EPiC Series in Computing

Volume 57, 2018, Pages 604–619

LPAR-22. 22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning

The Triguarded Fragment of First-Order Logic

Sebastian Rudolph[1] and Mantas Šimkus[2]

[1] Computational Logic Group, TU Dresden, Germany
sebastian.rudolph@tu-dresden.de
[2] Institute of Logic and Computation, TU Wien, Austria
simkus@dbai.tuwien.ac.at

Abstract

Past research into decidable fragments of first-order logic ($\mathbb{FO}$) has produced two very prominent fragments: the *guarded fragment* $\mathbb{GF}$, and the *two-variable fragment* $\mathbb{FO}^2$. These fragments are of crucial importance because they provide significant insights into decidability and expressiveness of other (computational) logics like *Modal Logics (MLs)* and various *Description Logics (DLs)*, which play a central role in Verification, Knowledge Representation, and other areas. In this paper, we take a closer look at $\mathbb{GF}$ and $\mathbb{FO}^2$, and present a new fragment that subsumes them both. This fragment, called the *triguarded fragment* (denoted $\mathbb{TGF}$), is obtained by relaxing the standard definition of $\mathbb{GF}$: quantification is required to be guarded only for subformulae with *three* or more free variables. We show that, in the absence of equality, satisfiability in $\mathbb{TGF}$ is N2ExpTime-complete, but becomes NExpTime-complete if we bound the arity of predicates by a constant (a natural assumption in the context of MLs and DLs). Finally, we observe that many natural extensions of $\mathbb{TGF}$, including the addition of equality, lead to undecidability.

1 Introduction

Function-free first-order logic (which we will denote by $\mathbb{FO}$ in this paper) plays a central role in Logic and exhibits many favorable properties. However, satisfiability checking of $\mathbb{FO}$ formulae is undecidable, which motivates the search for expressive, yet decidable fragments of $\mathbb{FO}$. Such fragments play a crucial role in the studies of various practically relevant (computational) logics like various Modal Logics (MLs), and the big family of Description Logics (DLs). The latter are logic-based knowledge representation languages, usually suitably limited to ensure the decidability of basic reasoning problems [2, 3, 19]. Many computational and model-theoretic properties of MLs and DLs can be explained by seeing them as fragments of $\mathbb{FO}$. In fact, most MLs and DLs fall into well-known decidable fragments of $\mathbb{FO}$, implying not only decidability, but also complexity results, model-theoretic properties, and limits of expressiveness. For instance, many standard MLs and DLs are subsumed by $\mathbb{FO}^2$, the fragment of $\mathbb{FO}$ with at most two variables [6, 8]. For $\mathbb{FO}^2$ without equality, the satisfiability problem has been known to be decidable for over five decades thanks to Scott [21]. The decidability of satisfiability in $\mathbb{FO}^2$ in the presence of equality is known since 1975 due to Mortimer [16], with the worst-case optimal NExpTime upper bound known since over two decades [13].

Another explanation for the decidability of MLs and DLs is the fact that they can often be translated into the *guarded fragment* $\mathbb{GF}$ of $\mathbb{FO}$ [1] (see also [11] for a discussion). Satisfiability checking in $\mathbb{GF}$ is 2ExpTime-complete in general, but it is ExpTime-complete under the assumption that the arities of predicates are bounded by a constant [12, 23]. The latter is notable as it implies the ExpTime upper bound for consistency checking in many standard DLs, since their translations into $\mathbb{FO}$ use predicate symbols of arity at most two. It has been observed that the above mentioned connection of knowledge representation formalisms to $\mathbb{GF}$ is somewhat more robust than their connection to $\mathbb{FO}^2$, which becomes more clear when looking at reasoning problems beyond consistency checking. Most notably, conjunctive query answering, which is decidable for most DLs, remains decidable for $\mathbb{GF}$, but becomes undecidable for $\mathbb{FO}^2$ [4, 18].

Given the importance of $\mathbb{GF}$ and $\mathbb{FO}^2$, this paper takes a deeper look at them, and studies a new, very expressive fragment of $\mathbb{FO}$ that subsumes both $\mathbb{GF}$ and $\mathbb{FO}^2$. The fragment, called the *triguarded fragment* (denoted $\mathbb{TGF}$), is obtained by relaxing the standard definition of $\mathbb{GF}$. In $\mathbb{GF}$, existential and universal quantification can only be used in (sub)formulae of the form $\exists \boldsymbol{x}.(R(\boldsymbol{t}) \wedge \psi)$ or $\forall \boldsymbol{x}.(R(\boldsymbol{t}) \rightarrow \psi)$, where $R(\boldsymbol{t})$ is an atomic formula such that $\boldsymbol{t}$ contains *all* free variables of ψ (the atom $R(\boldsymbol{t})$ "guards" the formula ψ). In $\mathbb{TGF}$, guardedness of quantification is required only in case ψ has *three* or more free variables (hence the name "triguarded"). This entails that quantification can be used in an unrestricted way for formulae with at most two free variables, and hence $\mathbb{FO}^2$ gets included in $\mathbb{TGF}$ seamlessly.

After providing the definition of $\mathbb{TGF}$, we study the satisfiability problem for $\mathbb{TGF}$ formulae. To this end, we first consider a syntactic variant of the problem: we study satisfiability of $\mathbb{GF}$ formulae in the presence of a built-in predicate U that must be interpreted as the set of all pairs of domain elements. In DL parlance, we consider the extension of $\mathbb{GF}$ with the *universal role*, and thus this fragment is denoted $\mathbb{GFU}$. Since the predicate U can be used to provide "spurious" guards to formulae with up to two free variables, $\mathbb{GFU}$ adds to $\mathbb{GF}$ precisely the expressivity needed to capture $\mathbb{TGF}$. Therefore, in the paper, we mainly focus on $\mathbb{GFU}$ instead of $\mathbb{TGF}$.

We show that in the equality-free case, satisfiability of formulae in $\mathbb{GFU}$ (and thus in $\mathbb{TGF}$) is N2ExpTime-complete. We establish the upper bound using a characterization of the satisfiability of a formula in $\mathbb{GFU}$ via *mosaics*, where a mosaic is a special (finite) collection of *types* that can be used to build a model for the input formula. The upper bound is then established via a procedure that guesses and verifies an appropriate mosaic. The matching lower bound can be obtained by a reduction from the tiling problem for a doubly exponential grid. We then consider the assumption that predicate arities are bounded by a constant. In this case, the mosaic construction gives rise to a NExpTime upper bound for satisfiability of formulae without equality. We note that $\mathbb{FO}^2$ is already NExpTime-hard (even without equality), which means that in the bounded-arity setting, $\mathbb{TGF}$ and $\mathbb{GFU}$ do not have higher complexity than their sublogic $\mathbb{FO}^2$. Subsequently, we show that satisfiability of $\mathbb{TGF}$ and $\mathbb{GFU}$ formulae becomes undecidable in the presence of equality (whereas, interestingly, the complexity of satisfiability in $\mathbb{GF}$ and $\mathbb{FO}^2$ is insensitive to the presence of equality).

The fragment $\mathbb{GFU}$ is similar to the fragment $\mathbb{GF}^{\times_2}$ of [10], which extends $\mathbb{GF}$ with *cross products* (allowing to capture statements like "all elephants are bigger than all mice" as in [20]). The difference is that $\mathbb{GF}^{\times_2}$, inspired by the database view, imposes a separation into a set of ground facts (the data) and a constant-free theory (the schema) [9]. Under this restriction on expressiveness (which is only implicit in [10]), $\mathbb{GF}^{\times_2}$ is in fact subsumed by the fragment $\mathcal{GF}|\mathcal{FO}^2$ from [14]. Using a resolution-based procedure, satisfiability in $\mathcal{GF}|\mathcal{FO}^2$ was shown to be in 2ExpTime, and in NExpTime in case of bounded predicate arities [14]. Instead of resolution, the proof of the 2ExpTime upper bound for $\mathbb{GF}^{\times_2}$ in [10] uses a reduction to satisfiability in plain $\mathbb{GF}$. As we shall see, the unrestricted availability of constants is key in the N2ExpTime-hardness of full $\mathbb{GFU}$ and $\mathbb{TGF}$, and thus is the main distinguishing feature

of the fragments introduced in this paper. We note that the undecidability of $\mathbb{GFU}$ and $\mathbb{TGF}$ in the presence of equality can be inferred from [14] (Section 4.2.3), where a reduction from satisfiability in the *Goldfarb class* is presented, and it can be applied to our fragments. Instead, in this paper we provide a more direct undecidability proof by a reduction from the tiling problem for an infinite grid.

