsf{FO}(<), \mathsf{FO}(<, \equiv), \mathsf{FO}(<, \mathsf{MOD})\}$, deciding $\mathcal{L}$-rewritability of $LTL_{bool}^{\Box\bigcirc}$ OMQs over Ξ-ABoxes is EXPSPACE-complete; the lower bound holds already for $LTL_{horn}^{\bigcirc}$ OMAQs.*

Proof. The upper bound follows from Proposition 5 and the fact that $\mathcal{L}$-definability of the language of an NFA can be checked in polynomial space [33]. Here, we sketch the proof of the matching lower bound for $LTL_{horn}^{\bigcirc}$ OMAQs, which is inspired by the reductions used for the PSPACE-hardness proofs of $\mathcal{L}$-definability of DFA languages in [18] and [33, Theorem 2].

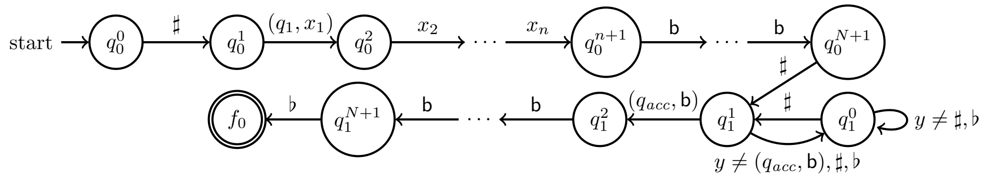
The structure of the proof is as follows: given a Turing machine $\boldsymbol{M}$ that decides a language using at most $N = \exp(n)$ tape cells on any input of size n, for some exponential function exp, we construct (following [33]) automata $\mathfrak{A}_<$, $\mathfrak{A}_\equiv$, and $\mathfrak{A}_{\mathsf{MOD}}$ of size polynomial in N whose languages $\boldsymbol{L}(\mathfrak{A}_<)$, $\boldsymbol{L}(\mathfrak{A}_\equiv)$, and $\boldsymbol{L}(\mathfrak{A}_{\mathsf{MOD}})$ are, respectively, $\mathsf{FO}(<)$-, $\mathsf{FO}(<, \equiv)$-, and $\mathsf{FO}(<, \mathsf{MOD})$-definable iff $\boldsymbol{M}$ rejects $\boldsymbol{x}$. Then we construct $LTL_{horn}^{\bigcirc}$ OMAQs $(\mathcal{O}_<, F)$, $(\mathcal{O}_\equiv, F)$, and $(\mathcal{O}_{\mathsf{MOD}}, F)$ of polynomial size in $|\boldsymbol{x}|$ and $|\boldsymbol{M}|$ that are rewritable into $\mathsf{FO}(<)$, $\mathsf{FO}(<, \equiv)$, and $\mathsf{FO}(<, \mathsf{MOD})$, respectively, iff the corresponding language $\boldsymbol{L}(\mathfrak{A}_\mathcal{L})$ is $\mathcal{L}$-definable.

Suppose $\boldsymbol{M} = (Q, \Sigma, \gamma, \mathsf{b}, q_0, q_{acc})$ with a set Q of states, tape alphabet Σ with b for blank, transition function γ, initial state q_0 and accepting state q_{acc}. Without loss of generality we assume that $\boldsymbol{M}$ erases the tape before accepting and has its head at the left-most cell in an accepting configuration, and if $\boldsymbol{M}$ does not accept the input, it runs forever. Given an input word $\boldsymbol{x} = x_1 \dots x_n$ over Σ, we represent configurations $\mathfrak{c}$ of the computation of $\boldsymbol{M}$ on $\boldsymbol{x}$ by an N-long word written on the tape (with sufficiently many blanks at the end), in which the symbol, y, in the active cell is replaced by the pair (q, y) for the current state q. The accepting computation of $\boldsymbol{M}$ on $\boldsymbol{x}$ is encoded by the word $\sharp\, \mathfrak{c}_1 \,\sharp\, \mathfrak{c}_2 \,\sharp\, \dots \,\sharp\, \mathfrak{c}_{k-1} \,\sharp\, \mathfrak{c}_k \flat$ over the alphabet $\Sigma' = \Sigma \cup (Q \times \Sigma) \cup \{\sharp, \flat\}$, with $\mathfrak{c}_1, \mathfrak{c}_2, \dots, \mathfrak{c}_k$ being the subsequent configurations.

In particular, $\mathfrak{c}_1$ is the initial configuration on $\boldsymbol{x}$ of the form $(q_0, x_1)x_2 \dots x_n \mathsf{b} \dots \mathsf{b}$, and $\mathfrak{c}_k$ is the accepting configuration the form $(q_{acc}, \mathsf{b})\mathsf{b} \dots \mathsf{b}$. As usual for this representation of computations, we may regard γ as a partial function from $\big(\Sigma \cup (Q \times \Sigma)\big)^3$ to $\Sigma \cup (Q \times \Sigma)$.

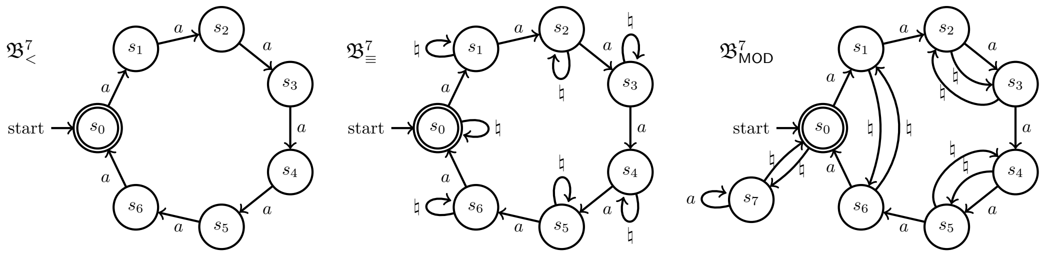
Let p be the first prime such that $p > N + 1$ and $p \not\equiv \pm 1 \pmod{10}$. By [12, Corollary 1.6], p is polynomial in N. Our first aim is to construct a $p+1$-long sequence $\mathfrak{A}_i$ of disjoint DFAs over Σ' such that each $\mathfrak{A}_i$ is of size polynomial in N and $|\boldsymbol{M}|$, it checks certain properties of an accepting computation on $\boldsymbol{x}$, and $\boldsymbol{M}$ accepts $\boldsymbol{x}$ iff the intersection of the $\boldsymbol{L}(\mathfrak{A}_i)$ is not empty and consists of the single word encoding the accepting computation on $\boldsymbol{x}$.

