Ordered Fragments of First-Order Logic

Reijo Jaakkola
University of Helsinki, Finland
Tampere University, Finland

Abstract

Using a recently introduced algebraic framework for classifying fragments of first-order logic, we study the complexity of the satisfiability problem for several ordered fragments of first-order logic, which are obtained from the ordered logic and the fluted logic by modifying some of their syntactical restrictions.

2012 ACM Subject Classification Theory of computation → Logic

Keywords and phrases ordered logic, fluted logic, complexity, decidability

Digital Object Identifier 10.4230/LIPIcs.MFCS.2021.62

Related Version *Full Version*: `https://arxiv.org/abs/2103.08046`

Funding The research leading to this work was supported by the Academy of Finland grants 324435 and 328987.

Acknowledgements The author wishes to thank Antti Kuusisto for many helpful discussions on fragments of first-order logic. The author also wishes to thank the anonymous reviewers for their valuable suggestions concerning the exposition of the motivation and the context of the present work.

1 Introduction

The study of computational properties of fragments of first-order logic is an active research area, which has been motivated by the general observation that most of the logics used in computer science applications, such as the description logics, can be translated into first-order logic [5]. The main goal of this area is to discover expressive fragments which have nice computational properties; in particular, their satisfiability problem – the problem of determining whether a given sentence of the fragment is satisfiable – should be decidable. Perhaps the most widely studied decidable fragments of first-order logic are the two-variable logic FO^2 and the guarded fragment GF, and their various extensions, see for example [2, 3, 11, 17]. Recently there has been an increasing interest on studying fragments that we refer to in this paper collectively as the *ordered fragments* [1, 12, 13, 14].

Informally speaking, we define a fragment of first-order logic to be ordered, if the syntax of the fragment restricts permutations of variables (with respect to some ordering of the variables) and the order in which the variables are to be quantified. To illustrate these restrictions, consider the sentence

$$\forall v_1(P(v_1) \to \exists v_2(R(v_1, v_2) \wedge \forall v_3 S(v_1, v_2, v_3))).$$

This sentence is ordered in the sense that variables occur in the right order in the atomic formulas, and they are quantified in the correct order. This particular sentence belongs to the most well-known member of this family of logics, namely the so-called fluted logic, which was proved to have a TOWER-complete satisfiability problem in [13].

Another important ordered fragment, which is also relevant for the present work, is the so-called ordered logic, which on the level of sentences is a fragment of fluted logic (for a formal definition of this logic we refer the reader to section 3). In [4] it was proved that the

46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021).
Editors: Filippo Bonchi and Simon J. Puglisi; Article No. 62; pp. 62:1–62:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

complexity of the satisfiability problem of this logic is in PSPACE, by reducing this problem to the satisfiability problem of modal logic over serial frames. It turns out that the satisfiability problem of this fragment is also PSPACE-complete, and the proof for PSPACE-hardness can be found in the full version of this paper.

Thus the aforementioned syntactical restrictions, which guarantee that the formulas of the fragment are ordered, seem to guarantee that the underlying fragments are decidable. Another aspect that makes the ordered fragments of first-order logic interesting is that they are orthogonal in expressive power with respect to other well-known fragments of first-order logic, such as the guarded fragments. For instance, the formula $\forall v_1 \forall v_2 \forall v_3 R(v_1, v_2, v_3)$ is clearly ordered, but it expresses a property that is, for example, neither expressible in GF nor in FO^2. Thus they form a genuinely new family of decidable fragments of first-order logic, and hence they provide us with a fresh perspective on the question of what makes a fragment of first-order logic decidable.

Ordered fragments can also be used to tame the complexity of decidable fragments. To give an example of what we mean by this, we mention the recent work conducted in [1] where the author showed, among other results, that even though the complexity of the satisfiability problem of GF is 2EXPTIME-complete, it becomes EXPTIME-complete if we restrict attention to the set of formulas that also belong to the fluted logic. More precisely the author introduced a new ordered fragment, namely the forward guarded fragment which contains as a proper subset the aforementioned intersection of GF and the fluted logic, and then proceeded to prove that the satisfiability problem of this stronger logic is EXPTIME-complete.

Since the syntax of ordered logics restricts heavily the permutations of variables and the order in which the variables are quantified, their syntax can often be presented naturally in a variable-free way. Indeed, the fluted logic was originally discovered by Quine as a by-product of his attempts to present the full syntax of first-order logic in a variable-free way by using the predicate functor logic [15, 16]. Interestingly, this approach was also adopted in the recent papers [12, 14], where the fluted logic was presented using its variable-free syntax.

Recently a research program was introduced in [6, 7, 10] for classifying fragments of first-order logic within an algebraic framework that is closely related to the aforementioned predicate functor logic. In a nutshell, the basic idea is to identify fragments of first-order logic with finite algebraic signatures (for more details, see the next section). The algebraic framework naturally suggests the idea of defining logics with limited permutations, and hence it is well suited for defining various logics that belong to the family of ordered fragments.

The main purpose of the present work is to apply the aforementioned algebraic framework to study how the complexities of ordered and fluted logic change, if we modify their syntax in various ways. The first question that we study in this paper is whether one could extend the syntax of ordered logic while maintaining the requirement that the complexity of the satisfiability problem remains relatively low. We will formalize different minimal extension of the ordered logic using additional algebraic operators and study the complexities of the resulting logics. The picture that emerges from our results seems to suggest that even if one modifies the syntax of the ordered logic in a very minimal way, the resulting logics will most likely have much higher complexity. For instance, if we relax even slightly the order in which the variables can be quantified, the resulting logic will have NEXPTIME-hard satisfiability problem. However, there are also exceptions to this rule, since the complexity of ordered logic with equality turns out to be the same as the complexity of the regular ordered logic.

Motivated by the recent study of one-dimensional guarded fragments conducted in [8], we will also study the one-dimensional fragments of fluted logic and ordered logic. Intuitively a logic is called one-dimensional if quantification is limited to applications of blocks of existential (universal) quantifiers such that at most one variable remains free in the quantified formula. Imposing the restriction of one-dimensionality to fluted logic and ordered logic decreases quite considerably the complexity of the underlying logics: the complexity of the one-dimensional fluted logic is NExpTime-complete while the complexity of the one-dimensional ordered logic (even with equality) is NP-complete. In the case of one-dimensional fluted logic we are able to add some further algebraic operators into its syntax without increasing its complexity.

We will also prove that several natural extensions of the ordered logic and the fluted logic are undecidable. First, for the ordered logic we are able to show that if we allow variables to be quantified in an arbitrary order, then the resulting logic is undecidable. Secondly, we are able to show that if we lift the restrictions on how the variables in the atomic formulas can be permuted in the one-dimensional fluted logic, then the resulting logic is undecidable. Finally, in the case of the full fluted logic, we can show that if we relax only slightly the way variables can be permuted and the order in which variables can be quantified, then the resulting logic is undecidable.

2 Algebraic way of presenting logics

The purpose of this section is to present the algebraic framework introduced in [6, 7, 10] for defining logics in an algebraic way. We will be working with purely relational vocabularies with no constants and function symbols. In addition we will not consider vocabularies with 0-ary relation symbols. Throughout this paper we will use the convention where the domain of a model $\mathfrak{A}$ will be denoted by the set A.

