The complexity of one-agent Refinement Modal Logic

Laura Bozzelli[1], Hans van Ditmarsch[2], and Sophie Pinchinat[3]

[1] Technical University of Madrid (UPM), Madrid, Spain, laura.bozzelli@fi.upm.es
[2] Logic, University of Sevilla, Spain, hvd@us.es & IMSc, Chennai, India
[3] IRISA/INRIA, University of Rennes, Sophie.Pinchinat@irisa.fr

Abstract. We investigate the complexity of satisfiability for one-agent Refinement Modal Logic (RML), a known extension of basic modal logic (ML) obtained by adding refinement quantifiers on structures. It is known that RML has the same expressiveness as ML, but the translation of RML into ML is of non-elementary complexity, and RML is at least *doubly* exponentially more succinct than ML. In this paper, we show that RML-satisfiability is 'only' *singly* exponentially harder than ML-satisfiability, the latter being a well-known PSPACE-complete problem. More precisely, we establish that RML-satisfiability is complete for the complexity class $\mathsf{AEXP}_{\mathsf{pol}}$, i.e., the class of problems solvable by alternating Turing machines running in single exponential time but only with a polynomial number of alternations (note that $\mathsf{NEXPTIME} \subseteq \mathsf{AEXP}_{\mathsf{pol}} \subseteq \mathsf{EXPSPACE}$).

1 Introduction

From propositional to refinement quantification in modal logics. Modal logics augmented with propositional quantifiers, which allow 'dynamic' model transformations (and, in particular, model restrictions), have been widely investigated in the literature starting from the seminal paper by Fine [8]. Fine distinguishes different propositional quantifications, which allow different kinds of model transformations, not all of which are, in our modern terms, bisimulation preserving. However, in the general case, propositional quantification can easily lead to undecidable logics [8, 9]. This has motivated, more recently, the introduction of bisimulation quantified logics [21, 11, 9, 16]: in this framework, the quantification is over the models which are bisimilar to the current model except for a propositional variable p (note that this includes model restriction). A novel way of quantification in rather dynamic modal logics is quantifying over all modally definable submodels of a given model [1]. The setting for these logics is how to quantify over information change; for example, in the logic APAL of [1], an expression that we might write as $\exists.\varphi$ for our purposes stands for "there is a formula ψ such that after model restriction with relativization to ψ, the formula φ holds". *Refinement modal logic* (see [19, 20] and the unpublished manuscript [3]) is a generalization of this perspective to more complex model transformations than mere model restrictions: this is achieved by existential and universal quantifiers which range over the *refinements* of the current structure (model). Unlike simulation, its dual, refinement corresponds to structural loss instead of structural gain, and it is more general than model restriction, since it is equivalent to bisimulation followed by model restriction [3]. Refinement quantification can be seen as implicit quantification over propositional variables, just as in bisimulation quantified logics we have explicit quantification over propositional

variables; in fact, it is equivalent to bisimulation quantification followed by relativization [3]. As amply illustrated in [3], refinement quantification has applications in many settings: in logics for games [17, 16], it may correspond to a player discarding some moves; for program logics [10], it may correspond to operational refinement; and for logics for spatial reasoning, it may correspond to sub-space projections [15].

Our contribution. We focus on complexity issues for (one-agent) Refinement Modal Logic (RML) [19, 20, 3], the extension of (one-agent) basic modal logic (ML) obtained by adding the existential and universal refinement quantifiers $\exists_r$ and $\forall_r$. It is known [20, 3] that RML has the same expressivity as ML, but the translation of RML into ML is of non-elementary complexity and no elementary upper bound is known for its satisfiability problem [3]. In fact, an upper bound in 2EXPTIME has been claimed in [20] by a tableaux-based procedure: the authors later concluded that the procedure is sound but not complete [3]. In this paper, our aim is to close that gap. We also investigate the complexity of satisfiability for some equi-expressive fragments of RML. In particular, we associate with each RML formula φ a parameter $\Upsilon_w(\varphi)$ corresponding to a slight variant of the classical quantifier alternation depth (measured w.r.t. $\exists_r$ and $\forall_r$), and for each $k \geq 1$, we consider the fragment RML^k consisting of the RML formulas φ such that $\Upsilon_w(\varphi) \leq k$. Moreover, we consider the existential (resp., universal) fragment $\mathsf{RML}^{\exists}$ (resp., $\mathsf{RML}^{\forall}$) obtained by disallowing the universal (resp., existential) refinement quantifier.