The DFA $\mathfrak{A}_0$ checks whether an input word starts with $\sharp\mathfrak{c}_1$ and ends with $\sharp\mathfrak{c}_k\flat$:



If $1 \le i \le N$, the DFA $\mathfrak{A}_i$ checks, for all j, whether $\gamma(\sigma^j_{i-1}, \sigma^j_i, \sigma^j_{i+1}) = \sigma^{j+1}_i$, where σ^k_l denotes the lth symbol of $\mathfrak{c}_k$. Finally, if $N + 1 \le i \le p$, then $\mathfrak{A}_i$ accepts all words with a single occurrence of $\flat$, which is the input's last character. It is not hard to check that the $\mathfrak{A}_i$ are such that $\boldsymbol{M}$ accepts $\boldsymbol{x}$ iff $\bigcap_{i=0}^p \boldsymbol{L}(\mathfrak{A}_i) \neq \emptyset$, in which case this intersection consists of a single word that encodes the accepting computation of $\boldsymbol{M}$ on $\boldsymbol{x}$.

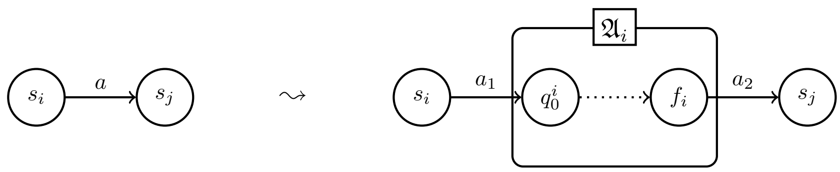
Now we use the $\mathfrak{A}_i$ to define the automata $\mathfrak{A}_<$, $\mathfrak{A}_\equiv$, and $\mathfrak{A}_{\mathsf{MOD}}$. To begin with, we construct DFAs $\mathfrak{B}^p_<$, $\mathfrak{B}^p_\equiv$ and $\mathfrak{B}^p_{\mathsf{MOD}}$, where $p > 5$ is a prime number, following the patterns shown in the picture below for $p = 7$:



In general, $\mathfrak{B}^p_{\mathsf{MOD}} = \big(\{s_i \mid i \le p\}, \{a, \natural\}, \delta^{\mathfrak{B}^p_{\mathsf{MOD}}}, s_0, \{s_0\}\big)$, where

- $\delta_a^{\mathfrak{B}^p_{\mathsf{MOD}}}(s_p) = s_p$, and $\delta_a^{\mathfrak{B}^p_{\mathsf{MOD}}}(s_i) = s_j$ if $i, j < p$ and $j \equiv i + 1 \pmod p$;
- $\delta_\natural^{\mathfrak{B}^p_{\mathsf{MOD}}}(s_0) = s_p$, $\delta_\natural^{\mathfrak{B}^p_{\mathsf{MOD}}}(s_p) = s_0$, and $\delta_\natural^{\mathfrak{B}^p_{\mathsf{MOD}}}(s_i) = s_j$ if $1 \le i, j < p$ and $i \cdot j \equiv p-1 \pmod p$, that is, $j = -1/i$ in the finite field $\mathbb{F}_p$.

Now take some fresh symbols a_1, a_2. We define the automata $\mathfrak{A}_<$, $\mathfrak{A}_\equiv$, $\mathfrak{A}_{\mathsf{MOD}}$ over the same alphabet $\Sigma_+ = \Sigma' \cup \{a_1, a_2, \natural\}$ by taking, respectively, $\mathfrak{B}^p_<$, $\mathfrak{B}^p_\equiv$, $\mathfrak{B}^p_{\mathsf{MOD}}$ and replacing each transition $s_i \to_a s_j$ in them by a fresh copy of $\mathfrak{A}_i$, for $i \le p$, as shown in the picture below, where q^i_0 is the initial state of $\mathfrak{A}_i$:



We make $\mathfrak{A}_<$, $\mathfrak{A}_\equiv$, $\mathfrak{A}_{\mathsf{MOD}}$ deterministic by adding a trash state tr looping on itself with every $y \in \Sigma_+$, and adding the missing transitions leading to tr. It follows that $\mathfrak{A}_<$, $\mathfrak{A}_\equiv$, and $\mathfrak{A}_{\mathsf{MOD}}$ are minimal DFAs of size polynomial in N and $|\boldsymbol{M}|$. Using the algebraic properties of respective syntactic monoids of these languages (see [33, Theorem 1]), one can prove that the languages $\boldsymbol{L}(\mathfrak{A}_<)$, $\boldsymbol{L}(\mathfrak{A}_\equiv)$, and $\boldsymbol{L}(\mathfrak{A}_{\mathsf{MOD}})$ are $\mathcal{L}$-definable for the respective $\mathcal{L}$ iff $\boldsymbol{M}$ rejects $\boldsymbol{x}$.

Now we define $LTL^{\bigcirc}_{horn}$ ontologies $\mathcal{O}_<$, $\mathcal{O}_\equiv$ and $\mathcal{O}_{\mathsf{MOD}}$ simulating $\mathfrak{A}_<$, $\mathfrak{A}_\equiv$ and $\mathfrak{A}_{\mathsf{MOD}}$ such that the size of each ontology is polynomial in $|\boldsymbol{x}|$ and $|\boldsymbol{M}|$.