2 Preliminaries

We assume the reader is familiar with the syntax and semantics of $\mathbb{FO}$, and thus we only present some notation. We use $\mathsf{N_P}$, $\mathsf{N_C}$ and $\mathsf{N_V}$ to denote countably infinite, mutually disjoint sets of *predicate symbols*, *constants* and *variables*, respectively. We will mostly use (possibly subscripted) P, R, B and H as predicate symbols. Given a formula φ, we use $\mathsf{N_C}(\varphi)$ and $\mathsf{N_P}(\varphi)$ to denote the set of constants and the set of predicate symbols that appear in φ, respectively. Elements of $\mathsf{N_C} \cup \mathsf{N_V}$ are called *terms*. An *atom* (or, *atomic formula*) is an expression of the form $R(\boldsymbol{t})$, where $\boldsymbol{t}$ is an n-tuple of terms, where n is the *arity* of the predicate symbol $R \in \mathsf{N_P}$. An atom is *ground*, if it has no variables. For convenience, given a tuple $\boldsymbol{t} = \langle t_1, \ldots, t_n \rangle$ of terms, we sometimes view $\boldsymbol{t}$ as the set $\{t_1, \ldots, t_n\}$. Given a tuple $\boldsymbol{x}$ of variables, an $\boldsymbol{x}$-*assignment* is any function $f : \mathsf{N_C} \cup \mathsf{N_V} \to \mathsf{N_C} \cup \mathsf{N_V}$ such that (i) $f(y) \in \mathsf{N_C}$ for all $y \in \boldsymbol{x}$, and (ii) $f(t) = t$ for all $t \notin \boldsymbol{x}$. Given a tuple $\boldsymbol{t} = \langle t_1, \ldots, t_n \rangle$ of terms and an $\boldsymbol{x}$-assignment f, we let $f(\boldsymbol{t}) = \langle f(t_1), \ldots, f(t_n) \rangle$. The semantics to formulae is given using *interpretations*. An interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty set (called *domain*), and $\cdot^{\mathcal{I}}$ is a function that maps (i) every constant $c \in \mathsf{N_C}$ to an element $c^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, and (ii) every predicate symbol $R \in \mathsf{N_P}$ to an n-ary relation over $\Delta^{\mathcal{I}}$, where n is the arity of R. We assume that 0-ary predicate symbols $\top$ and $\bot$ belong to $\mathsf{N_P}$, and they have the usual (built-in) meaning. The equality predicate $\approx$ also belongs to $\mathsf{N_P}$, and has the fixed meaning $\approx^{\mathcal{I}} = \{(e, e) \mid e \in \Delta^{\mathcal{I}}\}$ for all interpretations $\mathcal{I}$. We write $\mathcal{I} \models \varphi$, if an interpretation $\mathcal{I}$ is a model of a *closed* formula (or, a *sentence*) φ. We use $\mathit{free}(\varphi)$ to denote the set of free variables in a formula φ. Given a formula φ, we sometimes write $\varphi[\boldsymbol{x}]$ to indicate that $\boldsymbol{x}$ is an enumeration of $\mathit{free}(\varphi)$.

3 The Triguarded Fragment

We are now ready to introduce the *triguarded fragment* of $\mathbb{FO}$. Essentially, it is a relaxed variant of $\mathbb{GF}$ where guards are only required when quantifying over (sub)formulae with three or more free variables.

Definition 1. *The* triguarded fragment $\mathbb{TGF}$ *of first-order logic is defined as the smallest set of formulae closed under the following rules:*

(1) Every atomic formula belongs to $\mathbb{TGF}$.

(2) $\mathbb{TGF}$ *is closed under the propositional connectives* $\neg$, $\wedge$, $\vee$ *and* $\to$.

(3) If x *is a variable, and* φ *is a formula in* $\mathbb{TGF}$ *with* $|\mathit{free}(\varphi)| \leq 2$, *then* $\exists x.\varphi$ *and* $\forall x.\varphi$ *also belong to* $\mathbb{TGF}$.

(4) If $\boldsymbol{x}$ *is a non-empty tuple of variables,* φ *is a formula in* $\mathbb{TGF}$, α *is an atom, and* $\mathit{free}(\varphi) \subseteq \mathit{free}(\alpha)$, *then* $\exists \boldsymbol{x}.(\alpha \wedge \varphi)$ *and* $\forall \boldsymbol{x}.(\alpha \to \varphi)$ *also belong to* $\mathbb{TGF}$.

Observe that if we consider only the items (1), (2) and (3) in Definition 1 as legal rules to construct formulae, we can build all formulae of $\mathbb{FO}$ that use at most 2 variables, and thus $\mathbb{FO}^2 \subseteq \mathbb{TGF}$. If we consider the items (1), (2) and (4) in Definition 1, we can create all guarded formulae, and thus $\mathbb{GF} \subseteq \mathbb{TGF}$. The syntax of $\mathbb{TGF}$ also allows us to build formulae that are neither in $\mathbb{GF}$ nor in $\mathbb{FO}^2$, witnessed by formulae like

$$\forall x \forall y.((R_1(x,a) \wedge R_2(y,b)) \rightarrow \exists z.R_3(x,y,z,c)). \tag{1}$$

Our main goal in this paper is to understand the computational complexity of satisfiability in $\mathbb{TGF}$. To this end, we concentrate on a slightly different logic, which is effectively equivalent to $\mathbb{TGF}$, but which makes presentation significantly easier. In particular, there is a simple extension of $\mathbb{GF}$ that allows us to capture $\mathbb{TGF}$. Intuitively, $\mathbb{TGF} \not\subseteq \mathbb{GF}$ because $\mathbb{TGF}$ allows "unguarded" quantification in front of formulae φ, but only in case φ has no more than 2 free variables. Now, if our logic provided a "built-in" binary predicate whose extension always contains all pairs of domain elements, we could use it to guard φ. In particular, we consider next the binary *universal role* predicate $\mathsf{U} \in \mathsf{N_P}$, whose extension is fixed to be $\mathsf{U}^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ for all interpretations $\mathcal{I}$. Naturally, we don't allow U to be used in $\mathbb{GF}$ formulae, but note that in $\mathbb{FO}$, $\mathbb{FO}^2$ and $\mathbb{TGF}$, the built-in predicate U does not add expressiveness, because it can be axiomatized using a fresh ordinary binary predicate U and the $\mathbb{FO}^2$ sentence

$$\phi = \forall x \forall y.U(x,y) \tag{2}$$

Thus we can safely allow U to be used as a predicate symbol in formulae of $\mathbb{FO}$, $\mathbb{FO}^2$ and $\mathbb{TGF}$. Since ϕ is not in $\mathbb{GF}$, the addition of the built-in U to $\mathbb{GF}$ makes a big difference (as we shall see from complexity results). We now formally define $\mathbb{GFU}$, which extends $\mathbb{GF}$ with U, and in fact adds to $\mathbb{GF}$ the necessary expressivity to capture $\mathbb{TGF}$.

Definition 2. *Let $\mathbb{GFU}$ be the set of formulae of $\mathbb{TGF}$ that can be built using the items (1), (2) and (4) of Definition 1 only, possibly using the predicate U in atomic formulae.*

By using the U predicate as a guard for formulae with at most 2 free variables, we can convert any $\mathbb{TGF}$ formula into an equivalent formula in $\mathbb{GFU}$. For instance, the above example formula (1) can be transformed into the equivalent $\mathbb{GFU}$ formula

$$\forall x \forall y.(\mathsf{U}(x,y) \rightarrow ((R_1(x,a) \wedge R_2(y,b)) \rightarrow \exists z.R_3(x,y,z,c))). \tag{3}$$

Proposition 3. *For any $\varphi \in \mathbb{TGF}$, we can build in polynomial time an equivalent formula $\varphi' \in \mathbb{GFU}$. Moreover, $\mathsf{N_P}(\varphi') \subseteq \mathsf{N_P}(\varphi) \cup \{\mathsf{U}\}$.*

Proof. (Sketch) We obtain φ' from φ by replacing

- every subformula of the form $\exists x.\varphi[x]$ by $\exists x.(\mathsf{U}(x,x) \wedge \varphi[x])$,
- every subformula of the form $\exists x.\varphi[x,y]$ by $\exists x.(\mathsf{U}(x,y) \wedge \varphi[x,y])$,
- every subformula of the form $\exists xy.\varphi[x,y]$ by $\exists xy.(\mathsf{U}(x,y) \wedge \varphi[x,y])$,
- every subformula of the form $\forall x.\varphi[x]$ by $\forall x.(\mathsf{U}(x,x) \rightarrow \varphi[x])$,
- every subformula of the form $\forall x.\varphi[x,y]$ by $\forall x.(\mathsf{U}(x,y) \rightarrow \varphi[x,y])$,
- every subformula of the form $\forall xy.\varphi[x,y]$ by $\forall xy.(\mathsf{U}(x,y) \rightarrow \varphi[x,y])$.

In other words, we replace every unguarded quantification by a U-guarded one. It is easy to see that this translation does not change the meaning of the formula. □

Note that any $\mathbb{GFU}$ theory can be efficiently translated—while preserving satisfiability—into $\mathbb{TGF}$ by adding the axiomatization of the binary universal predicate (2) above and replacing U by U. Note that the transformations do not just preserve satisfiability but also modelhood (modulo the freshly introduced U). Hence $\mathbb{TGF}$ and $\mathbb{GFU}$ are equally expressive.

4 Characterizing Satisfiability via Mosaics

Thanks to Proposition 3, in order to check satisfiability in $\mathbb{TGF}$, it suffices to consider the satisfiability problem for $\mathbb{GFU}$, and thus in the rest of the paper we focus on $\mathbb{GFU}$. In this section, we study $\mathbb{GFU}$ in the equality-free setting, and provide a finite representation of models of satisfiable $\mathbb{GFU}$ formulae, which will be the basis of the satisfiability checking algorithm. In particular, we show that an equality-free $\mathbb{GFU}$ formula φ has a model iff there exists a *mosaic* for φ, which is a relatively small set of building blocks that can be used to construct a model for φ. In this way, checking satisfiability of φ reduces to verifying the existence of a mosaic for φ.

To simplify the structure of $\mathbb{GFU}$ formulae, we introduce a suitable *(Scott-like) normal form*, which is not much different from the ones used, e.g., in [13, 12].

