HAL open science

Generalized Bundled Fragments for First-Order Modal Logic

Mo Liu, Anantha Padmanabha, R Ramanujam, Yanjing Wang

To cite this version:

Mo Liu, Anantha Padmanabha, R Ramanujam, Yanjing Wang. Generalized Bundled Fragments for First-Order Modal Logic. 47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022), Aug 2022, Vienna, Austria. pp.70:1–70:14, 10.4230/LIPIcs.MFCS.2022.70 . hal-03765358

HAL Id: hal-03765358

https://inria.hal.science/hal-03765358v1

Submitted on 31 Aug 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Generalized Bundled Fragments for First-Order Modal Logic

Mo Liu
LORIA, Unviersity of Lorraine, Nancy, France

Anantha Padmanabha
DI ENS, École Normale Supérieure, Université PSL, CNRS, Inria, Paris, France

R. Ramanujam
Institute of Mathematical Sciences, HBNI, Chennai, India (Retired)
Azim Premji University, Bengaluru (Visiting)

Yanjing Wang
Department of Philosophy, Peking University, Beijing, China

Abstract

When we bundle quantifiers and modalities together (as in $\exists x\Box$, $\Diamond\forall x$ etc.) in first-order modal logic (FOML), we get new logical operators whose combinations produce interesting *bundled* fragments of FOML. It is well-known that finding decidable fragments of FOML is hard, but existing work shows that certain bundled fragments are decidable [14], without any restriction on the arity of predicates, the number of variables, or the modal scope. In this paper, we explore generalized bundles such as $\forall x\forall y\Box, \forall x\exists y\Diamond$ etc., and map the terrain with regard to decidability, presenting both decidability and undecidability results. In particular, we propose the loosely bundled fragment, which is decidable over increasing domains and encompasses all known decidable bundled fragments.

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

Keywords and phrases bundled fragments, first-order modal logic, decidability, tableaux

Digital Object Identifier 10.4230/LIPIcs.MFCS.2022.70

Related Version *Full Version*: `https://arxiv.org/abs/2202.01581`

Funding This work is supported by NSSF grant 19BZX135.
Anantha Padmanabha: The author is funded by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute).

Acknowledgements The authors thank the anonymous reviewers of MFCS2022 for their comments that improved the presentation of the paper.

1 Introduction

While propositional modal logic (ML) has had extensive applications in system verification and artificial intelligence, and first-order logic (FOL) in finite model theory and database theory, first-order modal logic (FOML) has been studied much less, as it seems to combine the worst of both computationally, leading to undecidability. FOML is a natural specification language for state transition systems where states are given by first-order descriptions of computational domains, with applicability in the realm of database updates, in the control of infinite-state systems, networks with unbounded parallelism and cryptographic protocols. This motivates the study of *decidable fragments* of FOML.

© Mo Liu, Anantha Padmanabha, R. Ramanujam, and Yanjing Wang;
licensed under Creative Commons License CC-BY 4.0
47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022).
Editors: Stefan Szeider, Robert Ganian, and Alexandra Silva; Article No. 70; pp. 70:1–70:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

This is a challenge, since even the two-variable fragment of FOML with one unary predicate is undecidable over almost all useful model classes [15]. This situation is to be contrasted against the robust decidability of propositional modal logics ([16, 1, 7]), and the many decidable fragments of first-order logic ([4]).

Despite such discouragement, there have been a few successful attempts: for instance, the *monodic restriction*, mandating only one free variable in the scope of any modal subformula, yields decidability when combined with a decidable fragment of FOL ([8, 19]). This idea, arising originally from description logics (cf. [9]) has led to applications in temporal and epistemic logics ([6], [10], [2, 3]).

Rather than placing a restriction on variables, or on *quantification scope* as in the case of guarded fragments [1], Wang ([17]) suggested a fragment in which the existential quantifier and the box modality were always bundled together to appear as a single quantifier-modality pair ($\exists x\Box$). The resulting fragment of FOML enjoys many attractive properties: finite tree model property, PSpace decision procedure and a simple axiom system, without any restriction on predicates or the occurrences of variables. The new operator $\exists x\Box$ captures the logical structure of various *knowing-wh* expressions such as *knowing what, knowing how, knowing why*, and so on (cf. [18]), e.g., *knowing how to achieve* φ can be rendered as *there is* a plan x such that the agent *knows that* x can be executed and will guarantee φ.

In [14], we took the next step by considering not only the combination $\exists x\Box$ but also its companion $\forall x\Box$: the logic with both of these combinations continued to be decidable (over increasing domain models). Such modal-quantifier combinations were thus called *bundled fragments* of FOML. Over models where the domain remains the same in all states, the fragment with both $\exists x\Box$ and $\forall x\Box$ is undecidable, while the $\exists x\Box$-fragment is still decidable.

Clearly, we can define more bundles such as $\Box\forall x$, $\Box\exists x$, etc., and indeed further combinations such as $\forall x\forall y\Box$, and combinations thereof. These (generalized) bundled fragments offer us many interesting possibilities for system specification:

- $\neg\exists x\Box\ (x < c)$: No element is guaranteed to be bounded by constant c (after update).
- $\exists x\Box\ \Box\exists y\ (x > y)$: There is an element that dominates some element after every update.
- $\Box\exists x\ \Box\forall y\ (x \leq y)$: All updates admit a local minimum.
- $\exists x\Box\ (\exists y\Box\ (x > y) \wedge \exists y\Box\ (x < y))$: There is an element that dominates another no matter the update and is dominated by another no matter the update.

Computationally this raises the natural question of what is the most general *bundled fragment* that is decidable. This is the project taken up in this paper. We consider all possible combinations of the bundled formulas and classify their decidability status. The classification is described in Table 1. We provide a trichotomy: decidable fragments, undecidable fragments and fragments that do not have a finite model property (but where decidability is open).

Towards proving the trichotomy, we first define the notion of *loosely bundled fragment* which subsumes many decidable bundled fragments of FOML and prove its decidability via a tableau method. We also prove the decidability of another combination of bundled operators where we allow only formulas of the form $\forall x\Box + \Box\forall x + \Box\exists x$ (but is not included in the loosely bundled fragment). This requires us to introduce a new proof technique that helps us switch quantifiers in a specific context.

Due to space restrictions, we present only the main ideas and proof techniques of the decidable fragments over increasing domain models in the paper. The proof details, as well as undecidability and lack of finite model property for the other fragments can be found in the detailed technical report in the arXiv [13].

Table 1 Satisfiability problem classification for combinations of bundled fragments over increasing domain models. A "$*$" means no matter the corresponding bundle is included or not.

$\forall\Box$	$\exists\Box$	$\Box\forall$	$\Box\exists$	Decidability
✓	✗	✗	✗	Subsumed by the Loosely Bundled Fragment
✗	✓	✗	✗	
✗	✗	✓	✗	
✗	✗	✗	✓	
✓	✓	✗	✗	
✗	✗	✓	✓	
$*$	✓	✓	$*$	Undecidable
✗	✓	✗	✓	No Finite Model Property
✓	✓	✗	✓	Undecidable
✓	✗	✓	✓	ExpSpace
Loosely Bundled Fragment				ExpSpace

2 Syntax and Semantics

The syntax of first-order modal logic is given by extending the first-order logic with modal operators. Note that we exclude equality, constants and function symbols from the syntax.

▶ **Definition 1** (FOML syntax). *Given a countable set of predicates $\mathcal{P}$ and a countable set of variables Var, the syntax of* FOML *is given by:*

$$\alpha ::= P(x_1, \ldots, x_n) \mid \neg\alpha \mid \alpha \wedge \alpha \mid \exists x\alpha \mid \Box\alpha$$

where $P \in \mathcal{P}$ has arity n and $x, x_1, \ldots, x_n \in \mathsf{Var}$.