While $\mathfrak{A}_0$ is of size exponential in n, it has a rather repetitive structure with many transitions of the "same type": $q_0^l \to_{\mathsf{b}} q_0^{l+1}$, for $n < l < N$. We deal with them with the help of counters, where a *counter* is a set $\mathbb{A} = \{A_j^i \mid i = 0, 1,\ j = 1, \dots, k\}$ of atoms for some k logarithmic in N that is used to store values between 0 and $2^k - 1$, which can be different at different time points. We can define Boolean formulas such as $[\mathbb{A} = c]$, $[\mathbb{A} > c]$, $[\mathbb{A} = \mathbb{B} + 1]$ with self-explanatory names that are true iff the value stored in the counters satisfies the corresponding condition. For example, the formula $[\mathbb{A} = \mathbb{B}] = \bigwedge_{j=1}^{k} \big((B_j^0 \to A_j^0) \land (B_j^1 \to A_j^1)\big)$ is true at a time point $m \in \mathbb{Z}$ in an interpretation $\mathcal{I}$ iff the values stored in $\mathbb{A}$ and $\mathbb{B}$ are the same. In particular, we can have counters $\mathbb{A}$ and $\mathbb{L}$ with atomic concepts A_j^i and L_j^i, for $i = 0, 1$, $j = 1, \dots, k$, and then the transitions $q_0^l \to_{\mathsf{b}} q_0^{l+1}$ for $n < l < N$ in $\mathfrak{A}_0$ are captured by the formula

$$[\mathbb{A} = 0] \land Q_0 \land [\mathbb{L} > n] \land [\mathbb{L} < N + 1] \land \mathsf{b} \to [(\bigcirc_{\!F} \mathbb{A}) = 0] \land \bigcirc_{\!F} Q_0 \land [(\bigcirc_{\!F} \mathbb{L}) = \mathbb{L} + 1],$$

which is equivalent to polynomially-many $LTL^{\bigcirc}_{horn}$ axioms. Using the same idea, we can encode all of the transitions of the automata $\mathfrak{A}_<$ and $\mathfrak{A}_\equiv$ by $LTL^{\bigcirc}_{horn}$ ontologies $\mathcal{O}_<$ and $\mathcal{O}_\equiv$ of size polynomial in n and $\boldsymbol{M}$.

Given a word $w = a_1 \dots a_k$, we denote by $\mathcal{A}_w$ the ABox constructed by taking $\bigcup\{a_j(j)\}$ and adding to it $X(0)$ to mark the beginning of the word, and $Y(k+1)$ to mark the end. We also add to the ontology axioms to ensure that an atomic concept F is entailed by the counters and the end word marker Y when the values of the counters correspond to the accepting state of $\mathfrak{A}_{\mathcal{L}}$. This way we have that $\mathfrak{A}_<$ accepts w iff $(\mathcal{O}_<, \mathcal{A}_w) \models \exists x\, F(x)$. Thus $(\mathcal{O}_<, F)$ is $\mathsf{FO}(<)$-rewritable iff $\boldsymbol{M}$ rejects $\boldsymbol{x}$, as required. $\mathcal{O}_\equiv$ is constructed very similarly (see $\mathfrak{B}^7_<$ vs. $\mathfrak{B}^7_\equiv$) and one can show that $(\mathcal{O}_\equiv, F)$ is $\mathsf{FO}(<, \equiv)$-rewritable iff $\boldsymbol{M}$ rejects $\boldsymbol{x}$.

Defining $\mathcal{O}_{\mathsf{MOD}}$ requires some additional tricks. Most importantly, we need to extend $\mathcal{O}_<$ with axioms for handling $\natural$-transitions between certain states of $\mathfrak{A}_{\mathsf{MOD}}$ as follows:

$$[\mathbb{A} = 0] \land S \land \natural \to [(\bigcirc_{\!F} \mathbb{A}) = p] \land \bigcirc_{\!F} S, \quad [\mathbb{A} = p] \land S \land \natural \to [(\bigcirc_{\!F} \mathbb{A}) = 0] \land S,$$
$$[\mathbb{A} > 0] \land [\mathbb{A} < p] \land S \land \natural \to [(\bigcirc_{\!F} \mathbb{A}) = \mathbb{J}] \land \bigcirc_{\!F} S.$$

Here, $\mathbb{J}$ is a new counter that stores the value $j = -1/i$ in the field $\mathbb{F}_p$, which is required to make sure that, for $i \neq 0, p$, we have $\mathcal{O}_{\mathsf{MOD}} \models [\mathbb{A} = i] \land S \land \natural \to [(\bigcirc_{\!F} \mathbb{A}) = j] \land \bigcirc_{\!F} S$. We achieve this as follows. To compute modular inverses using the standard algorithm [31, Exercise 4.5.2.39], we need to halve the number in a counter (easy), compare two counters (using an additional counter), add and subtract (using extra counters for carries). All of this can be done by means of $O(k)$ counters (a fixed number of counters per $O(k)$ steps of the algorithm) with polynomially-many additional axioms. So we compute j when required and store it in the counter $\mathbb{J}$. ◀

We also observe that $LTL^{\bigcirc}_{horn}$ ontologies can be encoded by positive existential queries mediated by covering axioms available in LTL_{krom}:

▶ **Theorem 7.** *$\mathcal{L}$-rewritability of LTL_{krom} OMPEQs over Ξ-ABoxes is* ExpSpace*-complete.*

Proof. Any $LTL^{\bigcirc}_{horn}$ OMAQ $\boldsymbol{q} = (\mathcal{O}, A)$ can be reduced to an $LTL^{\bigcirc}_{krom}$ OMPEQ $\boldsymbol{q}' = (\mathcal{O}', \varkappa)$. For example, we can encode $\mathcal{O} = \{\bigcirc_P A_1 \land A_2 \to A\}$ by $\varkappa = A \lor \Diamond_P \Diamond_F (\bigcirc_P A_1 \land A_2 \land \bar{A})$, for a fresh atom $\bar{A}$, and $\mathcal{O}' = \{A \land \bar{A} \to \bot, \top \to A \lor \bar{A}\}$. ◀

4 Deciding $\mathcal{L}$-rewritability of linear positive $LTL^{\bigcirc}_{horn}$ OMQs

As well known, deciding FO-rewritability of monadic datalog queries is 2ExpTime-complete [11,21], which goes down to PSpace for the important class of linear monadic queries [21,45]. It is not hard to see that any DFA can be simulated by a linear $LTL^{\bigcirc}_{horn}$ OMAQ, which gives a PSpace lower bound for deciding $\mathcal{L}$-rewritability. Also, recall from [6] that, for any $LTL^{\Box\bigcirc}_{horn}$ ontology $\mathcal{O}$ and ABox $\mathcal{A}$ consistent with $\mathcal{O}$, there is a *canonical model* $\mathcal{C}_{\mathcal{O},\mathcal{A}}$ of $\mathcal{O}$ and $\mathcal{A}$ such that $(\mathcal{O}, \mathcal{A}) \models A(k)$ iff $\mathcal{C}_{\mathcal{O},\mathcal{A}} \models A(k)$, for all $k \in \mathbb{Z}$. Given an interpretation $\mathcal{I}$, an OMQ $\boldsymbol{q}$ and $k \in \mathbb{Z}$, we denote by $\tau_{\mathcal{I}}(k)$ the $\boldsymbol{q}$-type of k in $\mathcal{I}$ (see the proof of Proposition 5).