Definition 4 (Normal Form)**.** *A sentence $\varphi \in \mathbb{GFU}$ is in* normal form *if it has the form $\bigwedge_{\psi\in\mathbf{A}} \psi \wedge \bigwedge_{\psi\in\mathbf{E}} \psi$, where $\mathbf{A}$ contain sentences of the form*

$$\forall \boldsymbol{x}.(R(\boldsymbol{t}) \to (\neg H_1(\boldsymbol{v}_1) \vee \ldots \vee \neg H_n(\boldsymbol{v}_n) \vee H_{n+1}(\boldsymbol{v}_{n+1}) \vee \ldots \vee H_m(\boldsymbol{v}_m))), \tag{4}$$

and $\mathbf{E}$ contain sentences of the form

$$\forall \boldsymbol{x}.(R(\boldsymbol{u}) \to \exists \boldsymbol{y}.H(\boldsymbol{v})). \tag{5}$$

We use $\mathbf{A}(\varphi)$ and $\mathbf{E}(\varphi)$ to denote the sets $\mathbf{A}$ and $\mathbf{E}$ of a formula φ as above. For a sentence $\psi = \forall \boldsymbol{x}.(R(\boldsymbol{u}) \to \exists \boldsymbol{y}.H(\boldsymbol{v}))$, we let $\mathit{width}(\psi)$ denote the number of variables that appear in $\boldsymbol{v}$. For a formula φ as above, $\mathit{width}(\varphi)$ is the maximal $\mathit{width}(\psi)$ over all $\psi \in \mathbf{E}(\varphi)$.

As usual, in case $m = 0$, the empty disjunction in (4) stands for $\bot$. Note that since (4) and (5) are in $\mathbb{GFU}$, each variable that appears in $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m$ also appears in $\boldsymbol{t}$, and each variable that appears in $\boldsymbol{v}$ also appears in $\boldsymbol{u}$. Observe that the sentence in (4) can be equivalently written as

$$\forall \boldsymbol{x}.(R(\boldsymbol{t}) \wedge H_1(\boldsymbol{v}_1) \wedge \ldots \wedge H_n(\boldsymbol{v}_n) \to H_{n+1}(\boldsymbol{v}_{n+1}) \vee \ldots \vee H_m(\boldsymbol{v}_m)). \tag{6}$$

For presentation reasons, in what follows we will mostly use the form (6) instead of (4) when speaking about sentences in $\mathbf{A}$. Note that (6) closely resembles a (guarded) disjunctive Datalog rule with $R(\boldsymbol{t})$ a guard atom.

The following statement ensures that we can focus our attention on formulae in normal form.

Proposition 5. *For any sentence $\varphi \in \mathbb{GFU}$, we can construct in polynomial time a sentence $\varphi' \in \mathbb{GFU}$ in normal form such that (a) φ is satisfiable iff φ' is satisfiable, and (b) the translation does not increase the arity of predicates, i.e., there is no predicate symbol in φ' whose arity is strictly greater than the arity of every predicate in φ.*

In order to prove Proposition 5, we provide the following normalization:

Definition 6. *Let χ be a $\mathbb{GFU}$ sentence. W.l.o.g., we assume χ does not use $\to$ and is in negation normal form. We define the sets $\mathbf{A}_\chi$ and $\mathbf{E}_\chi$ as follows, introducing for certain subformulae $\psi[\boldsymbol{z}]$ with free variables $\boldsymbol{z}$ fresh predicates p_ψ of arity $|\boldsymbol{z}|$.*

- *We let* $\mathbf{A}_\chi$ *contain the sentence*
$$\rightarrow p_\chi \tag{7}$$
- *For every subformula* $\varphi_1[\boldsymbol{x}] \wedge \varphi_2[\boldsymbol{y}]$ *of* χ*, we add to* $\mathbf{A}_\chi$ *the sentences*
$$\forall \boldsymbol{xy}.(p_{\varphi_1 \wedge \varphi_2}(\boldsymbol{x} \cup \boldsymbol{y}) \rightarrow p_{\varphi_1}(\boldsymbol{x})) \tag{8}$$
$$\forall \boldsymbol{xy}.(p_{\varphi_1 \wedge \varphi_2}(\boldsymbol{x} \cup \boldsymbol{y}) \rightarrow p_{\varphi_2}(\boldsymbol{y})) \tag{9}$$
- *For every subformula* $\varphi_1[\boldsymbol{x}] \vee \varphi_2[\boldsymbol{y}]$ *of* χ*, we add to* $\mathbf{A}_\chi$ *the sentence*
$$\forall \boldsymbol{xy}.(p_{\varphi_1 \vee \varphi_2}(\boldsymbol{x} \cup \boldsymbol{y}) \rightarrow p_{\varphi_1}(\boldsymbol{x}) \vee p_{\varphi_2}(\boldsymbol{y})) \tag{10}$$
- *For every subformula of* χ *that is a non-guard atom* $\alpha(\boldsymbol{x})$ *we add to* $\mathbf{A}_\chi$ *the sentence*
$$\forall \boldsymbol{x}.(p_{\alpha(\boldsymbol{x})}(\boldsymbol{x}) \rightarrow \alpha(\boldsymbol{x})) \tag{11}$$
- *For every subformula of* χ *that is a negated non-guard atom* $\neg\alpha(\boldsymbol{x})$ *add the sentence*
$$\forall \boldsymbol{x}.(p_{\neg\alpha(\boldsymbol{x})}(\boldsymbol{x}) \rightarrow \neg\alpha(\boldsymbol{x})) \tag{12}$$
- *For every subformula of* χ *with the shape* $\forall \boldsymbol{x}.\neg\alpha(\boldsymbol{x},\boldsymbol{y}) \vee \varphi(\boldsymbol{x},\boldsymbol{y})$ *we add to* $\mathbf{A}_\chi$ *the sentence*
$$\forall \boldsymbol{xy}.(\alpha(\boldsymbol{x},\boldsymbol{y}) \rightarrow \neg p_{\forall \boldsymbol{x}.\neg\alpha(\boldsymbol{x},\boldsymbol{y}) \vee \varphi[\boldsymbol{x},\boldsymbol{y}]}(\boldsymbol{y}) \vee p_{\varphi[\boldsymbol{x},\boldsymbol{y}]}(\boldsymbol{x},\boldsymbol{y})) \tag{13}$$
- *For every subformula of* χ *with the shape* $\exists \boldsymbol{x}.\varphi[\boldsymbol{x},\boldsymbol{y}]$ *we add to* $\mathbf{E}_\chi$ *the sentence*
$$\forall \boldsymbol{y}.(p_{\exists \boldsymbol{x}.\varphi[\boldsymbol{x},\boldsymbol{y}]}(\boldsymbol{y}) \rightarrow \exists \boldsymbol{x}.p_{\varphi[\boldsymbol{x},\boldsymbol{y}]}(\boldsymbol{x},\boldsymbol{y})) \tag{14}$$

Proof. (Proof sketch for Proposition 5) Given φ, we let $\varphi' = \bigwedge_{\psi \in \mathbf{A}_\varphi} \psi \wedge \bigwedge_{\psi \in \mathbf{E}_\varphi} \psi$. Equisatisfiability of φ and φ' follows from the fact that (i) every model of φ' is a model of φ and (ii) every model of φ can be extended to a model of φ' by choosing the interpretation of each auxiliary predicate such that it coincides with the valid variable assignments of the corresponding subformula.

Preservation of the maximal arity follows from the fact that by definition, for any set of variables that occur freely in some subformula of φ, there must be a joint guard for all of them, hence, for each freshly introduced predicate in φ' we find a guard predicate in φ with the same or higher arity. □

To define mosaics, we need the notion of a *type* for a formula φ. Types will form mosaics, and they can be seen as patterns (interpretations of restricted size) for building models of φ.

Definition 7 (Types)**.** *A* type τ for a formula φ *is any set of ground atoms with predicate symbols from* $\mathsf{N_P}(\varphi)$*. We let* $dom(\tau)$ *denote the set of constants that appear in a type* τ*, and let* $\mathcal{I}(\tau)$ *denote the interpretation such that (i)* $\Delta^{\mathcal{I}(\tau)} = dom(\tau)$*, and (ii)* $P^{\mathcal{I}(\tau)} = \{\boldsymbol{t} \mid P(\boldsymbol{t}) \in \tau\}$ *for all predicate symbols* P*. For a sentence* φ*, we write* $\tau \models \varphi$ *if* $\mathcal{I}(\tau) \models \varphi$*. Given a set of constants* F*, we let* $\tau|_F = \{P(\boldsymbol{t}) \in \tau \mid \boldsymbol{t} \subseteq F\}$*, i.e.,* $\tau|_F$ *is the restriction of* τ *to atoms whose all arguments are included in* F*.*

Of particular interest in our treatment is how a distinguished element of some type "looks like" in terms of the predicates it satisfies and its relationship to constants. This information is captured using the notion of *unary types*, in which we abstract from the concrete target constant by replacing it with a special variable.

Definition 8 (Unary Types)**.** *Assume a formula $\varphi \in \mathbb{GFU}$, and let x^φ be a special variable associated with φ. We let $base(\varphi)$ denote the set of all atoms $P(\boldsymbol{t})$ such that $\boldsymbol{t} \subseteq \mathsf{N_C}(\varphi) \cup \{x^\varphi\}$ and $P \in \mathsf{N_P}(\varphi)$. Any subset $\sigma \subseteq base(\varphi)$ is called a* unary type *for φ. Assume a constant c, and let f be the function such that (i) $f(x^\varphi) = c$, and (ii) $f(d) = d$ for all $d \in \mathsf{N_C}$. For a type τ, we define the unary type $\tau|_c^\varphi = \{R(\boldsymbol{t}) \in base(\varphi) \mid R(f(\boldsymbol{t})) \in \tau\}$.*

We are now ready to define mosaics, which will act as witnesses to satisfiability of $\mathbb{GFU}$ formulae (without equality). Roughly, a mosaic for a formula φ is a pair $(\mathcal{M}, \mathcal{X})$, where $\mathcal{X}$ is a collection of "placeholder" constants, and $\mathcal{M}$ is a set of types for φ. In order to be a proper witness to satisfiability, a mosaic must satisfy a collection of conditions. In particular, they ensure that in case φ is satisfiable, we will be able to construct a model by arranging together (possibly multiple) *instances* of types from $\mathcal{M}$. Intuitively, by an instance of a type $\tau \in \mathcal{M}$ we mean a concrete structure that is obtained by replacing the placeholder constants from $\mathcal{X}$ with concrete domain elements.