The boolean connectives $\vee, \rightarrow, \leftrightarrow$, and the modal operator $\Diamond$ which is the dual of $\Box$, and the quantifier $\forall$ are all defined in the standard way. The notion of free variables, denoted by $\mathsf{FV}(\alpha)$ is similar to what we have for first-order logic with $\mathsf{FV}(\Box\alpha) = \mathsf{FV}(\alpha)$. We write $\alpha(x)$ to mean that x occurs as a free variable of α. Also, $\alpha[y/x]$ denotes the formula obtained from α by replacing every free occurrence of x by y.

▶ **Definition 2** (FOML structure). *An increasing domain model for* FOML *is a tuple $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$ where $\mathcal{W}$ is a non-empty countable set called* worlds;[1] *$\mathcal{D}$ is a non-empty countable set called* domain; *$\mathcal{R} \subseteq (\mathcal{W} \times \mathcal{W})$ is the* accessibility relation. *The map $\delta : \mathcal{W} \mapsto 2^{\mathcal{D}}$ assigns to each $w \in \mathcal{W}$ a non-empty local domain set such that whenever $(w, v) \in \mathcal{R}$ we have $\delta(w) \subseteq \delta(v)$ and $\rho : (\mathcal{W} \times \mathcal{P}) \mapsto \bigcup_n 2^{\mathcal{D}^n}$ is the* valuation function, *which specifies the interpretation of predicates at every world over the local domain with appropriate arity. The model $\mathcal{M}$ is said to be a* constant domain model *if for all $w \in \mathcal{W}$ we have $\delta(w) = \mathcal{D}$.*

The monotonicity condition requiring $\delta(w) \subseteq \delta(v)$ for $(w, v) \in \mathcal{R}$ is required for evaluating the free variables present in the formula [11]. Because of this, the models are called *increasing domain models*.

For a given model $\mathcal{M}$ we denote $\mathcal{W}^{\mathcal{M}}, \mathcal{R}^{\mathcal{M}}$ etc to indicate the corresponding components. We simply use $\mathcal{W}, \mathcal{R}, \delta$ etc when $\mathcal{M}$ is clear from the context.

[1] Note that FOML can be translated into two-sorted FOL, and due to the Löwenheim–Skolem theorem for countable languages, every model has an equivalent countable model, cf. [5].

To evaluate formulas, we need an assignment function for variables. For a given model $\mathcal{M}$, an assignment function $\sigma : \mathsf{Var} \mapsto \mathcal{D}$ is *relevant* at $w \in \mathcal{W}$ if $\sigma(x) \in \delta(w)$ for all $x \in \mathsf{Var}$.

▶ **Definition 3** (FOML semantics). *Given an* FOML *model* $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$ *and* $w \in \mathcal{W}$, *and* σ *relevant at* w, *for all* FOML *formulas* α *define* $\mathcal{M}, w, \sigma \models \alpha$ *inductively as follows:*

$\mathcal{M}, w, \sigma \models P(x_1, \ldots, x_n)$	$\Leftrightarrow$	$(\sigma(x_1), \ldots, \sigma(x_n)) \in \rho(w, P)$
$\mathcal{M}, w, \sigma \models \neg\alpha$	$\Leftrightarrow$	$\mathcal{M}, w, \sigma \not\models \alpha$
$\mathcal{M}, w, \sigma \models \alpha \wedge \beta$	$\Leftrightarrow$	$\mathcal{M}, w, \sigma \models \alpha$ *and* $\mathcal{M}, w, \sigma \models \beta$
$\mathcal{M}, w, \sigma \models \exists x\alpha$	$\Leftrightarrow$	*there is some* $d \in \delta(w)$ *such that* $\mathcal{M}, w, \sigma_{[x \mapsto d]} \models \alpha$
$\mathcal{M}, w, \sigma \models \Box\alpha$	$\Leftrightarrow$	*for every* $u \in \mathcal{W}$ *if* $(w, u) \in \mathcal{R}$ *then* $\mathcal{M}, u, \sigma \models \alpha$

We sometimes write $\mathcal{M}, w \models \alpha(a)$ to mean $\mathcal{M}, w, [x \mapsto a] \models \alpha(x)$.

A formula α is *satisfiable* if there is some FOML structure $\mathcal{M}$ and $w \in \mathcal{W}$ and some assignment σ relevant at w such that $\mathcal{M}, w, \sigma \models \alpha$. In the sequel, we will only talk about the relevant σ for a given pointed model. Also, while evaluating α, it is enough to consider σ to be a partial function that gives an interpretation for the free variables of α. A formula α is *valid* if $\neg\alpha$ is not satisfiable.

2.1 Bundled fragments

The motivation for "bundling" is to restrict the occurrences of quantifiers using modalities. For instance, allowing only formulas of the form $\forall x \Box \alpha$ is one such bundling. We could also have $\Diamond \exists y \alpha$. Thus, there are many ways to "bundle" the quantifiers and modalities. We call these the "bundled operators/modalities". The following syntax defines all possible bundled operators of one quantifier and one modality:

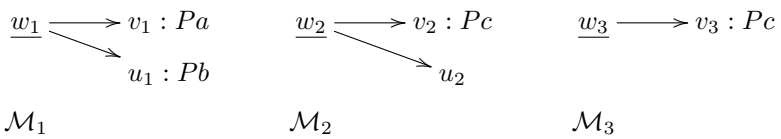
▶ **Definition 4** (Bundled-FOML syntax). *The bundled fragment of* FOML *is the set of all formulas constructed by the following syntax:*

$$\alpha ::= P(x_1, \ldots, x_n) \mid \neg\alpha \mid \alpha \wedge \alpha \mid \Box\alpha \mid \exists x \Box \alpha \mid \forall x \Box \alpha \mid \Box \exists x \alpha \mid \Box \forall x \alpha$$

where $P \in \mathcal{P}$ *has arity* n *and* $x, x_1, \ldots, x_n \in \mathsf{Var}$.

Note that the duals of the bundled operators give us the formulas of the form $\forall x \Diamond \alpha$, $\exists x \Diamond \alpha$, $\Diamond \forall x \alpha$, $\Diamond \exists x \alpha$. Also, note that $\Box\alpha$ can be defined using any one of the bundled operators where the quantifier is applied to a variable that does not occur in α. However, we retain $\Box\alpha$ in the syntax for technical convenience.

The following *constant domain* models may help to get familiar with bundles.



Let $\mathcal{D}^{\mathcal{M}_1} = \{a, b\}$, $\mathcal{D}^{\mathcal{M}_2} = \mathcal{D}^{\mathcal{M}_3} = \{c\}$. $\Box \exists x Px$ holds at w_1 and w_3 but not at w_2; $\exists x \Box Px$ holds only at w_3; $\neg \forall x \Box Px$ holds at w_1 and w_2; $\neg \Box \forall x \neg Px$ holds at all the w_i.

We denote $\forall\Box$-fragment to be the language that allows only atomic formulas, negation, conjunction, $\Box\alpha$ and $\forall x \Box \alpha$ (dually $\exists x \Diamond \alpha$) formulas, similarly for $\exists\Box$-fragment and so on. In general, these fragments are not equally expressive, e.g., as shown by [17], the $\exists\Box$-fragment cannot express $\Box\exists$, $\forall\Box$ and $\Box\forall$ bundles over models with increasing (or constant) domains.

In [14, 12] we proved that the $\forall\Box + \exists\Box$ and $\Box\forall + \Box\exists$ fragments are decidable over increasing domain models, while $\forall\Box$-fragment and $\Box\forall$-fragment are undecidable over constant domain models. Since simple bundles become undecidable over constant domain models, in this paper we focus on generalized bundles over increasing domain models.