▶ **Theorem 8.**
(i) *For any $\mathcal{L} \in \{\mathsf{FO}(<), \mathsf{FO}(<,\equiv), \mathsf{FO}(<,\mathsf{MOD})\}$, deciding $\mathcal{L}$-rewritability of linear $LTL^{\bigcirc}_{horn}$ OMAQs over Ξ-ABoxes is PSpace-complete.*
(ii) *For any $\mathcal{L} \in \{\mathsf{FO}(<), \mathsf{FO}(<,\equiv)\}$, deciding $\mathcal{L}$-rewritability of linear $LTL^{\bigcirc}_{horn}$ OMPQs over Ξ-ABoxes is PSpace-complete.*

Proof.
(i) We encode an OMAQ $\boldsymbol{q}$ as a polysize 2NFA $\mathfrak{A}^{\Xi}_{\mathcal{O}}$ over the alphabet 2^{Ξ}, having (among others) states q_L for $L \in idb(\mathcal{O}) \cup \{\bot\}$, with $\boldsymbol{L}_{\Xi}(\boldsymbol{q}) = \{\boldsymbol{a} \in \Sigma^*_{\Xi} \mid \emptyset^N \boldsymbol{a} \emptyset^N \in \boldsymbol{L}(\mathfrak{A}^{\Xi}_{\mathcal{O}})\}$, $idb(\mathcal{O})$ comprising the IDB predicates of $\mathcal{O}$ and $N = poly(|\boldsymbol{q}|)$. To illustrate, the following transitions are in $\mathfrak{A}^{\Xi}_{\mathcal{O}}$ for the axiom $\bigcirc_P^2 A' \land \bigcirc_P A \to B$ with IDB A: $q_A \to_{a,-1} q'$ for any $a \in 2^{\Xi}$, $q' \to_{q,1} q_h$ if $A' \in a$ and $q' \to_{q,1} q''$ otherwise, and $q'' \to_{a,1} q_B$ for any $a \in 2^{\Xi}$, where q_h is a fixed trash state. Then we transform, in PSpace, the 2NFA $\mathfrak{A}^{\Xi}_{\mathcal{O}}$ to a DFA $\mathfrak{A}'$ with $\boldsymbol{L}_{\Xi}(\boldsymbol{q}) = \boldsymbol{L}(\mathfrak{A}')$ in the same way as in [33, Section 5], but with different initial and accepting states, to reflect the fact that accepted words have $\emptyset^N$ as a prefix and suffix.

(ii) The canonical model property of $LTL^{\Box\bigcirc}_{horn}$ allows us to formulate the following criteria in terms of types of the canonical model and ABoxes (cf. [33, Theorem 1 (i), (ii)]):

▶ **Lemma 9.** *An $LTL^{\Box\bigcirc}_{horn}$ OMPQ $\boldsymbol{q} = (\mathcal{O}, \varkappa)$ is not $\mathsf{FO}(<)$-rewritable iff there exist ABoxes $\mathcal{A}$, $\mathcal{B}$, $\mathcal{D}$ and $k \geq 2$ such that the following conditions hold:*
- *$(\mathcal{O}, \mathcal{A}\mathcal{B}^k\mathcal{D})$ is consistent, $\neg\varkappa \in \tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^k\mathcal{D}}}(|\mathcal{A}|-1)$, $\tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^k\mathcal{D}}}(|\mathcal{A}|-1) = \tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^k\mathcal{D}}}(|\mathcal{A}\mathcal{B}^k|-1)$;*
- *either $\varkappa \in \tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^{k+1}\mathcal{D}}}(|\mathcal{A}\mathcal{B}|-1)$ and $\tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^{k+1}\mathcal{D}}}(|\mathcal{A}\mathcal{B}|-1) = \tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^{k+1}\mathcal{D}}}(|\mathcal{A}\mathcal{B}^{k+1}|-1)$ or $(\mathcal{O}, \mathcal{A}\mathcal{B}^{k+1}\mathcal{D})$ is inconsistent.*

Furthermore, $\boldsymbol{q}$ is not $\mathsf{FO}(<,\equiv)$-rewritable iff there also exist ABoxes $\mathcal{U}$ and $\mathcal{W}$ such that $\mathcal{B} = \mathcal{U}\mathcal{W}$, $|\mathcal{W}| = |\mathcal{U}|$ the following conditions hold:
- *$\tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^k\mathcal{D}}}(|\mathcal{A}\mathcal{B}^i|-1) = \tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^k\mathcal{D}}}(|\mathcal{A}\mathcal{B}^i\mathcal{U}|-1)$, for all $i < k$, and*
- *either $(\mathcal{O}, \mathcal{A}\mathcal{B}^{k+1}\mathcal{D})$ is inconsistent or $\tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^{k+1}\mathcal{D}}}(|\mathcal{A}\mathcal{B}^i|-1) = \tau_{\mathcal{C}_{\mathcal{O},\mathcal{A}\mathcal{B}^{k+1}\mathcal{D}}}(|\mathcal{A}\mathcal{B}^i\mathcal{U}|-1)$, for all i, $1 \leq i \leq k$.*

Moreover, if $\mathcal{O}$ is linear*, then $|\mathcal{A}|, |\mathcal{B}|, |\mathcal{D}|, |\mathcal{W}|, |\mathcal{U}|, k = 2^{O(|\boldsymbol{q}|)}$.*

Similarly to the algorithm of [33, Theorem 3], we guess the ABoxes $\mathcal{X}$ required by Lemma 9 in the form of quadruples of binary relations $\mathsf{b}(\mathcal{X})$ on the states of the 2NFA $\mathfrak{A}^{\Xi}_{\mathcal{O}}$, and then prove that checking the conditions of the lemma can be done in PSpace. ◀

We note that it is harder to transform [33, Theorem 1 (iii)] to a PSpace-checkable condition on canonical models and ABoxes. The complexity of $\mathsf{FO}(<,\mathsf{MOD})$-rewritability of linear OMPQs remains open.