Let A be an arbitrary set. As usual, a k-**tuple** over A is an element of A^k. We will use ϵ to denote the 0-ary tuple. Given a non-negative integer k, a k-ary **AD-relation** over A is a pair $T = (X, k)$, where $X \subseteq A^k$. Here 'AD' stands for arity-definite. Given a k-ary AD-relation $T = (X, k)$ over A, we will use $(a_1, ..., a_k) \in T$ to denote $(a_1, ..., a_k) \in X$. Given an AD-relation T, we will use $ar(T)$ to denote its arity.

Given a set A, we will use $\mathrm{AD}(A)$ to denote the set of all AD-relations over A. If $T_1, ..., T_k \in \mathrm{AD}(A)$, then the tuple $(A, T_1, ..., T_k)$ will be called an **AD-structure** over A. A bijection $g : A \to B$ is an **isomorphism** between AD-structures $(A, T_1, ..., T_k)$ and $(B, S_1, ..., S_k)$, if for every $1 \leq \ell \leq k$ we have that $ar(T_\ell) = ar(S_\ell)$, and g is an ordinary isomorphism between the relational structures $(A, rel(T_1), ..., rel(T_k))$ and $(B, rel(S_1), ..., rel(S_k))$, where $rel(T)$ denotes the underlying relation of an AD-relation.

The following definition was introduced in [7], where it was called arity-regular relation operator.

▶ **Definition 1.** *A k-ary **relation operator** F is a mapping which associates to each set A a function $F^A : \mathrm{AD}(A)^k \to \mathrm{AD}(A)$ and which satisfies the following requirements.*

1. *The operator F is isomorphism invariant in the sense that whenever two AD-structures $(A, T_1, ..., T_k)$ and $(B, S_1, ..., S_k)$ are isomorphic via g, the same mapping is also an isomorphism between the AD-structures $(A, F^A(T_1, ..., T_k))$ and $(B, F^B(S_1, ..., S_k))$.*
2. *There exists a function $\sharp : \mathbb{N}^k \to \mathbb{N}$ so that for every AD-structure $(A, T_1, ..., T_k)$ we have that the arity of the AD-relation $F^A(T_1, ..., T_k)$ is $\sharp(ar(T_1), ..., ar(T_k))$. In other words the arity of the output AD-relation is always determined fully by the sequence of arities of the input AD-relations.*

Given a set of relation operators $\mathcal{F}$ and a vocabulary τ, we can define a language $\mathrm{GRA}(\mathcal{F})[\tau]$ as follows, where $R \in \tau$ and $F \in \mathcal{F}$:

$$\mathcal{T} ::= \bot \mid \top \mid R \mid \mathrm{F}\underbrace{(\mathcal{T}, ..., \mathcal{T})}_{ar(\mathrm{F})\text{ times}}.$$

Here "GRA" stands for general relational algebra. We sometimes use the infix notation instead of the prefix notation, if the infix notation is more conventional. Furthermore we will drop the brackets in the case where F is unary operator.

If the underlying vocabulary τ is clear from context or irrelevant, we will write $\mathrm{GRA}(\mathcal{F})$ instead of $\mathrm{GRA}(\mathcal{F})[\tau]$. The members of $\mathrm{GRA}(\mathcal{F})$ will be referred to as **terms**. In the case where $\mathcal{F}$ is a finite set $\{F_1, .., F_n\}$, we will use $\mathrm{GRA}(F_1, ..., F_n)$ to denote $\mathrm{GRA}(\{F_1, ..., F_n\})$.

Given a model $\mathfrak{A}$ of vocabulary τ and $\mathcal{T} \in \mathrm{GRA}(\mathcal{F})[\tau]$, we define its **interpretation** $[\![\mathcal{T}]\!]_{\mathfrak{A}}$ recursively as follows:

1. If $\mathcal{T} = \bot$, then we define $[\![\mathcal{T}]\!]_{\mathfrak{A}} := (\varnothing, 0)$, and if $\mathcal{T} = \top$, then we define $[\![\mathcal{T}]\!]_{\mathfrak{A}} := (\{\epsilon\}, 0)$.
2. If $\mathcal{T} = R \in \tau$, then we define $[\![R]\!]_{\mathfrak{A}} = (R^{\mathfrak{A}}, ar(R))$.
3. If $\mathcal{T} = \mathrm{F}(\mathcal{T}_1, ..., \mathcal{T}_k)$, then we define $[\![\mathcal{T}]\!]_{\mathfrak{A}} = \mathrm{F}_A([\![\mathcal{T}_1]\!]_{\mathfrak{A}}, ..., [\![\mathcal{T}_k]\!]_{\mathfrak{A}})$.

Note that the interpretation of a term over $\mathfrak{A}$ is an AD-relation over A. The arity of this AD-relation (over some fixed model) is called the arity of the term $\mathcal{T}$ and we will denote it by $ar(\mathcal{T})$. Note that by definition the arity of the output relation is independent of the underlying model, which guarantees that $ar(\mathcal{T})$ is well-defined.

Given two k-ary terms $\mathcal{T}$ and $\mathcal{P}$ over the same vocabulary, we say that $\mathcal{T}$ is **contained** in $\mathcal{P}$, if for every model $\mathfrak{A}$ over τ and for every $(a_1, ..., a_k) \in A^k$ we have that if $(a_1, ..., a_k) \in [\![\mathcal{T}]\!]_{\mathfrak{A}}$ then $(a_1, ..., a_k) \in [\![\mathcal{P}]\!]_{\mathfrak{A}}$. We will denote this by $\mathcal{T} \models \mathcal{P}$. If $\mathcal{T}$ is a 0-ary term and $\mathfrak{A}$ is a model so that $[\![\mathcal{T}]\!]_{\mathfrak{A}} = (\{\epsilon\}, 0)$, then we denote this by $\mathfrak{A} \models \mathcal{T}$. Given a 0-ary term $\mathcal{T}$, we say that $\mathcal{T}$ is **satisfiable** if there exists a model $\mathfrak{A}$ so that $\mathfrak{A} \models \mathcal{T}$.

We will conclude this section by briefly indicating how we can compare the expressive power of algebras with fragments of FO. Let $k \geq 0$ and consider an FO-formula $\varphi(v_{i_1}, ..., v_{i_k})$, where $(v_{i_1}, ..., v_{i_k})$ lists all the free variables of φ, and $i_1 < ... < i_k$. If $\mathfrak{A}$ is a suitable model, then φ defines the AD-relation $[\![\varphi]\!]_{\mathfrak{A}} := (\{(a_1, ..., a_k) \mid \mathfrak{A} \models \varphi(a_1, ..., a_k)\}, k)$ over A. Given a k-ary term $\mathcal{T}$ and FO-formula $\varphi(v_{i_1}, ..., v_{i_k})$ over the same vocabulary, we say that $\mathcal{T}$ is **equivalent** with φ if for every model $\mathfrak{A}$ we have that $[\![\mathcal{T}]\!]_{\mathfrak{A}} = [\![\varphi]\!]_{\mathfrak{A}}$.

If $\mathcal{F}$ is a set of relation operators and $\mathcal{L} \subseteq \mathrm{FO}$, then we say that $\mathrm{GRA}(\mathcal{F})$ and $\mathcal{L}$ are **equivalent**, if for every $\mathcal{T} \in \mathrm{GRA}(\mathcal{F})$ there exists an equivalent formula $\varphi \in \mathcal{L}$, and conversely for every formula $\varphi \in \mathcal{L}$ there exists an equivalent term $\mathcal{T} \in \mathrm{GRA}(\mathcal{F})$. Similarly, we say that $\mathrm{GRA}(\mathcal{F})$ and $\mathcal{L}$ are **sententially equivalent**, if for every 0-ary term $\mathcal{T} \in \mathrm{GRA}(\mathcal{F})$ there exists an equivalent sentence $\varphi \in \mathcal{L}$, and conversely for every sentence $\varphi \in \mathcal{L}$ there exists an equivalent 0-ary term $\mathcal{T} \in \mathrm{GRA}(\mathcal{F})$.