Definition 9 (Mosaic)**.** *A* mosaic *for a sentence $\varphi \in \mathbb{GFU}$ in normal form is a pair $(\mathcal{M}, \mathcal{X})$, where $\mathcal{M}$ is a set of types for φ and $\mathcal{X} \subseteq \mathsf{N_C} \setminus \mathsf{N_C}(\varphi)$, satisfying the following:*

(A) $|\mathcal{X}| \leq width(\varphi)$;

(B) *For all* $\tau \in \mathcal{M}$, $dom(\tau) \subseteq \mathsf{N_C}(\varphi) \cup \mathcal{X}$;

(C) *For all* $\tau, \tau' \in \mathcal{M}$, $\tau|_{\mathsf{N_C}(\varphi)} = \tau'|_{\mathsf{N_C}(\varphi)}$;

(D) $\mathsf{U}(t, v) \in \tau$ *for all* $\tau \in \mathcal{M}$ *and each pair* $t, v \in dom(\tau)$;

(E) $\tau \models \psi$ *for all* $\tau \in \mathcal{M}$ *and all* $\psi \in \mathbf{A}(\varphi)$;

(F) *If* $\tau \in \mathcal{M}$, $\forall \boldsymbol{x}.(R(\boldsymbol{t}) \to \exists \boldsymbol{y}.H(\boldsymbol{v})) \in \mathbf{E}(\varphi)$, *and* $R(g(\boldsymbol{t})) \in \tau$ *for some $\boldsymbol{x}$-assigment g, then there is some* $\tau' \in \mathcal{M}$ *such that:*

  (a) $H(h(g(\boldsymbol{v}))) \in \tau'$ *for some $\boldsymbol{y}$-assignment h;*

  (b) $\tau|_F = \tau'|_F$, *where* $F = \mathsf{N_C}(\varphi) \cup \{g(x) \mid x \in \boldsymbol{x} \cap \boldsymbol{v}\}$.

(G) *If* $t_1 \in dom(\tau_1) \cap \mathcal{X}$ *and* $t_2 \in dom(\tau_2) \cap \mathcal{X}$ *for some* $\tau_1, \tau_2 \in \mathcal{M}$, *then there exists a type* $\tau \in \mathcal{M}$ *and a pair* v_1, v_2 *with* $dom(\tau) \cap \mathcal{X} = \{v_1, v_2\}$ *such that (i)* $v_1 \neq v_2$, *(ii)* $\tau_1|_{t_1}^\varphi = \tau|_{v_1}^\varphi$, *(iii)* $\tau_2|_{t_2}^\varphi = \tau|_{v_2}^\varphi$.

Intuitively, the conditions (A-G) ensure the following: (A) requires that only a small number of placeholder constants is used. Due to (B), types in mosaics only refer to original constants of the formula and the small number of place holder constants. The conditions (A) and (B) are important to ensure the relatively small size of mosaics. The condition (C) forces the types to agree on the participation of constants in predicates. (D) requires U to be correctly interpreted locally (i.e., within the individual types), and (E) requires each type to (locally) satisfy all sentences from $\mathbf{A}(\varphi)$. The condition (F) ensures that for each type locally satisfying the body of some sentence from $\mathbf{E}(\varphi)$, we find a matching type where also the head of that sentence is satisfied. Using (G) we make sure that any two representatives of unnamed domain elements (in terms of unary types) found across the types also occur together in one type.

The following soundness and completeness theorems show that mosaics properly characterize satisfiability of equality-free $\mathbb{GFU}$ formulae (and, due to Proposition 3, of equality-free $\mathbb{TGF}$ formulae).

Theorem 10 (Completeness)**.** *Let $\varphi \in \mathbb{GFU}$ be a formula in normal form. If φ is satisfiable, then there exists a mosaic $(\mathcal{M}, \mathcal{X})$ for φ.*

Proof. Assume that φ has some model $\mathcal{J}$. Since φ is equality-free, we can make the standard name assumption (SNA): $\mathsf{N_C}(\varphi) \subseteq \Delta^{\mathcal{J}}$ and $c^{\mathcal{I}} = c$ for all $c \in \mathsf{N_C}(\varphi)$. Now, let $\mathcal{I}$ be obtained from $\mathcal{J}$ by duplicating all anonymous individuals. Formally, let $\Delta_{anon} = \Delta^{\mathcal{J}} \setminus \mathsf{N_C}(\varphi)$ and $\Delta^{\mathcal{I}} = \mathsf{N_C}(\varphi) \cup \{1,2\} \times \Delta_{anon}$ (where we assume w.l.o.g. $\{1,2\} \times \Delta_{anon} \subseteq \mathsf{N_C}$ and $\{1,2\} \times \Delta_{anon} \cap \mathsf{N_C}(\varphi) = \emptyset$). Let $\pi : \Delta^{\mathcal{I}} \to \Delta^{\mathcal{J}}$ such that $\pi(c) = c$ for $c \in \mathsf{N_C}(\varphi)$ and $\pi((i,e)) = e$ otherwise. Now, for every $c \in \mathsf{N_C}(\varphi)$, we let $c^{\mathcal{I}} = c^{\mathcal{J}} = c$ and for each n-ary P, we let $P^{\mathcal{I}} = \{\boldsymbol{t} \in (\Delta^{\mathcal{I}})^n \mid \pi(\boldsymbol{t}) \in P^{\mathcal{J}}\}$. As φ does not contain equality, $\mathcal{J} \models \varphi$ implies $\mathcal{I} \models \varphi$. This duplication of anonymous individuals makes sure that for every non-constant domain element e, $\mathcal{I}$ contains a twin element $\tilde{e}$ different from e but with the same unary type. This property turns out to be crucial when dealing with the condition (G) from the definition of mosaics.

We show how to extract from $\mathcal{I}$ a mosaic $(\mathcal{M}, \mathcal{X})$ for φ. Note that, by construction, $\mathcal{I}$ satisfies the SNA.

Let $\mathcal{X}$ be any set with $\mathcal{X} \subseteq \mathsf{N_C}$, $\mathcal{X} \cap \Delta^{\mathcal{I}} = \emptyset$, and $|\mathcal{X}| = \mathit{width}(\varphi)$. We say a type τ can be *extracted* from $\mathcal{I}$ if τ can be obtained from $\mathcal{I}$ in 4 steps:

(a) Take any $S \subseteq \Delta^{\mathcal{I}}$ such that $\mathsf{N_C}(\varphi) \subseteq S$ and $|S| - |\mathsf{N_C}(\varphi)| \leq \mathit{width}(\varphi)$.

(b) Let $\tau^* = \{P(\boldsymbol{t}) \mid \boldsymbol{t} \subseteq S \wedge \boldsymbol{t} \in P^{\mathcal{I}}\}$.

(c) Let f be any injective function from $\mathit{dom}(\tau^*) \setminus \mathsf{N_C}(\varphi)$ to $\mathcal{X}$.

(d) Let τ be the type obtained from τ^* by replacing every occurence of $c \in \mathit{dom}(\tau^*) \setminus \mathsf{N_C}(\varphi)$ by $f(c)$.

The set $\mathcal{M}$ contains all types τ that can be extracted from $\mathcal{I}$. We now proceed to show that the constructed $(\mathcal{M}, \mathcal{X})$ is a mosaic for φ. To this end, we separately verify the properties from Definition 9:

(A) $|\mathcal{X}| \leq \mathit{width}(\varphi)$ is directly satisfied by construction (choice of $\mathcal{X}$).

(B) For all $\tau \in \mathcal{M}$, $\mathit{dom}(\tau) \subseteq \mathsf{N_C}(\varphi) \cup \mathcal{X}$. This is another immediate consequence from the construction.

(C) For all $\tau, \tau' \in \mathcal{M}$, $\tau|_{\mathsf{N_C}(\varphi)} = \tau'|_{\mathsf{N_C}(\varphi)}$. This is ensured by the fact that $\mathsf{N_C}(\varphi) \subseteq \mathit{dom}(\tau)$ as well as $\mathsf{N_C}(\varphi) \subseteq \mathit{dom}(\tau')$ and the uniform construction from $\mathcal{I}$.

(D) $\mathsf{U}(t,v) \in \tau$ for all $\tau \in \mathcal{M}$ and each pair $t, v \in \mathit{dom}(\tau)$. This is a direct consequence of the construction.

(E) $\tau \models \psi$ for all $\tau \in \mathcal{M}$ and all $\psi \in \mathbf{A}(\varphi)$.

Note that $\mathcal{I} \models \psi$ by assumption. By definition, $\mathcal{I}(\tau^*)$ is an induced substructure of $\mathcal{I}$, therefore $\mathcal{I}(\tau^*) \models \psi$ (since ψ is a formula in prenex form with only universal quantifiers). By construction, $\mathcal{I}(\tau^*)$ and $\mathcal{I}(\tau)$ are isomorphic, therefore $\mathcal{I}(\tau) \models \psi$ as well, and consequently $\tau \models \psi$.