5 FO($<$)-rewritability of $LTL^{\bigcirc}_{krom}$ OMAQs and $LTL^{\bigcirc}_{core}$ OMPQs

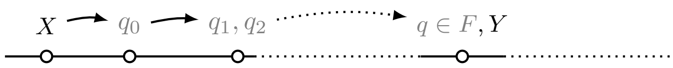
Our next aim is to look for non-trivial OMQ classes deciding FO-rewritability of which could be "easier" than PSPACE. Syntactically, the simplest type of axioms (4) are binary clauses: $C_1 \to C_2$ and $C_1 \wedge C_2 \to \bot$, known as *core* axioms, which together with $C_1 \vee C_2$ form the class Krom. In the atemporal case, the W3C standard language *OWL 2 QL* for ontology-based data access allows core clauses only and uniformly guarantees FO-rewritability [3, 17].

By Theorem 7, OMPEQs with Krom axioms can simulate $LTL^{\bigcirc}_{horn}$ OMAQs, and so are too complex for our aims. On the other hand, $LTL^{\bigcirc}_{krom}$ OMAQs and $LTL^{\bigcirc}_{core}$ OMPQs are all FO($<, \equiv$)-rewritable [6], so we can focus on deciding FO($<$)-rewritability in these classes.

▶ **Theorem 10.** FO($<$)*-rewritability of* $LTL^{\bigcirc}_{krom}$ *OMAQs over* Ξ*-ABoxes is* coNP*-complete.*

Proof. Given $\boldsymbol{q} = (\mathcal{O}, A)$, let $\boldsymbol{q}' = (\mathcal{O}', Y)$ with $\mathcal{O}' = \mathcal{O} \cup \{A \to \bot\}$ and fresh $Y \notin \Xi$. For any Ξ-ABox $\mathcal{A}$, we have $(\mathcal{O}, \mathcal{A}) \models \exists x A(x)$ iff $(\mathcal{O}', \mathcal{A}) \models \exists x\, Y(x)$ iff $(\mathcal{O}', \mathcal{A})$ is inconsistent. One can show, using Kromness, that if $\boldsymbol{L}_{BC} = \{\emptyset^n \mid \mathcal{O} \models B \to \bigcirc_F^{n+1} \neg C\}$ is FO($<$)-definable for all $B, C \in \Xi$, then so is $\boldsymbol{L}_\Xi(\boldsymbol{q}')$, and the OMAQ $\boldsymbol{q}$ is therefore FO($<$)-rewritable. We can construct a unary NFA accepting $\boldsymbol{L}_{BC}$ in polynomial time [6]. It is also readily seen that a unary language is FO($<$)-definable iff it is finite or cofinite. Therefore, deciding FO($<$)-definability of a unary NFA is coNP-complete (using [42, Theorem 6.1]) and FO($<$)-rewritability of an $LTL^{\bigcirc}_{krom}$ OMAQ is in coNP.

To show coNP-hardness, given a unary NFA $\mathfrak{A} = (Q, \{a\}, \delta, q_0, F)$, we define an $LTL^{\bigcirc}_{core}$ ontology $\mathcal{O}_{\mathfrak{A}}$ with the axioms $X \to \bigcirc_F q_0$, $p \to \bigcirc_F q$ for $(p, a, q) \in \delta$, and $Y \wedge q \to \bot$ for $q \in F$:



For a $\{X, Y\}$-ABox $\mathcal{A}$ we have $\mathcal{O}, \mathcal{A} \models \exists x A(x)$ iff there are $m, n \in \mathbb{Z}$ such that $X(n) \in \mathcal{A}$, $Y(m) \in \mathcal{A}$, and $a^{m-n-1} \in \boldsymbol{L}(\mathfrak{A})$. Therefore, the OMAQ $(\mathcal{O}_{\mathfrak{A}}, A)$ is FO($<$)-rewritable over $\{X, Y\}$-ABoxes iff $\boldsymbol{L}(\mathfrak{A})$ is FO($<$)-definable. ◀

In our next result, the ontology language is weaker (core, which is contained in both Krom and Horn), but the queries are more expressive.

▶ **Theorem 11.** FO($<$)*-rewritability of* $LTL^{\bigcirc}_{core}$ *OMPEQs* $\boldsymbol{q} = (\mathcal{O}, \varkappa)$ *over* Ξ*-ABoxes is* Π^p_2*-complete.*

Proof. Let $\mathcal{B} = \{w_1 \dots w_k \in \Sigma_\Xi^* \mid \forall i\, |w(i)| > 0,\ \sum_i |w(i)| \le |\varkappa|\}$. For $w \in \mathcal{B}$, consider the language $\boldsymbol{L}_w = \boldsymbol{L}(\emptyset^* w_1 \emptyset^* \dots \emptyset^* w_k \emptyset^*) \cap \boldsymbol{L}_\Xi(\boldsymbol{q})$. For $v, v' \in \Sigma_\Xi^*$, we write $v' \le v$ if $|v| = |v'|$ and $v'_i \subseteq v_i$, for all i.

As $\boldsymbol{q}$ is an $LTL^{\bigcirc}_{core}$ OMPEQ, we can prove by induction on $|\varkappa|$ that $(\mathcal{O}, \mathcal{A}) \models \exists x \varkappa(x)$ iff $(\mathcal{O}, \mathcal{A}') \models \exists x \varkappa(x)$, for some $\mathcal{A}' \subseteq \mathcal{A}$ with $|\mathcal{A}'| \le |\varkappa|$. Therefore, for every $v \in \Sigma_\Xi^*$, we have $v \in \boldsymbol{L}_\Xi(\boldsymbol{q})$ iff there is $v' \le v$ with $v' \in \boldsymbol{L}_w$ for some $w \in \mathcal{B}$. It follows that $\boldsymbol{L}_\Xi(\boldsymbol{q})$ is FO($<$)-definable iff $\boldsymbol{L}_w$ is FO($<$)-definable, for every $w \in \mathcal{B}$.

For $w = w_1 \dots w_k \in \mathcal{B}$ and $I = (i_0, \dots, i_k)$ let $v_{w,I} = \emptyset^{i_0} w_1 \emptyset^{i_1} \dots w_k \emptyset^{i_k}$. If $\boldsymbol{L}_w$ is FO($<$)-rewritable, then for every $j < k$, the set $\{l \mid v_{w,I'} \in \boldsymbol{L}_w, I' = (i_1, \dots, i_{j-1}, l, i_{j+1}, \dots, i_k)\}$ is finite or cofinite. For $c \in \mathbb{N}$ and I, let $I_{c \to j}$ be I with i_j replaced by $\min(c, i_j)$. We can find $c = 2^{O(|\mathcal{O}|)}$ such that $\boldsymbol{L}_w$ is FO($<$)-definable iff, for any $v_{w,I}$ with $\max(I) \le 2c$ and any $j \le |I|$, we have $v_{w,I} \in \boldsymbol{L}_w$ iff $v_{w,I_{c \to j}} \in \boldsymbol{L}_w$.