3 Relevant fragments and complexity results

The purpose of this section is to present the relevant FO-fragments that we are going to study and to present the main complexity results that we are able to obtain. Through out this section (X, k) and (Y, ℓ) are AD-relations over some set A.

We are going to start by defining formally the ordered logic OL, which will form the backbone for the rest of fragments studied in this paper.

▶ **Definition 2.** *Let $\overline{v}_\omega = (v_1, v_2, ...)$ and let τ be a vocabulary. For every $k \in \mathbb{N}$ we define the set $\mathrm{OL}^k[\tau]$ as follows.*

1. *Let $R \in \tau$ be an ℓ-ary relation symbol and consider the prefix $(v_1, ..., v_\ell)$ of $\overline{v}_\omega$ containing precisely ℓ-variables. If $k \geq \ell$, then $R(v_1, ..., v_\ell) \in \mathrm{OL}^k[\tau]$.*
2. *Let $\ell \leq \ell' \leq k$ and suppose that $\varphi \in \mathrm{OL}^\ell[\tau]$ and $\psi \in \mathrm{OL}^{\ell'}[\tau]$. Then $\neg\varphi, (\varphi \wedge \psi) \in \mathrm{OL}^k[\tau]$.*
3. *If $\varphi \in \mathrm{OL}^{k+1}[\tau]$, then $\exists v_{k+1}\varphi \in \mathrm{OL}^k[\tau]$.*

Finally we define $\mathrm{OL}[\tau] := \bigcup_k \mathrm{OL}^k[\tau]$.

▶ Remark 3. The way we have presented the syntax of OL here is slightly different from the way it is often presented in the literature. The two logics are nevertheless equivalent on the level of sentences.

The syntax of this logic is somewhat involved, but it can be given a very nice algebraic characterization using just three relation operators $\{\neg, \cap, \exists\}$, which we are going to define next. Recalling that if F is a relation operator, then F^A denotes the function to which F maps the set A, we can define the relation operators as follows.

$\neg$) We define $\neg^A(X, k) = (A^k \backslash X, k)$. We call $\neg$ the **complementation** operator.

$\cap$) If $k \neq \ell$, then we define $\cap^A((X, k), (Y, \ell)) = (\varnothing, 0)$. Otherwise we define

$$\cap^A((X, k), (Y, \ell)) = (X \cap Y, k).$$

We call $\cap$ the **intersection** operator.

$\exists$) If $k = 0$, then we define $\exists^A(X, k) = (X, k)$. Otherwise we define

$$\exists^A(X, k) = (\{(a_1, ..., a_k) \mid (a_1, ..., a_k, b) \in X, \text{ for some } b \in A\}, k - 1).$$

We call $\exists$ the **projection** operator.

The following proposition establishes the promised characterization result.

▶ **Proposition 4.** OL *and* GRA$(\neg, \cap, \exists)$ *are sententially equiexpressive.*

The complexity of OL is rather low and thus it is natural to ask how it changes if we add additional operators to the syntax of the logic. The first operator that is studied in this paper is the operator E, which we define as follows.

$\boldsymbol{E}$) If $k < 2$, then we define $E^A(X, k) = (X, k)$. Otherwise we define

$$E^A(X, k) = (\{(a_1, ..., a_k) \in X \mid a_{k-1} = a_k\}, k).$$

We call E the **equality** operator.

It turns out that the addition of equality does not increase the complexity of ordered logic. In our proof for the PSPACE upper bound, it will be convenient to extend the ordered logic with an additional operator I, which we define as follows.

$\boldsymbol{I}$) If $k \leq 1$, then we define $I^A(X, k) = (X, k)$, and otherwise we define

$$I^A(X, k) = (\{(a_1, ..., a_{k-1}) \in A^{k-1} \mid (a_1, ..., a_{k-1}, a_{k-1}) \in X\}, k - 1).$$

We call I the substitution operator.

In contrast with the equality operator, adding either of the following two operators to OL will result in a logic with NEXPTIME-hard satisfiability problem.

s) If $k < 2$, then we define $s^A(X, k) = (X, k)$. Otherwise we define

$$s^A(X, k) = (\{(a_1, ..., a_{k-2}, a_k, a_{k-1}) \mid (a_1, ..., a_k) \in X\}, k).$$

We call s the **swap** operator.

C) If $k \neq 1$ and $\ell \leq 1$, then we define $C^A((X,k),(Y,\ell)) = (\varnothing, 0)$. In the case where $1 = k \leq \ell$ (the case $1 = \ell \leq k$ is defined similarly) we will define

$$C^A((Y,\ell),(X,k)) = (\{(a_1,...,a_\ell) \in Y \mid a_\ell \in X\}, \ell).$$

We call C the **one-dimensional intersection**.

The intuition behind the swap operator is clear: it lifts in a minimal way the ordering restriction on the syntax of ordered logic. The one-dimensional intersection may appear to be somewhat unnatural, but the underlying intuition is that we want to lift the uniformity imposed by $\cap$ in a minimal way.

The other ordered fragment investigated in this paper is the fluted logic FL. We will not give a formal definition for this fragment here, but instead we will introduce its algebraic characterization using the operators $\{\neg, \dot\cap, \exists\}$, where $\dot\cap$ is defined as follows.

$\dot\cap$**)** If $m := max\{k,\ell\}$, then we define

$$\dot\cap^A((X,k),(Y,\ell)) = \big(\{(a_1,\dots,a_m) \mid (a_{m-k+1},\dots,a_m) \in X \text{ and } (a_{m-\ell+1},\dots,a_m) \in Y\},\, m\big),$$

We call $\dot\cap$ the **suffix intersection**.

The following result was proved in [7].

▶ **Proposition 5.** FL *and* $\mathrm{GRA}(\neg, \dot\cap, \exists)$ *are equiexpressive.*

It was proved in [13] that the satisfiability problem for FL is Tower-complete. The natural follow-up question is then to study what fragments of FL have more feasible complexity. In this paper we approach this question by studying the so-called *one-dimensional* fragment of fluted logic. To give this logic an algebraic characterization, we will need to introduce two additional operators, $\exists_1$ and $\exists_0$, which we define as follows.

$\exists_1$**)** If $k < 2$, then we define $\exists_1^A(X,k) = (X,k)$. Otherwise we define

$$\exists_1^A(X,k) = (\{a \in A \mid \text{There exists } \bar b \in A^{k-1} \text{ such that } a\bar b \in X\}, 1)$$

$\exists_0$**)** If $k = 0$, then we define $\exists_0^A(X,k) = (X,k)$. Otherwise we define $\exists_0^A(X,k)$ to be $(\{\epsilon\}, 0)$, if X is non-empty, and $(\varnothing, 0)$, if X is empty.

We call collectively the operators $\exists_1$ and $\exists_0$ **one-dimensional projection** operators. These operators correspond to quantification which leaves at most one free-variable free. Now we define the algebra $\mathrm{GRA}(\neg, \dot\cap, \exists_1, \exists_0)$ to be the one-dimensional fluted logic.