In order to present our results, first, we recall some computational complexity classes. We assume familiarity with the standard notions of complexity theory [12, 14]. We will make use of the levels Σ_k^{EXP} ($k \geq 1$) of the exponential-time hierarchy EH, which are defined similarly to the levels Σ_k^{P} of the polynomial-time hierarchy PH, but with NP replaced with NEXPTIME. In particular, Σ_k^{EXP} corresponds to the class of problems decided by single exponential-time bounded Alternating Turing Machines (ATM, for short) with at most $k-1$ alternations and where the initial state is existential [12]. Note that Σ_1^{EXP} = NEXPTIME. Recall that EH $\subseteq$ EXPSPACE and EXPSPACE corresponds to the class of problems decided by single exponential-time bounded ATM (with no constraint on the number of alternations) [4]. We are also interested in an intermediate class between EH and EXPSPACE, here denoted by $\mathsf{AEXP}_{\mathsf{pol}}$, that captures the precise complexity of some relevant problems [7, 12, 18] such as the first-order theory of real addition with order [7, 12]. Formally, $\mathsf{AEXP}_{\mathsf{pol}}$ is the class of problems solvable by single exponential-time bounded ATM with a polynomial-bounded number of alternations.

Our complexity results are summarized in Figure 1 where we also recall the well-known complexity of ML-satisfiability. For the upper bounds, the (technically non-trivial) main step in the proposed approach exploits a "small" size model property: we establish that like basic modal logic ML, RML enjoys a single exponential size model property.

We conclude this section by observing that our results are surprising for the following reasons. While our results essentially indicate that satisfiability of RML is "only" *singly* exponentially harder than satisfiability of ML, it is known [3] that RML is *doubly* exponentially more succinct than ML. Moreover, RML can be *doubly* exponentially more succinct than the more expressive logic given by the modal μ-calculus [3] (recall that satisfiability of modal μ-calculus is EXPTIME-complete [6]). Furthermore, satisfiability of RML extended with the universal and existential eventually modalities, and their duals, of standard CTL [5] is already non-elementarily decidable [3]. Due to lack of space, many proofs are omitted and can be found in the Appendix.

ML	$\mathsf{RML}^{\exists} = \mathsf{RML}^{1}$	$\mathsf{RML}^{\forall} \subseteq \mathsf{RML}^{2}$	RML^{k+1} $(k \geq 1)$	RML
PSPACE-complete	$\in$ NEXPTIME PSPACE-hard	$\in \Sigma_2^{\mathsf{EXP}}$ NEXPTIME-hard	$\in \Sigma_{k+1}^{\mathsf{EXP}}$ Σ_k^{EXP}-hard	$\mathsf{AEXP}_{\mathsf{pol}}$-complete

Fig. 1. Complexity results for satisfiability of RML and RML-fragments

2 Preliminaries

In the rest of this section, we fix a finite set P of atomic propositions.

Structures, tree structures, and refinement preorder. A (*one-agent Kripke*) *structure* (*over* P) is a tuple $M = \langle S, E, V \rangle$, where S is a set of states (or worlds), $E \subseteq S \times S$ is a transition (or accessibility) relation, and $V : S \mapsto 2^P$ is a P-*valuation* assigning to each state s the set of propositions in P which hold at s. For states s and t of M such that $(s,t) \in E$, we say that t is a *successor* of s. A *pointed* structure is a pair (M,s) consisting of a structure M and a designated initial state s of M.

A tree T is a prefix-closed subset of $\mathbb{N}^*$, where $\mathbb{N}$ is the set of natural numbers. The elements of T are called *nodes* and the empty word ε is the *root* of T. For $x \in T$, the set of children (or successors) of x is $\{x \cdot i \in T \mid i \in \mathbb{N}\}$. The *size* $|T|$ of T is the number of T-nodes. A (*rooted*) *tree structure* (over P) is a pair $\langle T, V \rangle$ such that T is a tree and $V : T \mapsto 2^P$ is a P-valuation over T. For $x \in T$, the *tree substructure of* $\langle T, V \rangle$ *rooted at* x is the tree structure $\langle T_x, V_x \rangle$, also denoted by $\langle T, V \rangle_x$, where $T_x = \{y \in \mathbb{N}^* \mid x \cdot y \in T\}$ and $V_x(y) = V(x \cdot y)$ for all $y \in T_x$. Note that a tree structure $\langle T, V \rangle$ corresponds to the pointed structure $(\langle T, E, V \rangle, \varepsilon)$, where $(x,y) \in E$ iff y is a child of x. Moreover, we can associate with any pointed structure (M,s) a tree structure, denoted by $\mathit{Unw}(M,s)$, obtained by unwinding M from s in the usual way.