(F) If $\tau \in \mathcal{M}$, $\forall \boldsymbol{x}.(R(\boldsymbol{t}) \to \exists \boldsymbol{y}.H(\boldsymbol{v})) \in \mathbf{E}(\varphi)$, and $R(g(\boldsymbol{t})) \in \tau$ for some $\boldsymbol{x}$-assigment g, then there is some $\tau' \in \mathcal{M}$ such that:

  (a) $H(h(g(\boldsymbol{v}))) \in \tau'$ for some $\boldsymbol{y}$-assignment h;

  (b) $\tau|_F = \tau'|_F$, where $F = \mathsf{N_C}(\varphi) \cup \{g(x) \mid x \in \boldsymbol{x} \cap \boldsymbol{v}\}$.

Let τ be created from S_τ and f_τ as described. From $R(g(\boldsymbol{t})) \in \tau$ follows $f_\tau^{-1}(g(\boldsymbol{t})) \in R^{\mathcal{I}}$. Then, since $\mathcal{I}$ must satisfy the above formula, we must find some $\boldsymbol{y}$-assignment $h_{\mathcal{I}}$ such that $h_{\mathcal{I}}(f_\tau^{-1}(g(\boldsymbol{v}))) \in H^{\mathcal{I}}$. We now let $S_{\tau'} = \mathsf{N_C}(\varphi) \cup \{h_{\mathcal{I}}(f_\tau^{-1}(g(v))) \mid v \in \boldsymbol{v}\}$. Furthermore, we choose $f_{\tau'}$ such that it coincides with f_τ on all elements from $S_\tau \cap S_{\tau'} \setminus \mathsf{N_C}(\varphi)$. Let now τ' be the type extracted via $S_{\tau'}$ and $f_{\tau'}$. Then (a) is satisfied via the $\boldsymbol{y}$-assignment h mapping every y to $f_{\tau'}(h_{\mathcal{I}}(y))$, and (b) is satisfied due to the choice of $f_{\tau'}$ and the definition of type extraction.

(G) If $t_1 \in \mathit{dom}(\tau_1) \cap \mathcal{X}$ and $t_2 \in \mathit{dom}(\tau_2) \cap \mathcal{X}$ for some $\tau_1, \tau_2 \in \mathcal{M}$, then there exists a type $\tau \in \mathcal{M}$ and a pair v_1, v_2 with $\mathit{dom}(\tau) \cap \mathcal{X} = \{v_1, v_2\}$ such that (i) $v_1 \neq v_2$, (ii) $\tau_1|^{\varphi}_{t_1} = \tau|^{\varphi}_{v_1}$, (iii) $\tau_2|^{\varphi}_{t_2} = \tau|^{\varphi}_{v_2}$;

For a domain element $e \in \Delta^{\mathcal{I}}$, we let $\mathcal{I}|^{\varphi}_e = \{R(\boldsymbol{t}) \in \mathit{base}(\varphi) \mid f(\boldsymbol{t}) \in R^{\mathcal{I}} \text{ with } f = \{\dot{x} \to e\} \cup \{c \to c^{\mathcal{I}} \mid c \in \mathsf{N_C}\}\}$.

Assume, τ_1 has been extracted via S_{τ_1} and f_{τ_1} such that $f_{\tau_1}(e_1) = t_1$ while τ_2 has been extracted via S_{τ_2} and f_{τ_2} such that $f_{\tau_2}(e_2) = t_2$. Note that then $\mathcal{I}|^{\varphi}_{e_1} = \tau|^{\varphi}_{t_1}$ and $\mathcal{I}|^{\varphi}_{e_2} = \tau|^{\varphi}_{t_2}$.

In case $e_1 \neq e_2$ we let $S'_\tau = \mathsf{N_C}(\varphi) \cup \{e_1, e_2\}$ and f_τ map e_1 to v_1 and e_2 to v_2 for two arbitrary distinct elements v_1, v_2 from $\mathcal{X}$. Then τ satisfies all requirements (i)-(iii).

Now consider the case $e_1 = e_2$. As argued above, $\tilde{e}_2 \in \Delta^{\mathcal{I}}$ with $\tilde{e}_2 \neq e_2 = e_1$ and $\mathcal{I}|^{\varphi}_{\tilde{e}_2} = \mathcal{I}|^{\varphi}_{e_1} = \tau_1|^{\varphi}_{t_1} = \tau_2|^{\varphi}_{t_2}$. Let now $S_\tau = \mathsf{N_C}(\varphi) \cup \{e_1, \tilde{e}_2\}$ and f_τ map e_1 to v_1 and $\tilde{e}_2$ to v_2 for two arbitrary distinct elements v_1, v_2 from $\mathcal{X}$. We again see that all requirements (i)-(iii) are satisfied.

□

Theorem 11 (Soundness)**.** *Let $\varphi \in \mathbb{GFU}$ be a formula in normal form. If there exists a mosaic $(\mathcal{M}, \mathcal{X})$ for φ, then φ is satisfiable.*

Proof. Assume a mosaic $(\mathcal{M}, \mathcal{X})$ for φ. An *instantiation* for a type $\tau \in \mathcal{M}$ is any injective function δ from $\mathit{dom}(\tau) \cap \mathcal{X}$ to $\mathsf{N_C} \setminus \mathcal{X}$. Given such τ and δ, we use $\delta(\tau)$ to denote the type that is obtained from τ by replacing every occurrence of a constant $c \in \mathit{dom}(\tau) \cap \mathcal{X}$ by $\delta(c)$. Our goal is to show how to inductively construct a possibly infinite sequence $\mathcal{S} = (\tau_0, \delta_0), (\tau_1, \delta_1), \ldots$ of pairs (τ_j, δ_j), where $\tau_j \in \mathcal{M}$ and δ_j is an instantiation for τ_j, such that $\bigcup_{i \geq 0} \delta_i(\tau_i) \models \varphi$.

In the base case, we let τ_0 be an arbitrary type from $\mathcal{M}$, and let δ_0 be any instantiation for τ_0.

For the inductive case, suppose $(\tau_0, \delta_0), \ldots, (\tau_{i-1}, \delta_{i-1})$ have been defined, where $i > 0$. We show how define the next segment $(\tau_i, \delta_i), \ldots, (\tau_m, \delta_m)$ of $\mathcal{S}$, where $m \geq i$ (we indeed may attach to $\mathcal{S}$ multiple new elements in one step). To this end, choose the smallest index $0 \leq j \leq i-1$ satisfying the following condition: there is $\forall \boldsymbol{x}.(R(\boldsymbol{t}) \to \exists \boldsymbol{y}.H(\boldsymbol{v})) \in \mathbf{E}(\varphi)$, and $R(g(\boldsymbol{t})) \in \delta_j(\tau_j)$ for some $\boldsymbol{x}$-assignment g. If such j does not exist, the construction of $\mathcal{S}$ is complete, and we can proceed to ($\star$) below, where we argue that $\bigcup_{0 \leq k < i} \delta_k(\tau_k) \models \varphi$. We assume that the above j exists. We first show in (†) how to define (τ_i, δ_i), and then in (‡) how to define the remaining $(\tau_{i+1}, \delta_{i+1}), \ldots, (\tau_m, \delta_m)$.

(†) From the $\boldsymbol{x}$-assignment g construct the following $\boldsymbol{x}$-assignment h. For every $x \in \boldsymbol{x}$, (i) let $h(x) = g(x)$, if $g(x) \in \mathit{dom}(\tau_j)$, and (ii) let $h(x) = \delta_j^-(g(x))$, if $g(x) \notin \mathit{dom}(\tau_j)$. Since $R(g(\boldsymbol{t})) \in \delta_j(\tau_j)$, we get $R(h(\boldsymbol{t})) \in \tau_j$. Since the condition (F) is satisfied by the mosaic, there exists a type $\tau' \in \mathcal{M}$ such that

(a) $H(f(g(\boldsymbol{v}))) \in \tau'$ for some $\boldsymbol{y}$-assignment f;

(b) $\tau|_F = \tau'|_F$, where $F = \mathsf{N_C}(\varphi) \cup \{g(x) \mid x \in \boldsymbol{x} \cap \boldsymbol{v}\}$.

We let $\tau_i = \tau'$, and define an injective function δ_i from $dom(\tau_i) \cap \mathcal{X}$ to $\mathsf{N_C} \setminus \mathcal{X}$ as follows. For every $c \in dom(\tau_i) \cap \mathcal{X}$, we let $\delta_i(c) = \delta_j(c)$ in case $c \in \{h(x) \mid x \in \boldsymbol{x} \cap \boldsymbol{v}\}$, and otherwise we let $\delta_i(c)$ be a fresh constant, i.e., a constant that does not appear in $\mathsf{N_C}(\varphi)$ or in the range of any instantition built so far.

(‡) Let N be the set of all constants that were freshly introduced in $\mathcal{S}$ by δ_i, i.e., N is the set of all $\delta_i(c)$ such that $c \in dom(\tau_i) \cap \mathcal{X}$ but $c \notin \{h(x) \mid x \in \boldsymbol{x} \cap \boldsymbol{v}\}$. Intuitively, in order to properly deal with the U predicate, we need to find in $\mathcal{M}$ proper types to connect every $c \in N$ with the relevant remaining constants of the sequence $\mathcal{S}$ constructed so far. Let $(d_1, d_1'), \ldots, (d_n, d_n')$ be an enumeration of all pairs (d, d') such that $d \in N$ and $d' \in \bigcup_{0 \leq k \leq i-1} ran(\delta_k)$. That is, d' is any constant that appears in the sequence S constructed so far but $d' \notin N \cup \mathsf{N_C}(\varphi)$. The definition of the segment $(\tau_{i+1}, \delta_{i+1}), \ldots, (\tau_m, \delta_m)$ of S in this inductive step is as follows. We let $m = i + n$, and for each $1 \leq k \leq n$, we select $(\tau_{i+1+k}, \delta_{i+1+k})$ as described next.