As one might expect, imposing the one-dimensionality requirement to formulas of FL will result in a logic with much lower complexity. The exact complexity of one-dimensional FL turns out to be NExpTime-complete, even for its extension with the swap and equality operators $\mathrm{GRA}(s, E, \neg, \dot\cap, \exists_1, \exists_0)$. In this paper we also study the one-dimensional fragment of ordered logic with equality operator $\mathrm{GRA}(E, \neg, \cap, \exists)$, for which the satisfiability problem turns out to be just NP-complete.

Besides just decidability results, we will also prove several undecidability results. To state some of these results, we will first define the following operator p.

p**)** If $k < 2$, then we define $p^A(X,k) = (X,k)$. Otherwise we define

$$p^A(X,k) = (\{(a_1,...,a_k) \mid (a_k, a_1, ..., a_{k-1}) \in X\}, k).$$

We call p the **cyclic permutation** operator.

Table 1 Complexities of the fragments. For each of the fragments the satisfiability problem is complete for the corresponding complexity class listed in the second column, excluding the case $\{s, E, \neg, C, \cap, \exists\}$ where the complexity is not known. All the results listed here are new.

$E, \neg, \cap, \exists_1, \exists_0$	NP
$E, \neg, \cap, \exists$	PSPACE
$s, \neg, C, \cap, \exists$	NEXPTIME
$E, \neg, C, \cap, \exists$	NEXPTIME
$s, E, \neg, C, \cap, \exists$	?
$s, E, \neg, \dot{\cap}, \exists_1, \exists_0$	NEXPTIME
$p, \neg, \cap, \exists$	Π^0_1
$p, \neg, \dot{\cap}, \exists_1, \exists_0$	Π^0_1
$s, \neg, \dot{\cap}, \exists$	Π^0_1

Adding p to an ordered fragments correspondence essentially to the removal of the syntactical restriction that variables should be quantified in a specific order. The following theorem collects our undecidability results.

▶ **Theorem 6.** *Suppose that $\mathcal{F}$ is a set of relation operators that contains $\{p, \neg, \cap, \exists\}$, $\{p, \neg, \dot{\cap}, \exists_1, \exists_0\}$ or $\{s, \neg, \dot{\cap}, \exists\}$. Now the satisfiability problem for $\mathrm{GRA}(\mathcal{F})$ is Π^0_1-hard.*

Let us conclude this section by mentioning briefly two complexity results that follow immediately from the literature and which complement the picture emerging from the results listed in Table 1. First, it is easy to verify that $\mathrm{GRA}(p, s, E, \neg, C, \cap, \exists_1, \exists_0)$ is essentially equivalent with one-dimensional uniform fragment UF_1, which was proved to be NEXPTIME-complete in [9]. The second result that we should mention is that the satisfiability problem for $\mathrm{GRA}(E, \neg, \dot{\cap}, \exists)$ is TOWER-complete, since it contains FL and it can be translated to FL with equality, for which the satisfiability problem was recently proved in [14] to be TOWER-complete.

4 Tables and normal forms

In this paper we are going to perform several model constructions and hence it is useful to start by collecting some definitions and tools that we are going to need in the later sections.

▶ **Definition 7.** *Let $k \in \mathbb{Z}_+$ and $\mathcal{F} \subseteq \{I, s\}$. A k-*table *with respect to $\mathcal{F}$ is a maximally consistent set of k-ary terms of the form $\mathcal{T}$ or $\neg\mathcal{T}$, where $\mathcal{T} \in \mathrm{GRA}(\mathcal{F})$. Given a model $\mathfrak{A}$ and $\overline{a} \in A^k$, we will use $tp_{\mathfrak{A}}(\overline{a})$ to denote the k-table realized by $\overline{a}$.*

We will identify k-tables ρ with the terms $\bigcap_{\alpha \in \rho} \alpha$, which makes sense since all of the algebraic signatures that we are going to consider always include the operator $\cap$. This allows us to use notation such as $\rho \models \rho'$, where ρ and ρ' are k-tables. Furthermore, we will refer to 1-tables also as 1-types. We say that $a \in A$ is **king**, if there is no other element in the model that realizes the same 1-type.

Notice that there is almost no "overlapping" between tables. For instance, if we consider tables for $\varnothing$, then the table realized by a tuple $(a_1, ..., a_k)$ will not imply *anything* about the table realized by any non-identity permutation of the tuple $(a_1, ..., a_k)$ or any sub-tuple of $(a_1, ..., a_k)$. And even if we are considering tables for $\{s\}$, the table realized by $(a_1, ..., a_k)$ will only imply something about the table realized by $(a_1, ..., a_k, a_{k-1})$.

▶ **Definition 8.** *Let $\mathfrak{A}$ and $\mathfrak{B}$ be models over the same vocabulary, and let $\mathcal{F}$ be a subset of $\{I, s, E, \neg, C, \cap, \dot\cap\}$. Let $\overline{a} \in A^k$ and $\overline{b} \in B^k$, where $k \in \mathbb{Z}_+$. We say that $\overline{a}$ and $\overline{b}$ are* similar *with respect $\mathcal{F}$, if for every k-ary term $\mathcal{T} \in \mathrm{GRA}(\mathcal{F})$ we have that $\overline{a} \in [\![\mathcal{T}]\!]_{\mathfrak{A}}$ if, and only if, $\overline{b} \in [\![\mathcal{T}]\!]_{\mathfrak{B}}$.*

In what follows we will not mention the set $\mathcal{F}$ explicitly, since it will always be clear from the context. For different subsets of $\{I, s, E, \neg, C, \cap, \dot\cap\}$ one can find explicit characterizations for when two tuples are similar using the notions of 1-types and tables. For example, if $\mathcal{F} = \{s, C, \neg, \cap\}$, then two tuples $\overline{a}$ and $\overline{b}$ are similar with respect to $\mathcal{F}$ if and only if $tp_{\mathfrak{A}}(\overline{a}) = tp_{\mathfrak{B}}(\overline{b}), tp_{\mathfrak{A}}(a_{k-1}) = tp_{\mathfrak{B}}(b_{k-1})$ and $tp_{\mathfrak{A}}(a_k) = tp_{\mathfrak{B}}(b_k)$.

We will next introduce two Scott-normal forms for our logics. In the normal forms we will use the operator $\cup$ which can be defined in a standard way in terms of $\neg$ and $\cap$.