Now, $\boldsymbol{q}$ is not FO($<$)-rewritable iff there are $w \in \mathcal{B}$, I and j with $\max(I) \le 2c$ and $j < |I|$ such that only one of $v_{w,I}$ and $v_{w,I_{c \to j}}$ is in $\boldsymbol{L}_w$. To check that $v_{w,I} \in \boldsymbol{L}_w$ can be done in NP, so FO($<$)-rewritability of $\boldsymbol{q}$ is in $\text{coNP}^{\text{NP}} = \Pi^p_2$.

The lower bound is established by reduction of $\forall\exists$3CNF. ◀

If we increase the expressive power of $LTL_{core}^{\bigcirc}$ OMPEQs $\boldsymbol{q} = (\mathcal{O}, \varkappa)$ by allowing $\Box$-operators in $\varkappa$, the problem of deciding $\mathsf{FO}(<)$-rewritability becomes more complex:

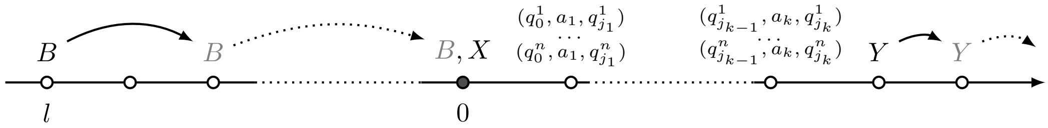
▶ **Theorem 12.** $\mathsf{FO}(<)$*-rewritability of* $LTL_{core}^{\bigcirc}$ *OMPQs over* Ξ*-ABoxes is* PSPACE*-complete.*

Proof. The upper bound is by Theorem 8. The lower one is proved by reduction of the PSPACE-complete DFA intersection problem [32]. Let $\mathfrak{A}_i = (Q_i, \Sigma, \delta_i, q_0^i, F_i)$, $i \leq n$, be DFAs that do not accept ε and have disjoint $Q_i = \{q_j^i\}$. We let $\Xi = \{X, Y, B\} \cup \bigcup_{i \leq n} \delta_i$ and $\varkappa = B \wedge X \wedge \Box_F\big(\big(\bigwedge_{i \leq n} \bigvee_{(q_k^i, a, q_l^i) \in \delta_i} (q_k^i, a, q_l^i)\big) \vee Y\big)$. The ontology $\mathcal{O}$ contains the following axioms: $B \to \bigcirc_F \bigcirc_F B$, $Y \to \bigcirc_F Y$, $X \wedge \bigcirc_F Y \to \bot$, $X \wedge \bigcirc_F (q_k^i, a, q_l^i) \to \bot$ for $k \neq 0$, $\bigcirc_F Y \wedge (q_k^i, a, q_l^i) \to \bot$ for $q_l^i \notin F_i$, $(q_k^i, a, q_l^i) \wedge (q_m^i, b, q_n^i) \to \bot$ for all $k \neq m$ or $l \neq n$, $(q_k^i, a, q_l^i) \wedge \bigcirc_F (q_m^i, b, q_n^i) \to \bot$ for all $l \neq m$, $(q_k^i, a, q_l^i) \wedge (q_m^j, b, q_n^j) \to \bot$ for all $a \neq b$.

Let $\boldsymbol{q} = (\mathcal{O}, \varkappa)$. We prove that $\bigcap_{i \leq n} L(\mathfrak{A}_i) \neq \emptyset$ iff $\boldsymbol{q}$ is not $\mathsf{FO}(<)$-rewritable.

(⇒). Suppose $w = a_1 \dots a_k \in \bigcap_{i \leq n} L(\mathfrak{A}_i)$ and let $(q_0^i, a_1, q_{l_1}^i) \dots (q_{l_{k-1}}^i, a_k, q_{l_k}^i)$ be the run of the ith automaton on w. Let $R_j^i = (q_{l_{j-1}}^i, a_j, q_{l_j}^i)$.

Consider $\mathcal{A}_w = \{X(0)\} \cup \big(\bigcup_{i \in [1,n]} \bigcup_{j \in [1,k]} \{R_j^i(j)\}\big) \cup \{Y(k+1)\}$. The answer to $\boldsymbol{q}$ over $\mathcal{A}_w \cup \{B(l)\}$ is yes iff $l \leq 0$ and even, because only in this case we have $\mathcal{O}, \mathcal{A}_w \cup \{B(l)\} \models B(0)$ and consequently $\mathcal{O}, \mathcal{A}_w \cup \{B(l)\} \models \varkappa(0)$. Since the set $\{l \mid \mathcal{O}, \mathcal{A}_w \cup \{B(l)\} \models \exists x \varkappa(x)\}$ is not $\mathsf{FO}(<)$-definable, the OMQ $\boldsymbol{q}$ is not $\mathsf{FO}(<)$-rewritable. The picture below illustrates the structure of the ABox $\mathcal{A}_w \cup \{B(l)\}$:



(⇐). Suppose $\bigcap_{i \leq n} L(\mathfrak{A}_i) = \emptyset$. Then, for any ABox $\mathcal{A}$ and k, we have $\mathcal{O}, \mathcal{A} \models \varkappa(k)$ iff the ABox $\mathcal{A}$ is inconsistent with $\mathcal{O}$, by the construction of $\boldsymbol{q}$. We can then easily construct the $\mathsf{FO}(<)$-rewriting of $\boldsymbol{q}$ by encoding the inconsistency axioms of $\mathcal{O}$ by $\mathsf{FO}(<)$-formulas. ◀

6 Conclusions

Motivated by ontology-based access to temporal data – a paradigm relying on FO-rewritability of ontology-mediated queries – we considered the problem of determining the optimal rewritability type and data complexity of answering any given *LTL* OMQ. We showed that this problem is closely related to deciding $\mathsf{FO}(<)$-, $\mathsf{FO}(<, \equiv)$- and $\mathsf{FO}(<, \mathsf{MOD})$-definability of regular languages given by DFAs, NFAs and 2NFAs of different size. Based on this correspondence, we showed how the clausal form of ontology axioms in OMQs, the temporal operators involved and the type of queries are reflected in the structure of automata accepting the OMQs' yes-data instances and the complexity of deciding their FO-definability.