Assume an arbirary $1 \leq k \leq n$. We let $c = \delta_i^-(d_k)$, and let $\tau = \tau_l$ for some $0 \leq l \leq i$ such that $d_k' \in ran(\delta_l)$. Let $c' = \delta_l^-(d_k')$. Due to Condition (G) in the definition of mosaics, there exists a type $\tau^* \in \mathcal{M}$ such that (i) $dom(\tau) \cap \mathcal{X} = \{v_1, v_2\}$ for some v_1, v_2 with $v_1 \neq v_2$, (ii) $\tau_i|_c^\varphi = \tau^*|_{v_1}^\varphi$, and (iii) $\tau|_{c'}^\varphi = \tau^*|_{v_2}^\varphi$. ‘ Then we set $\tau_{i+1+k} = \tau^*$, and let $\delta_{i+1+k} = \{(v_1, d_k), (v_2, d_k')\}$.

The above completes the construction of a candidate model $\mathcal{J}$ for φ defined by $\bigcup_{i \geq 0} \delta_i(\tau_i)$. We now show that $\mathcal{J}$ is indeed a model of φ.

First, observe that $\boldsymbol{t} \in P^{\mathcal{J}}$ exactly if $P(\boldsymbol{t}) \in \delta_i(\tau_i)$ for some i. Due to the definition of mosaic and the construction of $\mathcal{S}$, this is exactly the case if $P(\boldsymbol{t}) \in \delta_i(\tau_i)$ for **all** i where $\boldsymbol{t} \in ran(\delta_i)$.

We note two consequences of our construction:

(i) Every $\delta_i(\tau_i)$ is an induced substructure of $\mathcal{J}$.

(ii) For any two elements $e_1, e_2 \in \Delta^{\mathcal{J}}$, there is at least one (τ_i, δ_i) in $\mathcal{S}$ with $\{e_1, e_2\} \in ran(\delta_i)$.

We now show that the predicate U is interpreted in the intended way. Thanks to Fact (ii), we know that any two elements of $\Delta^{\mathcal{J}}$ co-occur in one type, hence condition (D) ensures $\mathsf{U}^{\mathcal{J}} = \Delta^{\mathcal{J}} \times \Delta^{\mathcal{J}}$.

Next we show $\mathcal{J} \models \varphi$.

We start with some sentence $\psi = \forall \boldsymbol{x}.(B_1(\boldsymbol{t}_1) \wedge \ldots \wedge B_n(\boldsymbol{t}_n) \rightarrow H_1(\boldsymbol{v}_1) \vee \ldots \vee H_m(\boldsymbol{v}_m))$ coming from $\mathbf{A}(\varphi)$. W.l.o.g. we assume $B_1(\boldsymbol{t}_1)$ to be the guarding atom, i.e. $\boldsymbol{x} \subseteq \boldsymbol{t}_1$. Now assume there is an arbitrary $\boldsymbol{x}$-assignment f such that $f(\boldsymbol{t}_i) \in B_i^{\mathcal{I}}$ for $1 \leq i \leq n$. Let (τ, δ) be the sequence element for which $B_1(f(\boldsymbol{t}_1)) \in \delta(\tau)$. As $\delta(\tau)$ is an induced substructure of $\mathcal{I}$ containing all elements from $f(\boldsymbol{x})$ as well as all constants, it follows that $B_i(f(\boldsymbol{t}_i)) \in \delta(\tau)$ for all $i > 1$ as well, therefore $B_i(\delta^{-1}(f(\boldsymbol{t}_i))) \in \tau$. Due to Condition (E), we have $\tau \models \psi$, therefore $H_j(\delta^{-1}(f(\boldsymbol{v}_j))) \in \tau$ for some j with $1 \leq j \leq m$. This ensures $H_j(f(\boldsymbol{v}_j)) \in \delta(\tau)$ and consequently $f(\boldsymbol{v}_j) \in H_j^{\mathcal{J}}$. Hence we have shown $\mathcal{J} \models \psi$.

Now consider some sentence $\psi = \forall \boldsymbol{x}.(R(\boldsymbol{t}) \rightarrow \exists \boldsymbol{y}.H(\boldsymbol{v}))$ coming from $\mathbf{E}(\varphi)$. Assume an arbitrary $\boldsymbol{x}$-assignment g such that $g(\boldsymbol{t}) \in R^{\mathcal{I}}$. Let (τ, δ) be the sequence element for which $R(g(\boldsymbol{t})) \in \delta(\tau)$. By construction (†) of $\mathcal{S}$ there must be a later sequence element (τ', δ') such that $H(f(g(\boldsymbol{v}))) \in \delta'(\tau')$ for some $\boldsymbol{y}$-assignment f. Hence we have shown $\mathcal{J} \models \psi$. □

5 Complexity of $\mathbb{TGF}$ without Equality

Using the characterization of the previous section, we can infer worst-case optimal upper bounds for satisfiability checking in $\mathbb{GFU}$, and thus in $\mathbb{TGF}$.

Theorem 12 (Complexity). *Deciding satisfiability of* $\mathbb{TGF}$ *and of* $\mathbb{GFU}$ *formulae without equality is* N2ExpTime*-complete. The problem is* NExpTime*-complete under the assumption that predicate arities are bounded by a constant.*

Proof. Due to Propositions 3 and 5, it suffices to show the two upper bounds for $\mathbb{GFU}$ formulae in normal form. Due to Theorems 10 and 11, we can decide the satisfiability of a formula $\varphi \in \mathbb{GFU}$ in normal form by checking the existence of a mosaic for φ. Our approach is to non-deterministically guess a pair $(\mathcal{M}, \mathcal{X})$ of a set $\mathcal{M}$ of types over $\mathsf{N_P}(\varphi)$ together with a set of constants $\mathcal{X}$ of cardinality at most $\mathit{width}(\varphi)$, and then verify that $(\mathcal{M}, \mathcal{X})$ is indeed a mosaic for φ. Note that given a candidate $(\mathcal{M}, \mathcal{X})$ as input we can check in polynomial time whether $(\mathcal{M}, \mathcal{X})$ satisfies all the conditions given in Definition 9. Observe that the number of ground atoms over the signature of φ with arguments from $\mathsf{N_C}(\varphi) \cup \mathcal{X}$ is bounded by $|\mathsf{N_P}(\varphi)| \cdot (|\mathsf{N_C}(\varphi)| + \mathit{width}(\varphi))^k$, where k is the maximal arity of predicates in φ. Consequently, we can restrict ourselves to candidates $(\mathcal{M}, \mathcal{X})$, where $\mathcal{M}$ has no more than $2^{|\mathsf{N_P}(\varphi)| \cdot (|\mathsf{N_C}(\varphi)| + \mathit{width}(\varphi))^k}$ types. Since this bound is double exponential in the size of φ, but only single exponential under the assumption that k is a constant, the two upper bounds follow.

The matching lower bound for the bounded arity follows from the complexity of $\mathbb{FO}^2$ [13]. N2ExpTime-hardness for unbounded arity follows from a reduction from the tiling problem of a grid of doubly exponential size [7] shown in the following.

Let k be a natural number. We will construct a $\mathbb{GFU}$ theory describing a tiling of a $2^{(2^k)} \times 2^{(2^k)}$ grid. We will have domain elements corresponding to elements of this grid. To identify the position of each of those grid elements, we have to assign them x- and y-coordinates between 0 and $2^{(2^k)} - 1$. We will express them in binary encoding, i.e. as 2^k-dimensional bitvectors. In order to express that the ℓth position in the bitvector corresponding to the x-coordinate of some grid element e carries a 0, we let $\mathit{Sel}_0(e, 0, \ell_{\text{binary}})$ hold, where ℓ_{binary} is a list of length k containing 0s and 1s, expressing the binary encoding of ℓ. That is, the arity of Sel_0 is $k+2$. In order to express that the ℓth bit is 1, we use $\mathit{Sel}_1(e, 0, \ell_{\text{binary}})$. To express the corresponding information for the y-coordinate, we use $\mathit{Sel}_0(e, 1, \ell_{\text{binary}})$ and $\mathit{Sel}_1(e, 1, \ell_{\text{binary}})$, respectively.

In the following, we omit leading universal quantifiers; all formulae are sentences. We make use of two distinguished constants, 0 and 1.

First, we will make sure that for every pair of x- and y-coordinates, a corresponding grid element exists. We do so by creating a binary tree structure of exponential depth, where at the ℓth level two *Next*-successors are created: one where the ℓth bit is set to 0 and one where it is set to 1. All the previously set bits are propagated toward the leaves of the tree, which then correspond to the grid elements.