▶ **Definition 9.** *Let $\mathcal{F} \subseteq \{I, s, E, C\}$.*

- *We say that a term $\mathcal{T} \in \mathrm{GRA}(\mathcal{F} \cup \{\neg, \cap, \exists\})$ is in* normal form*, if it has the following form*

$$\bigcap_{1 \le i \le m^1_\exists} \exists \kappa_i \cap \bigcap_{1 \le j \le m^1_\forall} \forall \lambda_j \cap \bigcap_{1 \le i \le m_\exists} \forall^{n_i}(\neg \alpha^\exists_i \cup \exists \beta^\exists_i) \cap \bigcap_{1 \le j \le m_\forall} \forall^{n_j}(\neg \alpha^\forall_j \cup \forall \beta^\forall_j),$$

  where $\kappa_i, \lambda_j, \alpha^\exists_i, \beta^\exists_i, \alpha_j$ and $\beta^\forall_j$ are terms of $\mathrm{GRA}(\mathcal{F} \cup \{\neg, \cap\})$, and the terms κ_i and λ_j are unary. Here $\forall$ is short-hand notation for $\neg\exists\neg$ and $\forall^n$ stands for a sequence of $\forall$ of length n.
- *We say that a term $\mathcal{T} \in \mathrm{GRA}(\mathcal{F} \cup \{\neg, \cap, \dot\cap, \exists_1, \exists_0\})$ is in* normal form*, if it has the following form*

$$\bigcap_{1 \le i \le m^1_\exists} \exists_0 \kappa_i \cap \bigcap_{1 \le j \le m^1_\forall} \forall_0 \lambda_j \cap \bigcap_{1 \le i \le m_\exists} \forall_0(\neg \alpha^\exists_i \cup \exists_1 \beta^\exists_i) \cap \bigcap_{1 \le j \le m_\forall} \forall_0(\neg \alpha^\forall_j \cup \forall_1 \beta^\forall_j),$$

  where $\kappa_i, \lambda_j, \alpha^\exists_i, \beta^\exists_i, \alpha_j$ and $\beta^\forall_j$ are terms of $\mathrm{GRA}(\mathcal{F} \cup \{\neg, \cap, \dot\cap\})$, and the terms κ_i and λ_j are unary. Here $\forall_0$ and $\forall_1$ are short-hand notations for $\neg\exists_0\neg$ and $\neg\exists_1\neg$ respectively.

In a rather standard fashion one can prove the following lemma.

▶ **Lemma 10.** *Let $\mathcal{F} \subseteq \{I, s, E, C\}$.*

1. *There is a polynomial time nondeterministic procedure, taking as its input a term $\mathcal{T} \in \mathrm{GRA}(\mathcal{F} \cup \{\neg, \cap, \exists\})$ and producing a term $\mathcal{T}'$ in normal form (over extended signature), such that*
   - *if $\mathfrak{A} \models \mathcal{T}$, for some structure $\mathfrak{A}$, then there exists a run of the procedure which produces a term $\mathcal{T}'$ in normal form so that $\mathfrak{A}' \models \mathcal{T}$ for some expansion $\mathfrak{A}'$ of $\mathfrak{A}$.*
   - *if the procedure has a run producing $\mathcal{T}'$ and $\mathfrak{A} \models \mathcal{T}'$, for some $\mathfrak{A}$, then $\mathfrak{A} \models \mathcal{T}$.*
2. *There is a polynomial time nondeterministic procedure, which operates similarly as the above procedure with the exception that it takes as its input a term in $\mathcal{T} \in \mathrm{GRA}(\mathcal{F} \cup \{\neg, \cap, \dot\cap, \exists_1, \exists_0\})$, and which satisfies the additional requirement that if $\mathcal{T}$ does not contain the operator $\dot\cap$, then neither does any of the terms that this procedure produces.*

To conclude this section, we will introduce some further notation and terminology which will be useful in the later sections of this paper. Consider a term $\mathcal{T}$ in normal form. Subterms of $\mathcal{T}$ that are of the form $\forall^{n_i}(\neg \alpha^\exists_i \cup \exists \beta^\exists_i)$ or $\forall_0(\neg \alpha^\exists_i \cup \exists_1 \beta^\exists_i)$ are called **existential requirements** and we will denote them with $\mathcal{T}^\exists_i$. Similarly subterms of the form $\forall^{n_j}(\neg \alpha^\forall_j \cup \forall \beta^\forall_j)$ or of the form $\forall_0(\neg \alpha^\forall_j \cup \forall_1 \beta^\forall_j)$ will be called **universal requirements** and we will denote them with $\mathcal{T}^\forall_j$. Consider a model $\mathfrak{A}$ and an existential requirement $\mathcal{T}^\exists_i$. If

$\mathcal{T}_i^{\exists}$ is of the form $\forall^{n_i}(\neg\alpha_i^{\exists} \cup \exists\beta_i^{\exists})$ and $\overline{a} \in [\![\alpha_i^{\exists}]\!]_{\mathfrak{A}}$, then an element $c \in A$ so that $\overline{a}c \in [\![\beta_i^{\exists}]\!]_{\mathfrak{A}}$ will be called a **witness** for $\overline{a}$ and $\mathcal{T}_i^{\exists}$. Similarly, if $\mathcal{T}_i^{\exists}$ is of the form $\forall_0(\neg\alpha_i^{\exists} \cup \exists_1\beta_i^{\exists})$ and $a \in [\![\alpha_i^{\exists}]\!]_{\mathfrak{A}}$, then a tuple $\overline{c} \in A^k$, where $k = ar(\beta_i^{\exists}) - 1$, is called a witness for a and $\mathcal{T}_i^{\exists}$.

5 Ordered logic with equality

In this section we will study the complexity of $\mathrm{GRA}(E, \neg, \cap, \exists)$, i.e. ordered logic with equality. We will start by proving that this logic has a polynomially bounded model property, which means that each satisfiable term has a model of size at most polynomial with respect to the size of the term.

Before proceeding with the proof, we will first note that w.l.o.g. we can assume that if an element c is a witness for some existential requirement $\mathcal{T}_i^{\exists}$ and a tuple $(a_1, ..., a_k)$, then $a_k \neq c$. This follows from the observation that if $\mathcal{T}_i^{\exists}$ is of the form $\forall^{n_i}(\neg\alpha_i^{\exists} \cup \exists\beta_i^{\exists})$ then it is equivalent with the following term $\forall^{n_i}(\neg(\alpha_i^{\exists} \cap \neg\exists E\beta_i^{\exists}) \cup \exists\beta_i^{\exists})$, where we can replace $\exists E\beta_i^{\exists}$ with $I\beta_i^{\exists}$.

▶ **Theorem 11.** *Let $\mathcal{T} \in \mathrm{GRA}(I, E, \neg, \cap, \exists)$ and suppose that $\mathcal{T}$ is satisfiable. Then $\mathcal{T}$ has a model of size bounded polynomially in $|\mathcal{T}|$.*

Proof. Let $\mathcal{T} \in \mathrm{GRA}(I, E, \neg, \cap, \exists)$ be a term in normal form. Let $\mathfrak{A}$ be a model of $\mathcal{T}$. Without loss of generality we will assume that $\mathfrak{A}$ contains at least two distinct elements. Our goal is to construct a bounded model $\mathfrak{B} \models \mathcal{T}$. As the domain of our model we will take the set

$$B = \{1, ..., m\} \times \{0, 1\},$$

where $m = \max\{m_{\exists}^1, m_{\exists}\}$. To define the model, we just need to specify the tables for all the k-tuples of elements from B. This will be done inductively, and in such a way that the following condition is maintained: for every $\overline{b} \in B^k$ there exists $\overline{a} \in A^k$ so that $\overline{b}$ is similar with $\overline{a}$. Maintaining this requirement will make sure that our model $\mathfrak{B}$ will not violate any universal requirements.