Interesting open problems include understanding the impact of the $\Box$-operators in linear and core ontologies on the complexity of deciding FO-rewritability, extending our analysis to *MTL*-ontologies where OMQs are not necessarily $\mathsf{FO}(\mathsf{RPR})$-rewritable, and so are outside of NC^1, and to 2D combinations of *LTL* with description logics, in particular *DL-Lite*.

It would be also interesting to experiment with algorithms for checking $\mathcal{L}$-rewritability of *LTL* OMQs and constructing rewritings into various types of SQL queries. For some $LTL_{bool}^{\Box\bigcirc}$ OMQs and linear $LTL_{horn}^{\bigcirc}$ OMQs, the best target rewriting language is $\mathsf{FO}(<, \mathsf{RPR})$, which can only be captured in SQL with recursion or procedural extensions that are not always supported by RDBMSs and are less efficient. The $\mathsf{FO}(<, \mathsf{MOD})$-rewritable OMQs can be implemented in the most basic SQL using the `count` operator, while $\mathsf{FO}(<, \equiv)$-rewritable ones do not need it.

References

1 S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

2 Foto N. Afrati and Christos H. Papadimitriou. The parallel complexity of simple logic programs. *J. ACM*, 40(4):891–916, 1993. doi:10.1145/153724.153752.

3 A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The DL-Lite family and relations. *Journal of Artificial Intelligence Research (JAIR)*, 36:1–69, 2009.

4 A. Artale, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev. The complexity of clausal fragments of LTL. In *Proc. of the 19th Int. Conf. on Logic for Programming, Artificial Intelligence and Reasoning, LPAR 2013*, volume 8312 of *Lecture Notes in Computer Science*, pages 35–52. Springer, 2013.

5 Alessandro Artale, Roman Kontchakov, Alisa Kovtunova, Vladislav Ryzhikov, Frank Wolter, and Michael Zakharyaschev. Ontology-mediated query answering over temporal data: A survey (invited talk). In Sven Schewe, Thomas Schneider, and Jef Wijsen, editors, *24th International Symposium on Temporal Representation and Reasoning, TIME 2017, October 16-18, 2017, Mons, Belgium*, volume 90 of *LIPIcs*, pages 1:1–1:37. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.TIME.2017.1.

6 Alessandro Artale, Roman Kontchakov, Alisa Kovtunova, Vladislav Ryzhikov, Frank Wolter, and Michael Zakharyaschev. First-order rewritability of ontology-mediated queries in linear temporal logic. *CoRR*, abs/2004.07221, 2020. arXiv:2004.07221.

7 J.-F. Baget, M. Leclère, M.-L. Mugnier, and E. Salvat. On rules with existential variables: Walking the decidability line. *Artificial Intelligence*, 175(9–10):1620–1654, 2011.

8 David A. Mix Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *J. Comput. Syst. Sci.*, 38(1):150–164, 1989.

9 David A. Mix Barrington, Kevin J. Compton, Howard Straubing, and Denis Thérien. Regular languages in NC^1. *J. Comput. Syst. Sci.*, 44(3):478–499, 1992. doi:10.1016/0022-0000(92)90014-A.

10 David A. Mix Barrington and Denis Thérien. Finite monoids and the fine structure of NC^1. *J. ACM*, 35(4):941–952, 1988. doi:10.1145/48014.63138.

11 Michael Benedikt, Balder ten Cate, Thomas Colcombet, and Michael Vanden Boom. 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*, pages 293–304. IEEE Computer Society, 2015. doi:10.1109/LICS.2015.36.

12 M. Bennett, G. Martin, K. O'Bryant, and A. Rechnitzer. Explicit bounds for primes in arithmetic progressions. *Illinois Journal of Mathematics*, 62(1–4):427–532, 2018.

13 L. Bernátsky. Regular expression star-freeness is PSPACE-complete. *Acta Cybern.*, 13(1):1–21, 1997. URL: http://www.inf.u-szeged.hu/actacybernetica/edb/vol13n1/Bernatsky_1997_ActaCybernetica.xml.

14 M. Bienvenu, B. ten Cate, C. Lutz, and F. Wolter. Ontology-based data access: A study through disjunctive datalog, CSP, and MMSNP. *ACM Transactions on Database Systems*, 39(4):33:1–44, 2014.

15 Stefan Borgwardt, Walter Forkel, and Alisa Kovtunova. Finding new diamonds: Temporal minimal-world query answering over sparse aboxes. In Paul Fodor, Marco Montali, Diego Calvanese, and Dumitru Roman, editors, *Rules and Reasoning - Third International Joint Conference, RuleML+RR 2019, Bolzano, Italy, September 16-19, 2019, Proceedings*, volume 11784 of *Lecture Notes in Computer Science*, pages 3–18. Springer, 2019. doi:10.1007/978-3-030-31095-0_1.

16 A. Calì, G. Gottlob, and A. Pieris. Towards more expressive ontology languages: The query answering problem. *Artificial Intelligence*, 193:87–128, 2012. doi:10.1016/j.artint.2012.08.002.

17 D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *Journal of Automated Reasoning*, 39(3):385–429, 2007.

18 Sang Cho and Dung T. Huynh. Finite-automaton aperiodicity is PSPACE-complete. *Theoretical Computer Science*, 88(1):99–116, 1991. URL: `https://core.ac.uk/download/pdf/82662203.pdf`.

19 C. Civili and R. Rosati. A broad class of first-order rewritable tuple-generating dependencies. In *Proc. of the 2nd Int. Datalog 2.0 Workshop*, volume 7494 of *Lecture Notes in Computer Science*, pages 68–80. Springer, 2012.

20 Kevin J. Compton and Claude Laflamme. An algebra and a logic for NC^1. *Inf. Comput.*, 87(1/2):240–262, 1990.

21 Stavros S. Cosmadakis, Haim Gaifman, Paris C. Kanellakis, and Moshe Y. Vardi. Decidable optimization problems for database logic programs (preliminary report). In *STOC*, pages 477–490, 1988. `doi:10.1145/62212.62259`.

22 Stéphane Demri, Valentin Goranko, and Martin Lange. *Temporal Logics in Computer Science*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2016.

23 Volker Diekert and Paul Gastin. First-order definable languages. In Jörg Flum, Erich Grädel, and Thomas Wilke, editors, *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]*, volume 2 of *Texts in Logic and Games*, pages 261–306. Amsterdam University Press, 2008.