Note that we can have more general bundled operators of the form $\forall x \forall y \Box \alpha$ etc. This naturally raises the question of what is the most general fragment of this form that is decidable. Towards this, we define a general fragment that subsumes all known decidable bundled fragments so far.

2.2 The Loosely Bundled Fragment

Note that a bundled formula of the form $\exists x \Box \alpha$ imposes a restriction that there is exactly one modal formula in the scope of $\exists x$. But this is a strong requirement. We weaken this condition to allow formulas of the form $\exists x \beta$ where β is a boolean combination of atomic formulas and modal formulas. Moreover, we can allow a quantifier alternation of the form $\exists x_1 \cdots \exists x_n \ \forall y_1 \cdots \forall y_m \ \beta$. As we will see, the fact that the existential quantifiers are outside the scope of universal quantifiers can help us to obtain decidability results over increasing domain models.

▶ **Definition 5** (LBF syntax). *The loosely bundled fragment of* FOML *is the set of all formulas constructed by the following syntax:*

$$\psi ::= P(z_1, \dots z_n) \mid \neg P(z_1, \dots z_n) \mid \psi \wedge \psi \mid \psi \vee \psi \mid \Box \alpha \mid \Diamond \alpha$$
$$\alpha ::= \psi \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \exists x_1 \dots \exists x_k \forall y_1 \dots \forall y_l \ \psi$$

where $k, l, n \geq 0$ and $P \in \mathcal{P}$ has arity n and $x_1, \dots x_k$, $y_1, \dots y_l$, $z_1, \dots, z_n \in$ Var.

Let LBF be the set of all formulas that can be obtained from the grammar of α above. Note that the syntax does not allow a quantifier alternation of the form $\forall x \exists y \ \alpha$. Also, inside the scope of quantifier prefix $\exists^* \forall^*$, we can only have boolean combinations of atomic and modal formulas. The $\exists^* \forall^*$ fragment in FO, (Bernays-Schönfinkel-Ramsey class) has a similar quantifier prefix structure but is different in spirit since there is no modality.

The loosely bundled fragment subsumes some of the combinations of bundled fragments. Hence proving the decidability for LBF implies the decidability for these combinations as well.

▶ **Proposition 6.** *The fragments $\forall \Box + \exists \Box$ and $\Box \forall + \Box \exists$ are subfragments of LBF.*

Note that many combinations of bundled operators (for instance $\Box \forall + \exists \Box$) do not form a subfragment of LBF.

3 Tableau Procedure

Note that the formulas of LBF are in negation normal form (where $\neg$ appears only in front of atomic formulas). We first define some useful terms and notations.

▶ **Definition 7.** *For any* FOML *formula φ:*

- *φ is a* literal *if φ is of the form $P(x_1, \dots x_n)$ or of the form $\neg P(x_1, \dots x_n)$*
- *φ is a* module *if φ is a* literal *or φ is of the form $\Delta \alpha$ where $\Delta \in \{\Box, \Diamond\}$*
- *The component of φ is defined inductively as follows:*
  - *If φ is a* module *then $C(\varphi) = \{\varphi\}$*
  - *If φ is of the form $\varphi_1 \wedge \varphi_2$ or $\varphi_1 \vee \varphi_2$ then $C(\varphi) = C(\varphi_1) \cup C(\varphi_2)$*
  - *If φ is of the form $\forall x \ \varphi_1$ or $\exists x \ \varphi_1$ then $C(\varphi) = \{\varphi\} \cup C(\varphi_1)$*
- *A formula φ is called* Existential-safe *if every $\psi \in C(\varphi)$ is a module or of the form $\forall x \ \psi'$. A finite set of formulas Γ is* Existential-safe *if every $\varphi \in \Gamma$ is* Existential-safe.

Intuitively, $\mathsf{C}(\varphi)$ is the set of all subformulas of φ that are "to be evaluated" at the current world. An Existential-safe formula φ does not need any witness from the current local domain in order to make the formula true. The notions of components and existential-safeness will play a role in the tableau-based decision procedure to be introduced below.

Before going into the specific tableau rules, we first explain the general method. A *tableau* is a tree-like structure generated from a single formula α by repeatedly applying a few rules with some auxiliary information as the root of the tree. Intuitively, a tableau for α is a pseudo model which can be transformed into a real model of α under some simple consistency conditions. We can then decide the satisfiability of a formula by trying to find a proper tableau. As in [17], a tableau T in our setting is a tree structure such that each node is a triple (w, Γ, σ) where w is a symbol or a finite sequence of symbols intended as the *name* of a possible world in the real model, Γ is a finite set of FOML-formulas, and σ is an assignment function for variables. Since we intend to use the set of variables as the domain in the tableau-induced real model, σ is simply a *partial* identity function on Var, i.e., $\sigma(x) = x$ for all $x \in Dom(\sigma) \subseteq \mathsf{Var}$, where the domain of σ, $Dom(\sigma)$, is intended to be the local domain of the real model. The intended meaning of the node (w, Γ, σ) is that all the formulas in Γ are satisfied on w with the assignment σ, thus we also write $(w : \Gamma, \sigma)$ for the triple.

A tableau rule specifies how the node in the premise of the rule is transformed to or connected with one or more new nodes given by the conclusion of the rule. Applying the rules can generate a tree-like structure, a tableau, which is saturated if every leaf node contains only literals. For any formula α, we refer to a saturated tableau of α simply as a tableau of α. Further, a saturated tableau is *open* if in every node $(w : \Gamma, \sigma)$ of the tableau, Γ does not contain both β and $\neg\beta$ for any formula β.[2]

We call a formula *clean* if no variable occurs both bound and free in it and every use of a quantifier quantifies a distinct variable. A finite set of formulas Γ is *clean* if $\bigwedge \Gamma$, the conjunction of all formulas in Γ, is clean. Note that every FOML-formula can be rewritten into an equivalent clean formula. For instance, the formulas $\exists x \Box Px \vee \forall x \Diamond Qx$ and $Px \wedge \Box \exists x Qx$ are not clean, whereas $\exists x \Box Px \vee \forall y \Diamond Qy$ and $Px \wedge \Box \exists y Qy$ are their clean equivalents respectively. Clean formulas help in handling the witnesses for existential formulas in the tableau in a syntactic way.

Consider a finite set of formulas Γ that is clean. Suppose we want to expand Γ to $\Gamma \cup \{\alpha_1, \ldots \alpha_k\}$, then even if each of α_i is clean, it is possible that a bound variable of α_i also occurs in some $\varphi \in \Gamma$ or another α_j. To avoid this, first, we rewrite the bound variables in each α_i one by one by using the fresh variables that do not occur in Γ and other previously rewritten α_j.

Such a rewriting can be fixed by always using the first fresh variable in a fixed enumeration of all the variables. When Γ and $\{\alpha_1, \ldots \alpha_k\}$ are clear from the context, we denote α_i^* to be such a fixed rewriting of α_i into a clean formula. It is not hard to see that the resulting finite set $\Gamma \cup \{\alpha_1^*, \ldots \alpha_k^*\}$ is clean.

3.1 Tableaux for LBF

The tableau rules for the LBF fragment are described in Fig. 1. The $(\wedge)$ and $(\vee)$ rules are standard, where we make a non-deterministic choice of one of the branches for $(\vee)$. The rule (END) says that if we are left with only modules and there are no $\Diamond$ formulas, then the branch does not need to be explored further. The $(\Diamond)$ rule creates one successor world for every $\Diamond$ formula at the current node and includes all the $\Box$ formulas that need to be

[2] Refer [17] for an illustration of a similar tableau construction.