We create the tree root.

$$\exists x.\, \mathit{ToSelect}(x, 0^{k+1}) \tag{15}$$

At the ℓth level ($\boldsymbol{z} = \ell_{\text{binary}}$), two successors are created with the ℓth bit set to 0 and 1, respectively ($b \in \{0, 1\}$).

$$\mathit{ToSelect}(x, \boldsymbol{z}) \to \exists y. \mathit{Next}(x, y) \wedge \mathit{Sel}(y, \boldsymbol{z}) \wedge \mathit{Sel}_b(y, \boldsymbol{z}) \tag{16}$$

At the ℓ+1st level (where the ℓth bit has just been selected), we indicate that in the next step, the ℓ+1st bit is to be selected ($0 \leq i \leq k-1$).

$$\mathit{Sel}(x, \boldsymbol{z}, 0, 1^i) \to \mathit{ToSelect}(x, \boldsymbol{z}, 1, 0^i) \tag{17}$$

The next two rules create versions of the *Next* predicate which carry all possible $(k+1)$ary bitvectors as additional parameter. We will need them as guards later.

$$\mathit{Next}(x, y) \quad \to \quad \mathit{Next}'(x, y, 0^{k+1}) \tag{18}$$

$$\mathit{Next}'(x, y, \boldsymbol{z}, 0, 1^i) \quad \to \quad \mathit{Next}'(x, y, \boldsymbol{z}, 1, 0^i) \tag{19}$$

We propagate earlier choices made for the bits along the *Next* predicate, making use of the auxiliary *Next*′ predicate, to keep everything guarded.

$$Sel_b(x, \boldsymbol{z}) \wedge Next'(x, y, \boldsymbol{z}) \rightarrow Sel_b(y, \boldsymbol{z}) \tag{20}$$

Once the last bit is set, we indicate that we have reached a leaf.

$$Sel(x, 1^{k+1}) \rightarrow Complete(x) \tag{21}$$

This finishes the creation (and "coordinatization") of the grid elements. In the next step, we enforce that any two grid elements with the same y-coordinate and subsequent x-coordinates are connected via a H predicate ($b \in \{0, 1\}$ and $0 \leq i \leq k - 1$).

$$\mathsf{U}(x, y) \wedge Complete(x) \wedge Complete(y) \rightarrow ChkH(x, y, 0, 0^k) \tag{22}$$
$$ChkH(x, y, 0, \boldsymbol{z}, 0, 1^i) \wedge Sel_1(x, 0, \boldsymbol{z}, 0, 1^i) \wedge Sel_0(y, 0, \boldsymbol{z}, 0, 1^i) \rightarrow ChkH(x, y, 0, \boldsymbol{z}, 1, 0^i) \tag{23}$$
$$ChkH(x, y, 0, \boldsymbol{z}, 0, 1^i) \wedge Sel_0(x, 0, \boldsymbol{z}, 0, 1^i) \wedge Sel_1(y, 0, \boldsymbol{z}, 0, 1^i) \rightarrow ChkH'(x, y, 0, \boldsymbol{z}, 1, 0^i) \tag{24}$$
$$ChkH'(x, y, 0, \boldsymbol{z}, 0, 1^i) \wedge Sel_b(x, 0, \boldsymbol{z}, 0, 1^i) \wedge Sel_b(y, 0, \boldsymbol{z}, 0, 1^i) \rightarrow ChkH'(x, y, 0, \boldsymbol{z}, 1, 0^i) \tag{25}$$
$$ChkH'(x, y, 0, 1^k) \wedge Sel_b(x, 0, 1^k) \wedge Sel_b(y, 0, 1^k) \rightarrow ChkH''(x, y, 1, 0^k) \tag{26}$$
$$ChkH''(x, y, 1, \boldsymbol{z}, 0, 1^i) \wedge Sel_b(x, 1, \boldsymbol{z}, 0, 1^i) \wedge Sel_b(y, 1, \boldsymbol{z}, 0, 1^i) \rightarrow ChkH''(x, y, 1, \boldsymbol{z}, 1, 0^i) \tag{27}$$
$$ChkH''(x, y, 1, 1^k) \wedge Sel_b(x, 1, 1^k) \wedge Sel_b(y, 1, 1^k) \rightarrow H(x, y) \tag{28}$$

Thereby, the atom $ChkH(x, y, 0, \ell_{\text{binary}})$ is supposed to hold for all grid elements x and y for which the lowest ℓ bits of their x-coordinate have the shape 1^ℓ and 0^ℓ, respectively. Moreover, the atom $ChkH'(x, y, 0, \ell_{\text{binary}})$ is supposed to hold for all grid elements x and y for which the lowest ℓ bits of x and y represent consecutive binary numbers. Finally, $ChkH''(x, y, 1, \ell_{\text{binary}})$ is supposed to hold for any x and y with consecutive x-coordinates and coinciding lowest ℓ bits of the y-coordinate. Consequently, $H(x, y)$ follows for every x and y with consecutive x-coordinates and coinciding y-coordinates.

In the same way, we make sure that any two grid elements with the subsequent y-coordinates and equal x-coordinates are connected via a V predicate ($b \in \{0, 1\}$ and $0 \leq i \leq k - 1$).

$$\mathsf{U}(x, y) \wedge Complete(x) \wedge Complete(y) \rightarrow ChkV(x, y, 1, 0^k) \tag{29}$$
$$ChkV(x, y, 1, \boldsymbol{z}, 0, 1^i) \wedge Sel_1(x, 1, \boldsymbol{z}, 0, 1^i) \wedge Sel_0(y, 1, \boldsymbol{z}, 0, 1^i) \rightarrow ChkV(x, y, 1, \boldsymbol{z}, 1, 0^i) \tag{30}$$
$$ChkV(x, y, 1, \boldsymbol{z}, 0, 1^i) \wedge Sel_0(x, 1, \boldsymbol{z}, 0, 1^i) \wedge Sel_1(y, 1, \boldsymbol{z}, 0, 1^i) \rightarrow ChkV'(x, y, 1, \boldsymbol{z}, 1, 0^i) \tag{31}$$
$$ChkV'(x, y, 1, \boldsymbol{z}, 0, 1^i) \wedge Sel_b(x, 1, \boldsymbol{z}, 0, 1^i) \wedge Sel_b(y, 1, \boldsymbol{z}, 0, 1^i) \rightarrow ChkV'(x, y, 1, \boldsymbol{z}, 1, 0^i) \tag{32}$$
$$ChkV'(x, y, 1, 1^k) \wedge Sel_b(x, 1, 1^k) \wedge Sel_b(y, 1, 1^k) \rightarrow ChkV''(x, y, 0, 0^k) \tag{33}$$
$$ChkV''(x, y, 0, \boldsymbol{z}, 0, 1^i) \wedge Sel_b(x, 0, \boldsymbol{z}, 0, 1^i) \wedge Sel_b(y, 0, \boldsymbol{z}, 0, 1^i) \rightarrow ChkV''(x, y, 0, \boldsymbol{z}, 1, 0^i) \tag{34}$$
$$ChkV''(x, y, 0, 1^k) \wedge Sel_b(x, 0, 1^k) \wedge Sel_b(y, 0, 1^k) \rightarrow V(x, y) \tag{35}$$

This way, we have established a doubly exponential grid. Encoding a tiling on top of such a grid is standard. □

6 Undecidability of $\mathbb{TGF}$ with Equality

In the presence of equality, we can show the undecidability of satisfiability of $\mathbb{GFU}$ (and hence of $\mathbb{TGF}$) by a reduction from the tiling problem for an infinite grid [7].[1] We can construct a $\mathbb{GFU}$ formula with equality such that its universal model represents an $\mathbb{N} \times \mathbb{N}$ grid. Thereby, the

[1] As mentioned in the introduction, this undecidability result can be inferred from the undecidability of the Goldfarb class, using the reduction in [14] (Section 4.2.3).

domain elements of the model correspond to grid positions and every position is connected to its upper neighbor by a binary predicate V and to its right neighbor by a binary predicate H.

In the following, we omit leading universal quantifiers; all formulae are sentences. We start our modeling by ensuring there is exactly one leftmost, bottommost position of the grid, i.e., the "origin".

$$\exists x.\mathit{Orig}(x) \tag{36}$$

$$\mathsf{U}(x,y) \wedge \mathit{Orig}(x) \wedge \mathit{Orig}(y) \rightarrow x \approx y \tag{37}$$

Any two domain elements co-occur together with the origin in a ternary auxiliary predicate *ChkFunc*.

$$\mathsf{U}(x,y) \rightarrow \exists z.\mathit{ChkFunc}(x,y,z) \wedge \mathit{Orig}(z) \tag{38}$$

Intuitively, $\mathit{ChkFunc}(x,y,z)$ indicates that we will enforce that if z is connected with both x and y by predicate V (or H), then x and y must coincide; in other words, as x and y are arbitrary elements, z has only one outgoing V-connection and one outgoing H-connection. The following two sentences implement this.

$$\mathit{ChkFunc}(x,y,z) \wedge H(z,x) \wedge H(z,y) \quad \rightarrow \quad x \approx y \tag{39}$$

$$\mathit{ChkFunc}(x,y,z) \wedge V(z,x) \wedge V(z,y) \quad \rightarrow \quad x \approx y \tag{40}$$

In particular, this makes sure that the origin has exactly one right and one upper neighbor. Also, we propagate this "local funtionality" enforcing predicate along the (known to be unique) V- and H-connections.

$$\mathit{ChkFunc}(x,y,z) \quad \rightarrow \quad \exists w.\mathit{ChkFunc}(x,y,w) \wedge H(z,w) \tag{41}$$

$$\mathit{ChkFunc}(x,y,z) \quad \rightarrow \quad \exists w.\mathit{ChkFunc}(x,y,w) \wedge V(z,w) \tag{42}$$

With these axioms alone, the corresponding universal model would resemble an infinite binary tree, with the origin as root and every node having (exactly) one H-successor and (exactly) one V-successor. The next axioms make sure that for every element e in our structure, the element reached from e via an H-V-path coincides with the element reached from e via a V-H-path, using another auxiliary 5-ary predicate *ChkSq* which is handled in a way that $\mathit{ChkSq}(x,y,z_1,z_2,z_3)$ is only entailed whenever z_1 has z_2 as right neighbor and z_3 as upper neighbor.