We will start by defining the 1-types for all the elements of B. Since $\mathfrak{A} \models \bigcap_{1 \le i \le m_{\exists}^1} \exists\kappa_i$, for every $1 \le i \le m_{\exists}^1$ there exists $a_i \in A$ so that $tp_{\mathfrak{A}}(a_i) \models \kappa_i$. We will define that for every $(i, j) \in B$, $tp_{\mathfrak{B}}((i, j)) = tp_{\mathfrak{A}}(a_i)$. Suppose then that we have defined the tables for k-tuples and we wish to define the tables for $(k+1)$-tuples. We will start by making sure that all the existential requirements are full-filled. So, let $1 \le i \le m_{\exists}$ and $\overline{b} \in B^k$ so that we have not assigned a witness for $\overline{b}$ and $\mathcal{T}_i^{\exists}$. By construction we know that there exists $\overline{a} \in A^k$ which is similar to $\overline{b}$. Now there exists $a_k \neq c \in A$ so that $\overline{a}c \in [\![\beta_i^{\exists}]\!]$. If $b_k = (i', j)$, then we will use the element $d = (i, j+1 \mod 2)$ as a witness for $\overline{b}$ by defining that $tp_{\mathfrak{B}}(\overline{b}d) = tp_{\mathfrak{A}}(\overline{a}c)$. Since we have reserved for every element $m_{\exists} \le m$ distinct witnesses for the existential requirements, the process of providing witnesses can be done without conflicts.

Having provided witnesses for k-tuples, we will still need to do define the $(k+1)$-tables for the remaining k-tables. So, let $\overline{b} \in B^k$ and $d \in B$ be elements so that the table of $\overline{b}d$ has not been defined. If $b_k = d$, then the table for $\overline{b}d$ is determined by the table for $\overline{b}$. Suppose then that $b_k \neq d$. Let $\overline{a} \in A^k$ be a k-tuple which is similar with $\overline{b}$. Pick an arbitrary $a_k \neq c \in A$ and define $tp_{\mathfrak{B}}(\overline{b}d) = tp_{\mathfrak{A}}(\overline{a}c)$. ◀

The above theorem can be used to show that if we assume that the underlying vocabulary to be bounded, i.e., there is a fixed constant bound on the maximum arity of relation symbols, then the complexity of the ordered logic is NP-complete.

▶ **Theorem 12.** *The satisfiability problem for* $\mathrm{GRA}(E, \neg, \cap, \exists)$ *over bounded vocabularies is NP-complete.*

In the case where the vocabulary is not assumed to be bounded, the complexity of the ordered logic turns out to be PSPACE-complete. The complete proof can be found in the full version of this paper, but we will sketch the basic idea here. The PSPACE-hardness can be proved by reducing the satisfiability problem of modal logic over serial frames to that of $\mathrm{GRA}(\neg, \cap, \exists)$. For the upper bound one can adapt the well-known algorithm of Ladner. The idea is to non-deterministically construct a model in a depth-first fashion by first guessing a set of 1-types of polynomial size (the domain of the model) and then guess tables for longer and longer tuples of elements.

▶ **Theorem 13.** *The satisfiability problem for* $\mathrm{GRA}(E, \neg, \cap, \exists)$ *is PSPACE-complete.*

6 Further extensions of ordered logic

In this section we will study extensions of ordered logic which are obtained by adding either the swap or the one-dimensional intersection (or both) into its syntax. It turns out that we can deduce easily from the literature sharp lower bounds for the relevant fragments.

▶ **Proposition 14.** *Let* $\mathcal{F}$ *be a set of of relation operators that contains either* $\{\neg, C, \cap, \exists\}$ *or* $\{s, \neg, \cap, \exists\}$*. Now the satisfiability problem for* $\mathrm{GRA}(\mathcal{F})$ *is NEXPTIME-hard.*

▶ Remark 15. We remark that the proof of the above proposition shows that the proposition holds even if we restrict attention to vocabularies which contain at most binary relation symbols. In particular, the satisfiability problems of $\mathrm{GRA}(\neg, C, \cap, \exists_1, \exists_0)$ and $\mathrm{GRA}(s, \neg, \cap, \exists_1, \exists_0)$ are also NEXPTIME-hard.

Now we will focus on proving the corresponding upper bounds on the complexities of $\mathrm{GRA}(\neg, C, \cap, \exists)$ and $\mathrm{GRA}(s, \neg, \cap, \exists)$ by showing that their least common extension $\mathrm{GRA}(s, \neg, C, \cap, \exists)$ has the exponentially bounded model property. The core of the argument is the same as the proof of the exponential model property for FO^2 given in [3].

▶ **Theorem 16.** *Let* $\mathcal{T} \in \mathrm{GRA}(s, \neg, C, \cap, \exists)$ *and suppose that* $\mathcal{T}$ *is satisfiable. Then* $\mathcal{T}$ *has a model of size bounded exponentially in* $|\mathcal{T}|$*.*

Proof. Let $\mathcal{T} \in \mathrm{GRA}(s, \neg, C, \cap, \exists)$ be a term in normal form and let $\mathfrak{A}$ be a model of $\mathcal{T}$. Our goal is to construct a bounded model $\mathfrak{B}$ so that $\mathfrak{B} \models \mathcal{T}$. As the domain of the model $\mathfrak{B}$ we will take the set

$$B := \{tp_{\mathfrak{A}}(a) \mid a \in A\} \times \{1, ..., m\} \times \{0, 1, 2\},$$

where $m = \max\{m^1_{\exists}, m_{\exists}\}$. Clearly $|B| \leq 2^{O(|\mathcal{T}|)}$. Again, to construct the model, we will need to specify the tables for all the k-tuples of elements from B. We will follow the same strategy as in the proof of theorem 11, i.e. the tables will be specified inductively while maintaining the condition that for every $\overline{b} \in B^k$ for which $tp_{\mathfrak{B}}(\overline{b})$ has been specified, there exists $\overline{a} \in A^k$ which is similar to $\overline{b}$.

We will start with the 1-types. For every $b = (tp_{\mathfrak{A}}(a), i, j) \in B$ we define that $tp_{\mathfrak{B}}(b) := tp_{\mathfrak{A}}(a)$. Suppose then that we have defined the tables for k-tuples. We start defining the tables for $(k+1)$-tuples by providing witnesses for all the relevant tuples. So, consider an existential requirement $\mathcal{T}_i^{\exists}$ and a tuple $\overline{b} \in B^k$ so that $\overline{b} \in [\![\alpha_i]\!]_{\mathfrak{B}}$. Suppose that $b_k = (tp_{\mathfrak{A}}(a), i', j)$. By construction there exists a tuple $\overline{a} \in A^k$ so that $\overline{b}$ and $\overline{a}$ are similar. Thus $\overline{a} \in [\![\alpha_i]\!]_{\mathfrak{A}}$. Since

$\mathfrak{A} \models \mathcal{T}_i^{\exists}$, there exists an element $c \in A$ which is a witness for $\overline{a}$ and $\mathcal{T}_i^{\exists}$. We will use the element $d = (tp_{\mathfrak{A}}(c), i, j+1 \mod 3) \in B$ as a witness for $\overline{b}$ by defining that $tp_{\mathfrak{B}}(\overline{b}d) := tp_{\mathfrak{A}}(\overline{a}c)$ and $tp_{\mathfrak{B}}((b_1, ..., b_{k-1}, d, b_k)) := tp_{\mathfrak{A}}((a_1, ..., a_{k-1}, c, a_k))$.

Before moving forward, let us argue that our method of assigning witnesses does not produce conflicts. Consider a tuple $\overline{b} = (b_1, ..., b_k) \in B^k$ and $d \in B$ so that we used d as a witness for $\overline{b}$ and some existential requirement $\mathcal{T}_i^{\exists}$. We will argue that the table for the tuple $(\overline{b}, d)$ was not defined in two different ways. First we note that we have reserved distinct elements for each of the existential requirements, and thus we used d as a witness for $\overline{b}$ only for the existential requirement $\mathcal{T}_i^{\exists}$. We then note that since we are assigning witnesses for tuples in a "cyclic" manner, we will not use b_k as a witness for the tuple $(b_1, ..., b_{k-1}, d)$. Since these cases are the only possible ways that we might have defined the table of the tuple $\overline{b}d$ in two different ways, we conclude that it is only defined once.