$\dfrac{w:\varphi_1 \vee \varphi_2, \Gamma, \sigma}{w:\varphi_1, \Gamma, \sigma \mid\mid w:\varphi_2, \Gamma, \sigma}\ (\vee)$ $\quad \dfrac{w:\varphi_1 \wedge \varphi_2, \Gamma, \sigma}{w:\varphi_1, \varphi_2, \Gamma, \sigma}\ (\wedge)$	$\dfrac{w:\exists x\varphi,\ \Gamma,\ \sigma}{w:\varphi,\ \Gamma,\ \sigma'}(\exists)$ where $\sigma' = \sigma \cup \{(x,x)\}$
$\dfrac{w:\forall y\varphi,\ \Gamma,\ \sigma}{w:\ \{\varphi^*[z/y] \mid z \in Dom(\sigma)\},\ \Gamma,\ \sigma}(\forall)$ where Γ is Existential-safe and every $\varphi^*[z/y]$ is a clean rewriting of $\varphi[z/y]$ with respect to $\Gamma \cup \{\varphi[z/y] \mid z \in Dom(\sigma)\}$	
Given $n \geq 1$: and $m, s \geq 0$ $\dfrac{w:\Diamond\varphi_1,\ldots,\Diamond\varphi_n\ \ \Box\beta_1,\ldots\Box\beta_m\ \ l_1,\cdots,l_s,\ \ \sigma}{\langle wv_i:\varphi_i,\ \{\beta_j \mid j \in [1,m]\},\ \sigma\rangle\ \text{ for all } i \in [1,n]}\ (\Diamond)$	Given $m \geq 1, s \geq 0$: $\dfrac{w:\Box\beta_1,\ldots\Box\beta_m,\ l_1,\ldots,l_s,\sigma}{w:l_1,\cdots,l_s,\sigma}\ (\texttt{END})$

Figure 1 Tableau rules for LBF, here every l_i is a literal.

satisfied along with the $\Diamond$ formula and σ is inherited in the successor worlds to preserve the increasing domain property. The $(\exists)$ rule picks x itself as the witness to satisfy $\exists x\varphi$ and $(\forall)$ rule expands the set of formulas to include a clean version of $\varphi[z/y]$ for every variable z in the current local domain.

Note that only the $(\Diamond)$ rule can change (the name of) the possible world, thus creating a new successor. It simply extends the name w by new symbols v_i for each successor. Therefore there can be many nodes in the tableau sharing the same world name but such nodes form a path. Given w we use t_w to denote the *last node* sharing the first component w. Given a node $t = (w : \Gamma, \sigma)$ in a tableau, we use $Dom(t)$ to denote the domain of σ.

Also, there is an implicit ordering on how the rules are applied: $(\Diamond)$ rule can be applied at a node (w, Γ, σ) only if all formulas of Γ are modules and hence may be applied only after the $(\wedge, \vee, \forall, \exists)$ rules have been applied as many times as necessary at w. Similarly $(\forall)$ rule can be applied only when Γ is Existential-safe which means that the $(\exists)$ rule cannot be applied anymore at the current node.

▶ **Proposition 8.** *For every tableau T and every node $v = (w, \Gamma, \sigma)$ in T, if v is a leaf then either Γ contains only literals or there is some rule that can be applied at v.*

The proposition is true since the LBF ensures that if Γ is not Existential-safe and the $(\wedge, \vee, \Diamond, \texttt{END})$ rules cannot be applied, then it has to be the case that there is some $\exists x\varphi \in \Gamma$, for which we can apply the $(\exists)$ rule.

▶ **Theorem 9.** *For any clean LBF formula θ, let σ_r be an identity mapping over $FV(\theta) \cup \{z\}$ where z does not occur in θ. There is an open tableau T with root $(r : \{\theta\}, \sigma_r)$ iff θ is satisfiable in an increasing domain model.*

Proof. First, we claim that the rules preserve the cleanliness of the formulas. To see this, we verify that for every rule, if Γ in the antecedent of the rule is clean, then the Γ' obtained after the application of the rules is also clean. This is obvious for $(\wedge), (\vee), (\Diamond)$ and $(\texttt{END})$ rules. The $(\exists)$ rule preserves cleanliness because it frees variable x which is not bound by any other quantifier in the antecedent. The $(\forall)$ rule preserves cleanliness by rewriting.

($\Rightarrow$): Let T be an open tableau rooted at $(r : \{\theta\}, \sigma_r)$. Define a model $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$ as follows:

- $\mathcal{W} = \{w \mid (w : \Gamma, \sigma)$ is a node in $T\}$
- $\mathcal{D} = \mathsf{Var}$
- $\mathcal{R} = \{(w, v) \mid v$ is of the form wv' for some $v'\}$
- For every $w \in \mathcal{W}$, define $\delta(w) = Dom(t_w)$ where t_w is the last node of w in T
- For every $w \in \mathcal{W}$ and $p \in \mathcal{P}$, define $\rho(w, P) = \{\overline{x} \mid P\overline{x} \in \Gamma$ where $t_w = (w, \Gamma, \sigma)\}$

Clearly, $\mathcal{M}$ is an increasing domain model, and since $z \in Dom(\sigma_r)$, there is no empty local domain. As T is an open tableau, ρ is well-defined.

Claim. For every node $(w : \Gamma, \sigma)$ in T and for every LBF formula φ, if $\varphi \in \Gamma$ then $\mathcal{M}, w, \sigma \models \varphi$.

The claim is proved using a standard argument by induction on the height of the nodes of T from the leaves to the root (details in [13]). Thus, from the claim it follows that $\mathcal{M}, r, \sigma_r \models \theta$ since the label of the root of T is $(r : \{\theta\}, \sigma_r)$.

($\Leftarrow$) From Proposition 8 it follows that we can always apply some rule until every leaf node $(w : \Gamma, \sigma)$ is such that Γ contains only literals. Thus every (partial) tableau can be extended to a saturated tableau. To prove that such a tableau is open, it suffices to show that all rules preserve satisfiability (details in [13]). ◀

Note that the depth of the tableau is linear in the size of the formula. However, as we have to rewrite formulas using new variables when applying ($\forall$) rule, the size of the domain is exponential in the size of the formula. Hence, the tableau procedure can be implemented in ExpSpace.

▶ **Corollary 10.** *LBF is decidable in ExpSpace.*

4 The (Un)decidability Border

Note that the fragment LBF cannot express formulas of the form $\forall x \exists y \Box\ \alpha$ and also $\forall x \Box \forall y \Box \forall z \alpha$. There are many combinations of the bundled fragments that can express thesse formulas (like $\exists\Box + \Box\forall$ and $\forall\Box + \exists\Box + \Box\exists$ fragments). In fact, we can prove that a bundled fragment is undecidable if we can assert both $\forall x \exists y \Box \alpha$ and $\forall x \Box \forall y \Box \forall z\ \alpha$ in the fragment.

To prove this, we can use tiling encoding where $\forall x \exists y \Box \alpha$ can be used to assert that every "grid point" x has a horizontal/vertical successor y. In this case, it is important that both quantifiers are applicable over the same local domain and $\Box\alpha$ in $\forall x \exists y \Box \alpha$ ensures that the witness y acts uniformly across all the descendants. The second formula $\forall x \Box \forall y \Box \forall z\ \alpha$ is used to verify the "diagonal property" of the grid. In the companion technical report of this paper [13] we prove these results formally.