Again, we start ensuring this for e being the origin and then work our way through the structure along the (unique) H- and V- connections.

$$\mathsf{U}(x,y) \rightarrow \exists z_1 z_2 z_3.\mathit{ChkSq}(x,y,z_1,z_2,z_3) \wedge \mathit{Orig}(z_1) \wedge H(z_1,z_2) \wedge V(z_1,z_3) \tag{43}$$

$$\mathit{ChkSq}(x,y,z_1,z_2,z_3) \rightarrow \exists w_1 w_2.\mathit{ChkSq}(x,y,z_2,w_1,w_2) \wedge H(z_2,w_1) \wedge V(z_2,w_2) \tag{44}$$

$$\mathit{ChkSq}(x,y,z_1,z_2,z_3) \rightarrow \exists w_1 w_2.\mathit{ChkSq}(x,y,z_3,w_1,w_2) \wedge H(z_3,w_1) \wedge V(z_3,w_2) \tag{45}$$

Finally, we ensure that if $\mathit{ChkSq}(x,y,z_1,z_2,z_3)$ holds and x is the right neighbor of z_2 and y is the upper neighbor of z_3, that then x and y must coincide.

$$\mathit{ChkSq}(x,y,z_1,z_2,z_3) \wedge V(z_2,x) \wedge H(z_3,y) \rightarrow x \approx y \tag{46}$$

This finishes our modeling of the infinite grid. It is now straightforward to model a tiling on top of this, and we obtain the following theorem.

Theorem 13 (Undecidability with Equality)**.** *Checking satisfiability of* $\mathbb{TGF}$ *formulae with equality is undecidable. The same applies to* $\mathbb{GFU}$ *formulae with equality.*

7 Further Undecidable Extensions

We review here some further natural extensions of $\mathbb{TGF}$ and find that they lead to undecidability.

Relaxing guardedness further. Unguarded quantification of subformulae with three variables would allow to express any formula of the three-variable fragment of $\mathbb{FO}$, denoted $\mathbb{FO}^3$, for which satisfiability is undecidable (as $\mathbb{FO}^3$ contains the class of $\mathbb{FO}$ sentences with quantifier prefix $\forall\exists\forall$ which is undecidable [15]).

Counting. $\mathbb{FO}^2$ can be extended by counting quantifiers of the shape $\exists^{=n}$, $\exists^{\leq n}$, and $\exists^{\geq n}$, yielding a logic denoted $\mathbb{C}^2$. This extension (which helps to capture DLs with cardinality restrictions) by itself does not lead to an increase in complexity of satisfiability checking [17]. Yet, this enrichment is detrimental when mixing it with the guarded fragment: via the $\mathbb{C}^2$ sentence $\forall x.\exists^{=1}y.F(x,y)$ we can enforce that F must be interpreted as a functional binary relation. Yet, adding a functional relation to $\mathbb{GF}$ is known to cause undecidability [12].

Conjunctive Queries. Instead of asking for satisfiability of a $\mathbb{TGF}$ theory, an often considered problem stemming from database theory is also if it entails a Boolean conjunctive query (i.e., an existentially quantified conjunction of atoms). However, conjunctive query entailment has been shown to be undecidable already for $\mathbb{FO}^2$ alone [18]. This also shows that any attempt of extending $\mathbb{TGF}$ such that it incorporates $\mathbb{FO}$ fragments that can express negated Boolean conjunctive queries (such as the unary negation fragment [22] or the guarded negation fragment [5]) will lead to undecidability.

Loose guardedness. It has been shown that $\mathbb{GF}$ remains decidable if the guardedness restriction is relaxed, leading to notions such as the loosely guarded fragment, the packed fragment or the clique-guarded fragment. For most restrictive notion of those, the loosely guarded fragment [24], the guard does not need to be one atom containing all free variables, rather it can be a conjunction of atoms with the property that any pair of free variables occurs together in one of those conjuncts. It is not hard to see that in the presence of the U predicate (or if such a predicate can be axiomatized as in $\mathbb{TGF}$), we can create a "loose guard" $\bigwedge_{\{x,y\}\subseteq \boldsymbol{x}} \mathsf{U}(x,y)$ for any set $\boldsymbol{x}$ of free variables. This allows to quantify over the full domain, hence every $\mathbb{FO}$ formula is equivalent to such a loosely guarded one. Consequently, a hypothetical "loosely triguarded fragment" would be as expressive as $\mathbb{FO}$, hence undecidable.

8 Conclusion

In this paper, we have introduced the triguarded fragment of $\mathbb{FO}$ which subsumes both $\mathbb{GF}$ and $\mathbb{FO}^2$. We clarified the computational complexity of satisfiability checking in this fragment, both for the bounded and unbounded arity case. We discussed that diverse natural extensions of the fragment lead to undecidability.

We foresee several avenues of future work. It seems that, while full equality leads to undecidability, it should be possible to allow for equality atoms of the form $x \approx c$ with $c \in \mathsf{N_C}$ without impacting the complexity results. This would allow to capture a modeling feature known as *nominals* in DLs.

While both $\mathbb{GF}$ [12] and $\mathbb{FO}^2$ [16] are known to have the finite model property, the status of $\mathbb{TGF}$ in this respect is open. On a first glance, it seems the arguments for establishing the finite model property of the two fragments are incompatible and neither can be easily adapted to show that property for $\mathbb{TGF}$. Still, we conjecture that $\mathbb{TGF}$ has the finite model property which would imply that satisfiability and finite satisfiability (and their respective complexities) coincide.

Acknowledgments

We thank Emanuel Kieroński and the anonymous reviewers of this work for the valuable comments. We are also grateful to Pierre Bourhis, Michael Morak, and Andreas Pieris for clarifying some questions regarding their paper [10].

Sebastian Rudolph has been supported by the Institute of Logic and Computation (E192) at TU Wien and received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 771779). Mantas Šimkus has been supported by the Austrian Science Fund (FWF) projects P30360 and P30873.

References

[1] Hajnal Andréka, Johan F. A. K. van Benthem, and István Németi. Modal languages and bounded fragments of predicate logic. *J. of Philosophical Logic*, 27(3):217–274, 1998.

[2] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, second edition, 2007.

[3] Franz Baader, Ian Horrocks, Carsten Lutz, and Ulrike Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.

[4] Vince Bárány, Georg Gottlob, and Martin Otto. Querying the Guarded Fragment. *Logical Methods in Computer Science*, Volume 10, Issue 2, May 2014.

[5] Vince Bárány, Balder ten Cate, and Luc Segoufin. Guarded negation. *J. of the ACM*, 62(3):22:1–22:26, 2015.

[6] Patrick Blackburn and Johan Van Benthem. Modal logic: a Semantic Perspective. In Frank Wolter Patrick Blackburn, Johan van Benthem, editor, *Handbook of Modal Logic*, pages 1–82. Elsevier, 2006.

[7] Egon Börger, Erich Grädel, and Yuri Gurevich. *The Classical Decision Problem*. Springer, 1997.

[8] Alexander Borgida. On the relative expressiveness of description logics and predicate logics. *Artif. Intell.*, 82(1-2):353–367, 1996.

[9] Pierre Bourhis, Michael Morak, and Andreas Pieris. Personal Communication (23rd of July 2018).

[10] Pierre Bourhis, Michael Morak, and Andreas Pieris. Making cross products and guarded ontology languages compatible. In *Proc. of IJCAI 2017*, 2017.

[11] Erich Grädel. Description logics and guarded fragments of first order logic. In *Proc. of DL 1998*, 1998.

[12] Erich Grädel. On the restraining power of guards. *J. Symb. Log.*, 64(4):1719–1742, 1999.

[13] 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.

[14] Yevgeny Kazakov. *Saturation-Based Decision Procedures for Extensions of the Guarded Fragment*. PhD thesis, Universität des Saarlandes, Saarbrücken, Germany, March 2006.

[15] Harry R. Lewis. *Unsolvable Classes of Quantificational Formulas*. Addison-Wesley, 1979.

[16] Michael Mortimer. On languages with two variables. *Math. Log. Q.*, 21(1):135–140, 1975.

[17] Ian Pratt-Hartmann. Complexity of the two-variable fragment with counting quantifiers. *J. of Logic, Language and Information*, 14:369–395, 2005.

[18] Riccardo Rosati. The limits of querying ontologies. In Thomas Schwentick and Dan Suciu, editors, *Proc. 11th Int. Conf. Database Theory (ICDT'07)*, volume 4353 of *LNCS*, pages 164–178. Springer, 2007.

[19] Sebastian Rudolph. Foundations of description logics. In Axel Polleres, Claudia d'Amato, Marcelo Arenas, Siegfried Handschuh, Paula Kroner, Sascha Ossowski, and Peter F. Patel-Schneider, ed-

itors, *Reasoning Web. Semantic Technologies for the Web of Data – 7th International Summer School 2011*, volume 6848 of *LNCS*, pages 76–136. Springer, 2011.

[20] Sebastian Rudolph, Markus Krötzsch, and Pascal Hitzler. All elephants are bigger than all mice. In *Proc. of DL 2008*, 2008.

[21] Dana Scott. A decision method for validity of sentences in two variables. *Journal of Symbolic Logic*, 27(377):74, 1962.

[22] Luc Segoufin and Balder ten Cate. Unary negation. *Logical Methods in Computer Science*, 9(3), 2013.

[23] Balder ten Cate and Massimo Franceschet. Guarded fragments with constants. *Journal of Logic, Language and Information*, 14(3):281–288, 2005.

[24] Johan van Benthem. Dynamic bits and pieces. Technical Report LP-97-01, ILLC, University of Amsterdam, 1997. Available at http://www.illc.uva.nl/Publications/reportlist.php?Series=LP.