We will now assign tables for the remaining $(k+1)$-tuples. So, consider a tuple $\overline{b} \in B^k$ and $d = (tp_{\mathfrak{A}}(c), i, j) \in B$ so that we have not defined the table for the tuple $\overline{b}d$. By construction there exists a tuple $\overline{a} \in A^k$ which is similar to $\overline{b}$. Let $c \in A$ be an element that realizes the 1-type of d (and which is not necessarily distinct from a_k). Now we define that $tp_{\mathfrak{B}}(\overline{b}d) := tp_{\mathfrak{A}}(\overline{a}c)$ and that $tp_{\mathfrak{B}}((b_1, ..., b_{k-1}, d, b_k)) = tp_{\mathfrak{A}}((a_1, ..., a_{k-1}, c, a_k))$. ◀

▶ **Corollary 17.** *The satisfiability problem for* $\mathrm{GRA}(s, \neg, C, \cap, \exists)$ *is* NExpTime*-complete.*

We will conclude this section by considering $\mathrm{GRA}(E, \neg, C, \cap, \exists)$ and $\mathrm{GRA}(s, E, \neg, C, \cap, \exists)$. An easy modification in the argument of theorem 11 yields a bounded model property for the first logic.

▶ **Theorem 18.** *Let* $\mathcal{T} \in \mathrm{GRA}(E, \neg, C, \cap, \exists)$ *and suppose that* $\mathcal{T}$ *is satisfiable. Then* $\mathcal{T}$ *has a model of size bounded exponentially in* $|\mathcal{T}|$.

Proof. Let $\mathcal{T}$ be a term in normal form and assume that $\mathfrak{A}$ is a model of $\mathcal{T}$. If $K = \{tp_{\mathfrak{A}}(a) \mid a \text{ is a king}\}$, then one can take as the domain of the bounded model $\mathfrak{B}$ the set

$$B := K \cup (\{tp_{\mathfrak{A}}(a) \mid a \text{ is not a king}\} \times \{1, ..., m\} \times \{0, 1\}),$$

where $m = \max\{m_{\exists}^1, m_{\exists}\}$. One can now adapt the proof of theorem 11 to obtain a model $\mathfrak{B}$ of $\mathcal{T}$ with domain B. ◀

▶ **Corollary 19.** *The satisfiability problem for* $\mathrm{GRA}(E, \neg, C, \cap, \exists)$ *is* NExpTime*-complete.*

The logic $\mathrm{GRA}(s, E, \neg, C, \cap, \exists)$ turns out to be more tricky. We have not been able to verify whether this logic is undecidable, but we can show that it does not have the finite model property, see the full version of this paper.

7 One-dimensional ordered logics

In this section we consider logics that are obtained from the ordered logic and the fluted logic by imposing the restriction of one-dimensionality. We will first show that the satisfiability problem of the one-dimensional fluted logic, which has been extended with the operators s and E, is NExpTime-complete. As usual, we will prove this by showing that the logic has the bounded model property. The proof is heavily influenced by similar model constructions performed in [9] and [8], which were based on the classical construction of [3].

▶ **Theorem 20.** *Let* $\mathcal{T} \in \mathrm{GRA}(s, E, \neg, \dot{\cap}, \exists_1, \exists_0)$ *and suppose that* $\mathcal{T}$ *is satisfiable. Then* $\mathcal{T}$ *has a model of size bounded exponentially in* $|\mathcal{T}|$.

Proof. Let $\mathcal{T} \in \mathrm{GRA}(s, E, \neg, \dot{\cap}, \exists_1, \exists_0)$ be a term in normal form and suppose that $\mathcal{T}$ is satisfiable. Let us fix an arbitrary model $\mathfrak{A}$ of $\mathcal{T}$, which we will use to construct a bounded model $\mathfrak{B}$ for $\mathcal{T}$. Let $K \subseteq A$ denote the set of kings of A. For each existential requirement $\mathcal{T}_i^\exists$ of $\mathcal{T}$ and $k \in [\![\alpha_i^\exists]\!]_{\mathfrak{A}} \cap K$, we will pick some witness $\overline{c}$. Let C denote the resulting set. Next, we let P denote the set of non-royal 1-types realized by elements of $\mathfrak{A}$. Fix some function $f : P \to A$ with the property that $tp_{\mathfrak{A}}(f(\pi)) = \pi$, for every $\pi \in P$. For every existential requirement $\mathcal{T}_i^\exists$ and $\pi \in P$ so that $\pi \models \alpha_i^\exists$, we will pick some witness $\overline{c}^{\pi,i}$. Let $W_{\pi,i}$ denote the set of elements occurring in $\overline{c}^{\pi,i}$ that are not kings.

As the domain of the bounded model $\mathfrak{B}$ we will then take the following set

$$B = C \cup \bigcup_{\pi,i,j} W_{\pi,i,j},$$

where π ranges over P, i ranges over $\{1, ..., m\}$, where $m = \max\{m_\exists^1, m_\exists\}$, and j ranges over $\{0, 1, 2\}$. The sets $W_{\pi,i,j}$ are pairwise disjoint copies of the sets $W_{\pi,i}$. Clearly $|B| \leq 2^{O(|\mathcal{T}|)}$. We will make $\mathfrak{B} \upharpoonright C$ isomorphic with $\mathfrak{A} \upharpoonright C$. Furthermore, we will make each of the structures $\mathfrak{B} \upharpoonright (K \cup W_{\pi,i,j})$ isomorphic with the corresponding structures $\mathfrak{A} \upharpoonright (K \cup W_{\pi,i})$.

We will then provide witnesses for elements of B. Since we have already provided witnesses for kings, we need to only provide witnesses for non-royal elements of the court and for elements in $(B \backslash C)$. We will start with the non-royal elements of the court. Consider an existential requirement $\mathcal{T}_i^\exists$ and let $b \in (C \backslash K) \cap [\![\alpha_i^\exists]\!]_{\mathfrak{A}}$. If there exists a witness for b and $\mathcal{T}_i^\exists$ in C, then nothing needs to be done. So suppose that there does not exists a witness for b and $\mathcal{T}_i^\exists$ in C. If π is the 1-type of b in $\mathfrak{B}$, then we know that there exists a witness $\overline{c}$ for $f(\pi)$ and $\mathcal{T}_i^\exists$. We have now two cases.

Suppose first that the length of $\overline{c}$ is one, i.e. $\overline{c} = c$, for some $c \in A$. If $c = a$, then b is already a witness for itself in $\mathfrak{B}$. If $c \neq a$, then we define $tp_{\mathfrak{B}}(b, d) = tp_{\mathfrak{B}}(a, c)$ and $tp_{\mathfrak{B}}(d, b) = tp_{\mathfrak{B}}(c, a)$, where d denotes the single element of $W_{\pi,i,0}$ (note that d can't be a king, since otherwise b and $\mathcal{T}_i^\exists$ would have had a witness in C). Suppose then that the length of $\overline{c}$ is $k > 1$. If $\overline{d} \in (W_{\pi,i,0} \cup K)^k$ denotes the corresponding witness, then we define $tp_{\mathfrak{B}}(b\overline{d}) = tp_{\mathfrak{A}}(a\overline{c})$ and $tp_{\mathfrak{B}}(bd_1, ..., d_k, d_{k-1}) = tp_{\mathfrak{B}}(ac_1, ..., c_k, c_{k-1})$. Note that since b does not occur in $\overline{d}$ and $\overline{d}$ contains at least one non-royal element, the above definitions do not lead into any conflicts with the structure that we have assigned for $\mathfrak{B} \upharpoonright C$.