Also, note that there are fragments like $\exists\Box + \Box\exists$ where $\forall x \exists y \Box \alpha$ is expressible but not $\forall x \Box \forall y \Box \forall z\ \alpha$. In these cases, we can prove that such fragments do not have a finite model property. This is also proved in the companion technical report [13] where we show that this fragment gives a formula that can induce a linear order on the local domain of some world in the model and assert that this linear order does not have a maximal element.

This leaves us with the fragments that cannot express $\forall x \exists y \Box \alpha$ formulas and LBF is one such fragment which we proved to be decidable. The fragments $\forall\Box + \exists\Box$ and $\Box\forall + \Box\exists$ also fall in this category and since they are subfragments of LBF, decidability follows. So we *only* need to consider the fragment $\forall\Box + \exists\Box + \Box\forall$ to complete the terrain (cf. Table. 1).

5 The $\forall\Box + \Box\forall + \Box\exists$ Fragment

In this fragment, we are allowed to express $\forall x\Box\alpha$, $\Box\forall x\alpha$ and $\Box\exists x\alpha$ and their duals. Note that this fragment is not closed under subformulas. For instance, $\varphi := \forall x \left(\exists y\Diamond\alpha \vee \forall z\Box\beta\right)$ is a subformula of $\varphi' := \Diamond\forall x \left(\exists y\Diamond\alpha \vee \forall z\Box\beta\right)$. But φ' is in the fragment and φ is not in the fragment. We say that φ is a subformula of $\forall\Box + \Box\forall + \Box\exists$ if there is some formula $\varphi' \in \forall\Box + \Box\forall + \Box\exists$ such that φ is a subformula of φ'.

Note that even though we cannot express formulas of the form $\forall x\exists y\Box\varphi$ in the fragment, $\forall x\exists y\Diamond\varphi$ is still allowed. For instance, the formula $\Diamond\forall x \left(\exists y\Diamond\alpha\right)$ is in the fragment. Thus, we can have $\forall x\exists y\Diamond\varphi$ but not $\forall x\exists y\Box\varphi$. Intuitively this means that the different witnesses y for each x can work on *different* successor worlds. The fragment cannot enforce the interaction between x and y at all successors. This property can be used to prove that we can reuse the witnesses by creating new successor subtrees as required.

To get the decidability for $\forall\Box + \Box\forall + \Box\exists$ fragment, the main idea is to prove that the formulas of the form $\forall x\exists y\Diamond\varphi$ can be satisfied by picking some boundedly many witnesses y that will work for *all* x. This is the same as proving that if $\forall x\exists y\Diamond\varphi$ is satisfiable then $\exists y_1, \ldots \exists y_l \forall x\big(\bigvee \Diamond\varphi[y/y_i]\big)$ is satisfiable (where l is bounded). We illustrate the proof idea with an example.

▶ **Example 11.** Consider the formula $\alpha := \forall x\Big(\Box\neg Pxx \wedge \exists y\Diamond Pxy\Big)$ which is a subformula of the fragment. Let $\mathcal{T}, r \models \alpha$ where $\mathcal{T}$ is a tree model rooted at r. Now we will modify $\mathcal{T}$ to obtain $\mathcal{M}$ which is also a tree model rooted at r such that $\mathcal{M}, r \models \exists y_1\exists y_2\forall x \Big(\Box\neg Pxx \wedge \big(\Diamond Pxy_1 \vee \Diamond Pxy_2 \big)\Big)$.

The model $\mathcal{M}$ is obtained by extending $\mathcal{T}$ in the following way. Let $\delta^{\mathcal{T}}(r) = \mathcal{D}_r$. To obtain $\mathcal{M}$, first we extend the local domain of r by adding a fresh element a. The idea is that for every $d \in \mathcal{D}_r$ (when assigned to x) we will ensure that the new element a can be picked as the y-witness. To achieve this, we do the following: For every $d \in \mathcal{D}_r$ let $d' \in \mathcal{D}_r$ and $(r, s^d) \in \mathcal{R}^{\mathcal{T}}$ such that $\mathcal{T}, s^d \models Pdd'$. Let $\mathcal{T}^d$ be the subtree of $\mathcal{T}$ rooted at s^d. We will create a new copy of $\mathcal{T}^d$ and call its root u^d. Now, in the new subtree rooted at u^d, we make the new element a "behave" like d' and we add an edge from r to u^d. So, in particular, $\mathcal{M}$ will have $(r, u^d) \in \mathcal{R}^{\mathcal{M}}$ such that $\mathcal{M}, u^d \models Pda$. Since we do this construction for every $d \in \mathcal{D}_r$ we obtain that for all $d \in \mathcal{D}_r$ we have $\mathcal{M}, r \models \Diamond Pda$.

Now note that while evaluating α at $(\mathcal{M}, r)$ the $\forall x$ quantification will now also apply to a (since a is added to the local domain at r in $\mathcal{M}$). But then, we cannot use a itself as the witness for a since we also need to ensure that $\mathcal{M}, r \models \forall x\Box\neg Pxx$. Hence we will add another element b that acts as a witness for a. Further, b also needs a witness. But now we can choose a to be the witness for b since that does not violate the formula $\forall x\Box\neg Pxx$.

So to complete the construction, we pick some arbitrary $d \in \mathcal{D}_r$ for which we have some $d' \in \mathcal{D}_r$ and $(r, s^d) \in \mathcal{R}^{\mathcal{T}}$ such that $\mathcal{T}, s^d \models Pdd'$. We create two copies of $\mathcal{T}^d$ (subtree rooted at s^d) and call their roots as v^d and w^d respectively. In the subtree rooted at v^d we ensure that a and b "behave" like d, d' respectively and in the subtree rooted at w^d we ensure that a and b "behave" like d', d respectively. In particular, we have $\mathcal{M}, v^d \models Pab$ and $\mathcal{M}, w^d \models Pba$. Finally we add edges from r to v^d and from r to w^d in $\mathcal{M}$.

Thus, we have: $\delta^{\mathcal{M}}(r) = \delta^{\mathcal{T}}(r) \cup \{a, b\}$ and $\mathcal{M}, r \models \exists y_1\exists y_2\forall x \Big(\Box\neg Pxx \wedge \big(\Diamond Pxy_1 \vee \Diamond Pxy_2 \big)\Big)$. With the above construction, this assertion can be verified by assigning y_1 and y_2 to a and b respectively.

Note that in principle, it is possible for an $\exists$ quantified formula to occur in the scope of a $\forall$ quantifier as a boolean combination with other $\exists$ quantified formulas and modules. Moreover, these additional formulas can assert some "type" information that may force us to pick additional witnesses. For example, if the formula is

$$\forall x\Big[\Big(\Box(\neg Pxx \wedge Rx) \vee \Box(\neg Pxx \wedge \neg Rx)\Big) \wedge \exists y \Diamond\big(Rx \to (Pxy \wedge \neg Ry) \ \wedge \ \neg Rx \to (Pxy \wedge Ry)\big)\Big]$$

then we need two initial y-witnesses a_1, a_2 where one is used for witness whose "type" is $\Box(\neg Pyy \wedge Ry)$ and other for witness whose "type" is $\Box(\neg Pyy \wedge \neg Ry)$ and we also need the corresponding additional witnesses b_1, b_2. In general, the formula can force us to pick witnesses of a particular "1-type" which means we might need exponentially many witnesses.

Thus, we need to replace one $\exists$ inside the scope of $\forall$ by $2l$ many $\exists$ quantifiers outside the scope of $\forall$ where l is bounded exponentially in the size of the given formulas. We now prove this formally.

For any formula φ if $\alpha \in \mathsf{C}(\varphi)$ we denote this by $\varphi[\alpha]$. This means that α does not occur inside the scope of any modality in φ. Further, for every $\alpha \in \mathsf{C}(\varphi)$ and a formula β, we denote $\varphi[\beta/\alpha]$ obtained by rewriting φ where every occurrence of α in φ is replaced by β. In particular, we are interested in the case where α is of the form $\exists y \Diamond \psi$. Thus we always consider $\varphi[\exists y \Diamond \psi]$.