24 Cristina Feier, Antti Kuusisto, and Carsten Lutz. Rewritability in monadic disjunctive datalog, MMSNP, and expressive description logics. *Logical Methods in Computer Science*, 15(2), 2019. `doi:10.23638/LMCS-15(2:15)2019`.

25 M. Fisher, C. Dixon, and M. Peim. Clausal temporal resolution. *ACM Trans. Comput. Logic*, 2(1):12–56, 2001.

26 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

27 D. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-Dimensional Modal Logics: Theory and Applications*, volume 148 of *Studies in Logic*. Elsevier, 2003.

28 Olga Gerasimova, Stanislav Kikot, Agi Kurucz, Vladimir V. Podolskii, and Michael Zakharyaschev. A data complexity and rewritability tetrachotomy of ontology-mediated queries with a covering axiom. In Diego Calvanese, Esra Erdem, and Michael Thielscher, editors, *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020*, pages 403–413, 2020. `doi:10.24963/kr.2020/41`.

29 Mark Kaminski, Yavor Nenov, and Bernardo Cuenca Grau. Datalog rewritability of disjunctive datalog programs and non-Horn ontologies. *Artif. Intell.*, 236:90–118, 2016. `doi:10.1016/j.artint.2016.03.006`.

30 Hans W. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, Computer Science Department, University of California at Los Angeles, USA, 1968.

31 Donald Ervin Knuth. *The art of computer programming, Volume II: Seminumerical Algorithms, 3rd Edition*. Addison-Wesley, 1998. URL: `https://www.worldcat.org/oclc/312898417`.

32 Dexter Kozen. Lower bounds for natural proof systems. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pages 254–266, 1977. `doi:10.1109/SFCS.1977.16`.

33 Agi Kurucz, Vladislav Ryzhikov, Yury Savateev, and Michael Zakharyaschev. Deciding FO-rewritability of regular languages, 2021. `arXiv:2105.06202`.

34 L. Libkin. *Elements Of Finite Model Theory*. Springer, 2004.

35 C. Lutz, F. Wolter, and M. Zakharyaschev. Temporal description logics: A survey. In *Proc. of the 15th Int. Symposium on Temporal Representation and Reasoning (TIME 2008)*, pages 3–14, 2008.

36 Carsten Lutz and Leif Sabellek. Ontology-mediated querying with the description logic EL: trichotomy and linear datalog rewritability. In *Proc. of the 26th Int. Joint Conf. on Artificial Intelligence (IJCAI 2017)*, pages 1181–1187, 2017.

37 Jerzy Marcinkowski. 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*, pages 13–24. IEEE Computer Society, 1996. `doi:10.1109/LICS.1996.561299`.

38 Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Linking data to ontologies. *Journal on Data Semantics*, 10:133–173, 2008.

39 Alexander Rabinovich. A proof of Kamp's theorem. *Logical Methods in Computer Science*, 10(1), 2014.

40 Vladislav Ryzhikov, Yury Savateev, and Michael Zakharyaschev. Deciding FO-rewritability of ontology-mediated queries in linear temporal logic, 2021. URL: `https://www.dcs.bbk.ac.uk/~vlad/time21-full.pdf`.

41 Jacques Stern. Complexity of some problems from the theory of automata. *Inf. Control.*, 66(3):163–176, 1985. `doi:10.1016/S0019-9958(85)80058-9`.

42 Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time: Preliminary report. In Alfred V. Aho, Allan Borodin, Robert L. Constable, Robert W. Floyd, Michael A. Harrison, Richard M. Karp, and H. Raymond Strong, editors, *Proceedings of the 5th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1973, Austin, Texas, USA*, pages 1–9. ACM, 1973. `doi:10.1145/800125.804029`.

43 Howard Straubing. *Finite Automata, Formal Logic, and Circuit Complexity*. Birkhauser Verlag, 1994.

44 Jeffrey D. Ullman and Allen Van Gelder. Parallel complexity of logical query programs. *Algorithmica*, 3:5–42, 1988. `doi:10.1007/BF01762108`.

45 Ron van der Meyden. Predicate boundedness of linear monadic datalog is in PSPACE. *Int. J. Found. Comput. Sci.*, 11(4):591–612, 2000. `doi:10.1142/S0129054100000351`.

46 Moshe Y. Vardi. Decidability and undecidability results for boundedness of linear recursive queries. In Chris Edmondson-Yurkanan and Mihalis Yannakakis, editors, *Proceedings of the Seventh ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, March 21-23, 1988, Austin, Texas, USA*, pages 341–351. ACM, 1988. `doi:10.1145/308386.308470`.

47 Moshe Y. Vardi. Automata-theoretic techniques for temporal reasoning. In Patrick Blackburn, J. F. A. K. van Benthem, and Frank Wolter, editors, *Handbook of Modal Logic*, volume 3 of *Studies in logic and practical reasoning*, pages 971–989. North-Holland, 2007. `doi:10.1016/s1570-2464(07)80020-6`.

48 Moshe Y. Vardi and Pierre Wolper. An automata-theoretic approach to automatic program verification (preliminary report). In *Proc. of the Symposium on Logic in Computer Science (LICS'86)*, pages 332–344, 1986.

49 Przemyslaw Andrzej Walega, Bernardo Cuenca Grau, Mark Kaminski, and Egor V. Kostylev. Datalogmtl over the integer timeline. In Diego Calvanese, Esra Erdem, and Michael Thielscher, editors, *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020*, pages 768–777, 2020. `doi:10.24963/kr.2020/79`.

50 Przemyslaw Andrzej Walega, Bernardo Cuenca Grau, Mark Kaminski, and Egor V. Kostylev. Tractable fragments of datalog with metric temporal operators. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 1919–1925. ijcai.org, 2020. `doi:10.24963/ijcai.2020/266`.

51 Guohui Xiao, Diego Calvanese, Roman Kontchakov, Domenico Lembo, Antonella Poggi, Riccardo Rosati, and Michael Zakharyaschev. Ontology-based data access: A survey. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 5511–5519. ijcai.org, 2018. `doi:10.24963/ijcai.2018/777`.

52 Guohui Xiao, Linfang Ding, Benjamin Cogrel, and Diego Calvanese. Virtual knowledge graphs: An overview of systems and use cases. *Data Intell.*, 1(3):201–223, 2019. `doi:10.1162/dint_a_00011`.