Thus we have managed to provide witnesses for elements in $C \backslash K$. To provide witnesses for elements of $(B \backslash C)$, we can do roughly the same as above with the exception that instead of $W_{\pi,i,0}$, we will use - assuming that $b \in W_{\pi',i',j}$ - the set $W_{\pi,i,j+1 \mod 3}$. Let us then briefly argue that the above procedure for producing witnesses can be executed without conflicts. First we note that we do not face any conflicts when assigning witnesses for some b and $\mathcal{T}_i^\exists$ and then for b and $\mathcal{T}_{i'}^\exists$, where $i \neq i'$, since for every j the sets $W_{\pi,i,j}$ and $W_{\pi,i',j}$ are disjoint. Secondly we note that we do not face any conflicts when assigning witnesses for some b and $\mathcal{T}_i^\exists$ and then for $b \neq b'$ and $\mathcal{T}_i^\exists$, since in the first case we assign a table for the tuple $b\overline{d}$ and in the second case for $b'\overline{d}$, and neither of these tables imply anything about the other table. Finally we note that since we are assigning witnesses in a cyclic manner, if we use $\overline{d}$ as a witness for $b \notin C$ and $\mathcal{T}_i^\exists$, then we we are never using any tuple containing b as a witnesses for any of the elements in $\overline{d}$.

To complete the structure, for every k we need to define the tables for tuples $\overline{b} \in B^k$. We can do this inductively with respect to k as follows. Suppose first that there exists distinct elements $b, b' \in B$ so that we have not assigned table for the pair (b, b'). Now we choose a pair of distinct elements $a, a' \in A$ with the same 1-types as b and b', and then define $tp_{\mathfrak{B}}(b, b') = tp_{\mathfrak{A}}(a, a')$ and $tp_{\mathfrak{B}}(b', b) = tp_{\mathfrak{A}}(a', a)$. Note that such elements a, a' exists even if the elements b, b' would have the same 1-types, since at least one of them is not a king.

Suppose then that we have defined the tables for every $\overline{d} \in B^k$. Let $b \in B$ and $\overline{d} \in B^k$ be so that we have not defined the table for the tuple $(b, \overline{d})$. By construction there exists $\overline{c} \in A^k$ which is similar with $\overline{d}$. Let $a \in A$ be an arbitrary element which has the same 1-type as b. We then define $tp_{\mathfrak{B}}(b, \overline{d}) = tp_{\mathfrak{A}}(a, \overline{c})$ and $tp_{\mathfrak{B}}(b, d_1, ..., d_k, d_{k-1}) = tp_{\mathfrak{A}}(a, c_1, ..., c_k, c_{k-1})$. Continuing this way it is clear that we can define tables for all the tuples of B^k in such a way that we do not violate any of the universal requirements. ◀

▶ **Corollary 21.** *The satisfiability problem for* $\mathrm{GRA}(s, E, \neg, \dot{\cap}, \exists_1, \exists_0)$ *is* NEXPTIME*-complete.*

We will conclude this section by considering the one-dimensional ordered logic with equality, which is the logic $\mathrm{GRA}(E, \neg, \cap, \exists_1, \exists_0)$. Perhaps unsurprisingly, the satisfiability problem for this logic is NP-complete.

▶ **Theorem 22.** *The satisfiability problem for* $\mathrm{GRA}(E, \neg, \cap, \exists_1, \exists_0)$ *is NP-complete.*

8 Conclusions

In this paper we have studied systematically how the complexities of various ordered fragments of first-order logic change if we modify slightly the underlying syntax. The general picture that emerges is that even if we relax only slightly the restrictions on the syntax, the complexity of the logic can increase drastically. On the other hand, we have seen that adding the further restriction of one-dimensionality on the logics can greatly decrease the complexity of the logic.

There are several directions in which the work conducted in this paper can be continued. Perhaps the most immediate technical problem is whether the logic $\mathrm{GRA}(s, E, \neg, C, \cap, \exists)$ is decidable. As we have seen, this logic does not have the finite model property, and thus we don't expect that traditional model building techniques can be used to prove that it is decidable. On the other hand, we have not been able to prove that this logic is undecidable using standard tiling arguments.

References

1 Bartosz Bednarczyk. Exploiting forwardness: Satisfiability and query-entailment in forward guarded fragment. In *JELIA*, pages 179–193. Springer, 2021.

2 Erich Grädel. On the restraining power of guards. *The Journal of Symbolic Logic*, 64(4):1719–1742, 1999.

3 Erich Grädel, Phokion G Kolaitis, and Moshe Y Vardi. On the decision problem for two-variable first-order logic. *Bulletin of symbolic logic*, 3(1):53–69, 1997.

4 Andreas Herzig. A new decidable fragment of first order logic. In *3rd Logical Biennial Summer School and Conference in Honour of SC Kleene*, 1990.

5 Ullrich Hustadt, Renate A Schmidt, and Lilia Georgieva. A survey of decidable first-order fragments and description logics. *Journal of Relational Methods in Computer Science*, 1(251-276):3, 2004.

6 Reijo Jaakkola and Antti Kuusisto. Algebraic classifications for fragments of first-order logic and beyond. *arXiv Preprint*, 2020. `arXiv:2005.01184v1`.

7 Reijo Jaakkola and Antti Kuusisto. Algebraic classifications for fragments of first-order logic and beyond. *arXiv Preprint*, 2021. `arXiv:2005.01184v2`.

8 Emanuel Kieronski. One-dimensional guarded fragments. *arXiv Preprint*, 2019. `arXiv:1904.04572`.

9 Emanuel Kieroński and Antti Kuusisto. Complexity and expressivity of uniform one-dimensional fragment with equality. In *International Symposium on Mathematical Foundations of Computer Science*, pages 365–376. Springer, 2014.

10 Antti Kuusisto. On games and computation. *arXiv Preprint*, 2019. `arXiv:1910.14603`.

11 Martin Otto. Two variable first-order logic over ordered domains. *The Journal of Symbolic Logic*, 66(2):685–702, 2001.

12 Ian Pratt-Hartmann. Fluted logic with counting. In *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

13 Ian Pratt-Hartmann, Wiesław Szwast, and Lidia Tendera. The fluted fragment revisited. *The Journal of Symbolic Logic*, 84(3):1020–1048, 2019.

14 Ian Pratt-Hartmann and Lidia Tendera. The fluted fragment with transitive relations. *arXiv Preprint*, 2020. `arXiv:2006.11169`.

15 Willard V Quine. Variables explained away. *Proceedings of the american philosophical society*, 104(3):343–347, 1960.

16 WV Quine. Predicate functors revisited. *The Journal of Symbolic Logic*, 46(3):649–652, 1981.

17 Wieslaw Szwast and Lidia Tendera. On the decision problem for the guarded fragment with transitivity. In *Proceedings 16th Annual IEEE Symposium on Logic in Computer Science*, pages 147–156. IEEE, 2001.