For every $l \geq 0$ if $\overline{y} = y_1, y_1' \ldots y_l, y_l'$ are fresh variables, we denote $\overline{y}\Diamond\psi$ to be the formula $\bigvee\limits_{i \leq l} \big(\Diamond\psi[y_i/y] \vee \Diamond\psi[y_i'/y]\big)$ which is a big disjunction where each disjunct replaces y in ψ with one of y_i or y_i'. Further, we denote $\varphi[\overline{y}\Diamond\psi/\exists y\Diamond\psi]$ as simply $\varphi[\overline{y}\Diamond\psi]$.

For instance, for the formula $\varphi := \Big(Px \ \vee \ \exists y \Diamond Qxy\Big)$ where $\psi := \exists y \Diamond Qxy$, for $l = 2$ and $\overline{y} = y_1, y_1', y_2, y_2'$ being fresh variables, $\varphi[\overline{y}\Diamond\psi]$ is given by: $\Big(Px \ \vee \big(\Diamond Qxy_1 \vee \Diamond Qxy_1' \vee \Diamond Qxy_2 \vee \Diamond Qxy_2'\big)\Big)$.

The size of a formula, denoted by $|\varphi|$, is the number of symbols occurring in φ and for a finite set of formulas Γ, let $|\Gamma| = \sum\limits_{\varphi \in \Gamma} |\varphi|$.

▶ **Lemma 12.** *Let Γ' be a clean finite set of formulas such that every $\alpha \in \Gamma$ is a subformula of $\forall\Box + \Box\forall + \Box\exists$ where $\Gamma' = \Gamma \cup \{\forall x \varphi[\exists y \Diamond \psi]\}$. If $\bigwedge \Gamma \wedge \forall x \varphi[\exists y \Diamond \psi]$ is satisfiable then there exists $l \leq 2^{|\Gamma'|}$ such that $\bigwedge \Gamma \ \wedge \ \exists y_1 \exists y_1' \exists y_2 \exists y_2' \ldots \exists y_l \exists y_l' \ \forall x \ \varphi[\overline{y}\Diamond\psi]$ is satisfiable, where $\overline{y} = y_1, y_1', \ldots y_l, y_l'$ are fresh variables.*

Note that the $\exists y$ quantifier is pulled outside the scope of the $\forall x$ quantifier and replaced with a bounded number of witnesses $y_1, y_1', y_2', y_2' \ldots y_l, y_l'$. Consequently $\Diamond \ \psi$ is replaced with a disjunction each replacing y with one of y_i or y_i' for every $i \leq l$.

To prove the lemma first we formally define the tree editing operation described in the example. Given a tree model $\mathcal{T}$ rooted at r, let $d \notin \mathcal{D}^{\mathcal{T}}$. To add the new domain element d to a local domain of r, we also need to specify the "type" of the new element d at r and its descendants. Towards this, we pick some domain element c that is already present in $\delta(r)$ and assign the type of d to the type of c at every world.

▶ **Definition 13.** *Given a tree model $\mathcal{T} = (\mathcal{W}, \mathcal{D}, \mathcal{R}, \delta, \rho)$ rooted at r, let $d \notin \mathcal{D}$ and $c \in \delta(r)$. Define the operation of "adding d to $\delta(r)$ by mimicking c", denoted by $\mathcal{T}_{d \mapsto c} = (\mathcal{W}, \mathcal{D}', \mathcal{R}, \delta', \rho')$ where:*

- $\mathcal{D}' = \mathcal{D} \cup \{d\}$
- *for all $w \in \mathcal{W}$ we have $\delta'(w) = \delta(w) \cup \{d\}$*

- *For every $w \in \mathcal{W}$ and predicate P define*
  $\rho'(w, P) = \{\overline{e'} \mid$ *there is some* $\overline{e} \in \rho(w, P)$ *and* $\overline{e'}$ *is obtained from* $\overline{e}$ *by replacing zero or more occurrences of* c *in* $\overline{e}$ *by* $d\}$.

Suppose that we want to extend the domain with $\overline{d} = d_1 \cdots d_n$ *which are fresh. Let* $\omega : \overline{d} \mapsto \mathcal{D}'$ *where* $\mathcal{D}' \subseteq \delta^{\mathcal{T}}(r)$ *and we want each* d_i *to mimic* $\omega(d_i)$. *Then we denote* $\mathcal{T}_\omega$ *to be the tree obtained by* $\left(\left(\mathcal{T}_{d_1 \mapsto \omega(d_1)}\right)_{d_2 \mapsto \omega(d_2) \ldots}\right)_{d_n \mapsto \omega(d_n)}$.

▶ **Proposition 14.** *Let* $\mathcal{T} = (\mathcal{W}, \mathcal{D}, \mathcal{R}, \delta, \rho)$ *be a tree model rooted at* r *with* $\mathcal{T}_{d \mapsto c}$ *being an extended tree where* $d \notin \mathcal{D}$ *and* $c \in \delta(r)$. *Then for all interpretations* σ *and for all* FOML *formulas* φ *and for all* $w \in \mathcal{W}$ *we have:*
$\mathcal{T}, w, \sigma_{[x \mapsto c]} \models \varphi$ *iff* $\mathcal{T}_{d \mapsto c}, w, \sigma_{[x \mapsto c]} \models \varphi$ *iff* $\mathcal{T}_{d \mapsto c}, w, \sigma_{[x \mapsto d]} \models \varphi$.

The proposition holds since there is no equality in the syntax and at every world in the extended model, the new element d "behaves" like c and the old elements "behave" like themselves (details in [13]). Now we are ready to prove Lemma 12.

Proof of Lemma 12. Let $\mathcal{T}$ be a tree model rooted at r such that $\mathcal{T}, r, \sigma \models \bigwedge \Gamma \wedge \forall x \varphi[\exists y \Diamond \psi]$.

For every domain element $a \in \delta^T(r)$ define:

$$\Pi(r, a) = \bigcup_{\forall x' \alpha \in \Gamma'} \{\lambda \mid \lambda \in \mathsf{C}(\alpha) \text{ and } T, r, \sigma_{[x' \mapsto a]} \models \lambda\}$$

Note that $\Pi(r, a)$ formalizes the notion of "type" of a at the world r. This includes the information of all subformulas that are true at the current world, when a universal variable is instantiated with a. Also, since the size of $|\mathsf{C}(\alpha)|$ is at most the size of Γ', the set $\Pi(r) = \{\Pi(r, a) \mid a \in \delta^T(r)\}$ has size $|\Pi(r)| = l$ where $l \leq 2^{|\Gamma'|}$. Enumerate $\Pi(r) = \{\Lambda_1, \ldots \Lambda_l\}$ and for every $i \leq l$ pick $a_i \in \delta^T(r)$ such that $\Pi(r, a_i) = \Lambda_i$. Now let $\overline{d} = d_1, d'_1, d_2, d'_2, \ldots d_l, d'_l$ be fresh domain elements and let $\omega : \overline{d} \mapsto \{a_1, a_2, \ldots a_l\}$ where for all $i \leq l$ we have $\omega(d_i) = \omega(d'_i) = a_i$. We define the required model $\mathcal{M} = (\mathcal{W}', \mathcal{D}', \mathcal{R}', \delta', \rho')$ as follows:

Let $\mathcal{M}_0 = \mathcal{T}_\omega$ be the new tree model rooted at r obtained by adding $d_1, d'_1, \ldots, d_l, d'_l$ to $\delta^T(r)$ where each d_i and d'_i mimics a_i. Now $\mathcal{M}$ is obtained by extending $\mathcal{M}_0$ as follows:

For every $c \in \delta(r)$ such that $\mathcal{T}, r, [x \mapsto c] \models \exists y \Diamond \psi$ we pick $c' \in \delta^T(r)$ and $r \to s^c$ be such that $\mathcal{T}, s^c, [xy \mapsto cc'] \models \psi$. Let $\mathcal{T}^c$ be the sub-tree of $\mathcal{T}$ rooted at s^c and $\Pi(r, c') = \Lambda_j$. Create a new subtree $\mathcal{T}_0^c = \mathcal{T}_{\omega'}^c$ where for all $h \neq j$ we have $\omega'(d_h) = \omega'(d'_h) = a_h$ and $\omega'(d_j) = \omega'(d'_j) = c'$. Let u^c be the root of $\mathcal{T}_0^c$. Add an edge from r to u^c in $\mathcal{M}$.
Further, for every $i \leq l$ if $\mathcal{T}, r, \sigma_{[x \mapsto a_i]} \models \exists y \Diamond \psi$ then let $b \in \delta^T(r)$ and $r \to s^i \in \mathcal{R}^T$ be such that $\mathcal{T}, s^i, \sigma_{[xy \mapsto a_i b]} \models \psi$. Let $\Pi(r, b) = \Lambda_j$. Then create $\mathcal{T}_1^i = \mathcal{T}_{\omega_1}^i$ and $\mathcal{T}_2^i = \mathcal{T}_{\omega_2}^i$ where ω_1 and ω_2 are defined as follows:

- For all $h \neq j$, $\omega_1(d_h) = \omega_1(d'_h) = \omega_2(d_h) = \omega_2(d'_h) = a_h$
- $\omega_1(d_j) = a_j$ and $\omega_1(d'_j) = b$
- $\omega_2(d_j) = b$ and $\omega_2(d'_j) = a_j$

Let v^i and w^i be the root of $\mathcal{T}_1^i$ and $\mathcal{T}_2^i$ respectively. Add the edges from r to v^i and from r to w^i in $\mathcal{M}$. The two copies of subtrees are intended to provide witnesses for $\exists y \Diamond \psi$ for d_j and d'_j respectively. We need the two copies to ensure that ω_1 and ω_2 are well defined in the case when $i = j$ and $a_j \neq b$.

We now explain the idea behind the construction. Note that $\mathcal{T}_0^c$ rooted at u^c is created for every $c \in \delta^T(r)$ such that $\mathcal{T}, r, \sigma_{[x \mapsto c]} \models \exists y \Diamond \psi$. If c' is the picked witness for c with $(r, s^c) \in \mathcal{R}^T$ such that $\mathcal{T}, s^c, \sigma_{[xy \mapsto cc']} \models \psi$ and $\Pi(r, c') = \Lambda_j$ then by construction, $\mathcal{T}_0^c$ is

rooted u^c where d_j mimics c' in the subtree rooted at u^c. All these together indicate that we can use d_j and the subtree rooted at u^c in $\mathcal{M}$ to verify that $\mathcal{M}, u^c, \sigma_{[xy \mapsto cd_j]} \models \psi$. Also note that for all $h \neq j$ the fresh elements d_h and d'_h mimic a_h in the subtree $\mathcal{T}_0^c$ (i.e, we have not added any extra "types").

Further, we want the type of d_i and d'_i at r in $\mathcal{M}$ to mimic the type of a_i at r in $\mathcal{T}$. All type information is taken care of in $\mathcal{M}_0$ where both d_i and d'_i mimic a_i except the formula $\exists y \Diamond \psi$. So if $\mathcal{T}, r, \sigma_{[x \mapsto a_i]} \models \exists y \Diamond \psi$ then we need a witness to verify $\mathcal{M}, r, \sigma_{[x \mapsto d_i]} \models \exists y \Diamond \psi$ and $\mathcal{M}, r, \sigma_{[x \mapsto d'_i]} \models \exists y \Diamond \psi$. If the witness for y for a_i is b and $\Pi(r, b) = \Lambda_j$ then we want d'_j to be the witness for d_i and d_j to be the witness for d'_i.

Consequently if s^i is the world such that $a \to s^i \in \mathcal{R}^T$ and $\mathcal{T}, s^i, [xy \mapsto a_i b] \models \psi$ then we create two new copies of subtree $\mathcal{T}^i$ rooted at s^i and call it $\mathcal{T}_1^i$ and $\mathcal{T}_2^i$. By construction, in particular, the new element d_i mimics a_i and d'_j mimics b in $\mathcal{T}_1^i$. Similarly d'_i mimics a_i and d_j mimics b in $\mathcal{T}_2^i$. Thus, we can pick d'_j to be the witness for d_i (and consider $\mathcal{T}_1^i$) and pick d_j to be the witness for d'_i (and consider $\mathcal{T}_2^i$).

Also, it is important to note that for every $r \to v \in \mathcal{R}^M$, if d_i mimics c and d'_i mimics c' at v then we will always have $\Pi(r, c) = \Pi(r, c') = \Lambda_i = \Pi(r, a_i)$. Now it can be verified that $\mathcal{M}, r, \sigma \models \bigwedge \Gamma \ \wedge \ \exists y_1 \exists y'_1 \ldots \exists y_l \exists y'_l \ \forall x \varphi[\overline{y} \Diamond \psi]$.

The details are provided in [13]. ◀

▶ **Corollary 15.** *Let Γ' be a clean finite set of formulas such that every $\alpha \in \Gamma$ is a subformula of $\forall\Box + \Box\forall + \Box\exists$ where $\Gamma' = \Gamma \cup \{\forall x \varphi[\exists y \Diamond \psi]\}$. If $\bigwedge \Gamma \wedge \forall x \varphi[\exists y \Diamond \psi]$ is satisfiable then $\bigwedge \Gamma \wedge \exists y_1 \exists y'_1 \exists y_2 \exists y'_2 \ldots \exists y_l \exists y'_l \ \forall x \ \varphi[\overline{y} \Diamond \psi]$ is satisfiable, where $l = 2^{|\Gamma'|}$ and $\overline{y} = y_1, y'_1, \ldots y_l, y'_l$ are fresh variables.*

To see why the corollary is true, by Lemma 12 we get some $l \leq 2^{|\Gamma'|}$, and we can pad sufficiently many dummy variables to get a strict equality. This gives us a useful tableau rule which we call $(\forall\exists\Diamond)$ rule for $\forall\Box + \Box\forall + \Box\exists$ fragment, described in Fig. 2. The full tableau rules for $\forall\Box + \Box\forall + \Box\exists$ is given by the tableau rules of `LBF` (Fig. 1) along with the $(\forall\exists\Diamond)$-rule.

$$\frac{w : \forall x \ \varphi[\exists y \Diamond \psi], \ \Gamma, \ \sigma}{w : \forall x \ \varphi[\overline{y} \Diamond \psi], \ \Gamma, \ \sigma'} (\forall\exists\Diamond)$$

where $l = 2^{|\Gamma| + |\varphi|}$ and $\overline{y} = y_1, y'_1, \ldots y_l, y'_l$ are fresh variables and $\sigma' = \sigma \cup \{(y_i, y_i) \ , (y'_i, y'_i) \mid i \leq l\}$

Figure 2 (The $\forall\exists\Diamond$) rule for $\forall\Box + \Box\forall + \Box\exists$ fragment.

▶ **Theorem 16.** *For any clean $\forall\Box + \Box\forall + \Box\exists$ formula θ, let σ_r be an identity mapping over $FV(\theta) \cup \{z\}$ where z does not occur in θ. There is an open tableau with $(r : \{\theta\}, \sigma_r)$ as the root iff θ is satisfiable in an increasing domain model.*

The proof follows along the lines of Theorem 9. The only interesting part of the proof is to show that the $(\forall\exists\Diamond)$ rule preserves satisfiability and this is by Lemma 12(details in [13]).

Note that at if we start with a formula of length n then the application of $(\forall\exists\Diamond)$ rule will blow up the formula to size 2^n. So we have a tableau procedure that can be implemented as an algorithm in ExpSpace.

▶ **Corollary 17.** *The fragment $\forall\Box + \Box\forall + \Box\exists$ is decidable in ExpSpace.*

6 Conclusion

In this paper, we have studied the decidability of bundled fragments of FOML, where we have no restrictions on the use of variables or arity of relations. Specifically, we proved the decidability of the loosely bundled fragment LBF and the $\forall\Box + \Box\forall + \Box\exists$ fragment. The decidability of these fragments hinges on the observation that $\forall x \exists y \Box \alpha$ is not expressible. A NexpTime lower bound follows for $\forall\Box + \exists\Box$ and $\Box\forall + \Box\exists$ (via encoding the corresponding version of the tiling problem [13]), which implies the same lower bound for LBF and $\forall\Box + \Box\forall + \Box\exists$. There is a significant gap between the upper and lower complexity bounds and we need sharper technical tools for investigating lower bounds for bundled fragments.

Note that the quantifier prefix in LBF is of the form $\exists^*\forall^*$ and hence any extension of this quantifier prefix or extending LBF with negation closure will result in a fragment that will be able to express $\forall x \exists y \Box\ \alpha$ (and hence will not have the finite model property, [13]). In this sense, LBF is the largest fragment in which $\forall x \exists y \Box \alpha$ is not (syntactically) expressible. For the fragment $\forall\Box + \Box\forall + \Box\exists$ we introduced a technique to pull out $\exists$ quantifiers outside the immediate scope of $\forall$ and obtain a finite model property. This technique may be useful in studying other fragments of first-order modal logic.

The results in the paper, along with those in [13], provide a trichotomy classification of combinations of bundled operators over increasing domain models (Table 1). The fragments in which we can express both $\forall x \exists y \Box \alpha$ and $\forall x \Box \forall y \Box \forall z \beta$ are undecidable, fragments in which we can express the former but not the latter lack finite model property (but decidability is open) and fragments where we cannot express the former are decidable. Similar trichotomy can also be proved for satisfiability over constant domain models [13].

We have considered only the "pure" fragments, without constants, function symbols, or equality. The addition of constants is by itself simple, but equality complicates things considerably. Since equality is extensively used in specifications, mapping fragments with equality is an important direction. The study of bundles over models with various frame conditions is also relevant for applications. Unfortunately, while it is clear that equivalence frames lead to undecidability [17], even with transitive frames the situation is unclear. Obtaining good decidable fragments over linear frames is an important challenge.

In the context of verification of infinite-state systems, we are often more interested in the model checking problem than in satisfiability. If the domain is finite, the problem is no different from model checking of first-order modal logic. However, we are usually interested in the specification being checked against a finitely specified (potentially infinite) model, e.g., when the domain elements form a *regular* infinite set. This is a direction to be pursued in the context of bundled fragments.

We have presented tableau-based decision procedures that are easily implementable, but inference systems for reasoning in these logics require further study.

References

1 Hajnal Andréka, István Németi, and Johan van Benthem. Modal languages and bounded fragments of predicate logic. *J. Philos. Log.*, 27(3):217–274, 1998. doi:10.1023/A:1004275029985.

2 Francesco Belardinelli and Alessio Lomuscio. Quantified epistemic logics for reasoning about knowledge in multi-agent systems. *Artif. Intell.*, 173(9-10):982–1013, 2009. doi:10.1016/j.artint.2009.02.003.

3 Francesco Belardinelli and Alessio Lomuscio. Interactions between knowledge and time in a first-order logic for multi-agent systems: Completeness results. *J. Artif. Intell. Res.*, 45:1–45, 2012. doi:10.1613/jair.3547.

4 Egon Börger, Erich Grädel, and Yuri Gurevich. *The classical decision problem*. Springer Science & Business Media, 2001.

5 Torben Braüner and Silvio Ghilardi. First-order modal logic. In P. Blackburn, J. van Benthem, and F. Wolter, editors, *Handbook of Modal Logic*, pages 549–620. Elsevier, 2007.

6 Clare Dixon, Michael Fisher, Boris Konev, and Alexei Lisitsa. Practical first-order temporal reasoning. In *Proceedings of TIME 2008*, pages 156–163, 2008. doi:10.1109/TIME.2008.15.

7 Harald Ganzinger, Christoph Meyer, and Margus Veanes. The two-variable guarded fragment with transitive relations. In *Proceedings of LICS '99*, pages 24–34, 1999. doi:10.1109/LICS.1999.782582.

8 Ian Hodkinson, Frank Wolter, and Michael Zakharyaschev. Decidable fragment of first-order temporal logics. *Ann. Pure Appl. Log.*, 106(1-3):85–134, 2000. doi:10.1016/S0168-0072(00)00018-X.

9 Ian Hodkinson, Frank Wolter, and Michael Zakharyaschev. Monodic fragments of first-order temporal logics: 2000-2001 A.D. In Robert Nieuwenhuis and Andrei Voronkov, editors, *Proceedings of LPAR '01*, volume 2250, pages 1–23. Springer, 2001. doi:10.1007/3-540-45653-8_1.

10 Ian Hodkinson, Frank Wolter, and Michael Zakharyaschev. Decidable and undecidable fragments of first-order branching temporal logics. In *Proceedings LICS 2002*, pages 393–402, 2002. doi:10.1109/LICS.2002.1029847.

11 G. E. Hughes and M. J. Cresswell. *A New Introduction to Modal Logic*. Routledge, 1996.

12 Mo Liu. On the decision problems of some bundled fragments of first-order modal logic. Master's thesis, Peking University, 2019. URL: https://arxiv.org/abs/2201.02336.

13 Mo Liu, Anantha Padmanabha, Ramaswamy Ramanujam, and Yanjing Wang. Are bundles good deals for FOML? *CoRR*, abs/2202.01581, 2022. arXiv:2202.01581.

14 Anantha Padmanabha, R Ramanujam, and Yanjing Wang. Bundled Fragments of First-Order Modal Logic: (Un)Decidability. In *Proceedings of FSTTCS 2018*, volume 122, pages 43:1–43:20, 2018. doi:10.4230/LIPIcs.FSTTCS.2018.43.

15 Mikhail N. Rybakov and Dmitry Shkatov. Undecidability of first-order modal and intuitionistic logics with two variables and one monadic predicate letter. *Stud Logica*, 107(4):695–717, 2019. doi:10.1007/s11225-018-9815-7.

16 Moshe Y Vardi. Why is modal logic so robustly decidable? Technical report, Rice University, 1997.

17 Yanjing Wang. A new modal framework for epistemic logic. In *Proceedings of TARK 2017*, pages 515–534, 2017. doi:10.4204/EPTCS.251.38.

18 Yanjing Wang. Beyond Knowing That: A New Generation of Epistemic Logics. In *Outstanding Contributions to Logic*, volume 12, pages 499–533. Springer Nature, 2018. doi:10.1007/978-3-319-62864-6_21.

19 Frank Wolter and Michael Zakharyaschev. Decidable fragments of first-order modal logics. *J. Symb. Log.*, 66(3):1415–1438, 2001. URL: http://www.jstor.org/stable/2695115.