For two structures $M = \langle S, E, V \rangle$ and $M' = \langle S', E', V' \rangle$, a *refinement from* M *to* M' is a relation $\mathfrak{R} \subseteq S \times S$ such that for all $(s,s') \in \mathfrak{R}$: (i) $V(s) = V'(s')$, and (ii) if $(s',t') \in E'$ for some $t' \in S'$, then there is some state $t \in S$ such that $(s,t) \in E$ and $(t,t') \in \mathfrak{R}$. If, additionally, the inverse of $\mathfrak{R}$ is a refinement from M' to M, then $\mathfrak{R}$ is a *bisimulation* from M to M'. For states $s \in S$ and $s' \in S'$, (M',s') is a *refinement* of (M,s) (resp., (M,s) and (M',s') are *bisimilar*), written $(M,s) \succcurlyeq (M',s')$ (resp., $(M,s) \approx (M',s')$), if there is a refinement (resp., bisimulation) $\mathfrak{R}$ from M to M' s.t. $(s,s') \in \mathfrak{R}$. Note that $\succcurlyeq$ is a preorder (i.e., reflexive and transitive) and $\approx$ is an equivalence relation over pointed structures.

Remark 1. For each pointed structure (M,s), $(M,s) \approx \mathit{Unw}(M,s)$.

Refinement Modal Logic. We recall the syntax and semantics of one-agent *refinement modal logic* (RML) [20, 3], an equally expressive extension of basic modal logic [2] obtained by adding the *existential and universal refinement quantifiers*. For technical convenience, the syntax of RML formulas φ over P is given in *positive form* as:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Diamond\varphi \mid \Box\varphi \mid \exists_r \varphi \mid \forall_r \varphi$$

where $p \in P$, $\Diamond\varphi$ reads as "possibly φ", $\Box\varphi$ reads as "necessarily φ", and $\exists_r$ and $\forall_r$ are the existential and universal refinement quantifiers. The *dual* $\widetilde{\varphi}$ of a RML formula φ is inductively defined as: $\widetilde{p} = \neg p$, $\widetilde{\neg p} = p$, $\widetilde{\varphi \vee \psi} = \widetilde{\varphi} \wedge \widetilde{\psi}$, $\widetilde{\Diamond\varphi} = \Box\widetilde{\varphi}$, $\widetilde{\Box\varphi} = \Diamond\widetilde{\varphi}$, $\widetilde{\exists_r \varphi} = \forall_r \widetilde{\varphi}$, and $\widetilde{\forall_r \varphi} = \exists_r \widetilde{\varphi}$. The size $|\varphi|$ of a formula φ is the number of distinct subformulas of φ.

RML is interpreted over pointed structures (M,s). The satisfaction relation $(M,s) \models \varphi$ is inductively defined as follows (we omit the clauses for boolean connectives):

$$
\begin{array}{ll}
(M,s) \models p & \texttt{iff} \quad p \in V(s) \text{ where } M = \langle S,E,V \rangle \\
(M,s) \models \Diamond\varphi & \texttt{iff} \quad \text{for some successor } t \text{ of } s \text{ in } M, (M,t) \models \varphi \\
(M,s) \models \Box\varphi & \texttt{iff} \quad \text{for all successors } t \text{ of } s \text{ in } M, (M,t) \models \varphi \\
(M,s) \models \exists_r\varphi & \texttt{iff} \quad \text{for some refinement } (M',s') \text{ of } (M,s), (M',s') \models \varphi \\
(M,s) \models \forall_r\varphi & \texttt{iff} \quad \text{for all refinements } (M',s') \text{ of } (M,s), (M',s') \models \varphi
\end{array}
$$

Note that $(M,s) \models \varphi$ iff $(M,s) \not\models \widetilde{\varphi}$. If $(M,s) \models \varphi$, we say that (M,s) *satisfies* φ, or also that (M,s) is a *model* of φ. A RML formula φ is *satisfiable* if φ admits some model.

Fragments of RML. Let ML be the fragment of RML obtained by disallowing the refinement quantifiers, which corresponds to basic modal logic [2], and $\mathsf{RML}^{\forall}$ and $\mathsf{RML}^{\exists}$ be the fragments of RML obtained by disallowing the existential refinement quantifier and the universal refinement quantifier, respectively. Moreover, we introduce a family $\{\mathsf{RML}^k\}_{k\geq 1}$ of RML-fragments, where RML^k consists of the RML formulas whose *weak refinement quantifier alternation depth* (see Definition 1 below) is at most k.

Definition 1 (Weak Refinement Quantifier Alternation Depth). We first define the *weak alternation length* $\ell(\chi)$ of finite sequences $\chi \in \{\exists_r, \forall_r\}^*$ of refinement quantifiers: $\ell(\varepsilon) = 0$, $\ell(Q) = 1$ for every $Q \in \{\exists_r, \forall_r\}$, and $\ell(QQ'\chi)$ is $\ell(Q'\chi)$ if $Q = Q'$, and $\ell(Q'\chi)+1$ otherwise. For a RML formula φ, let $T(\varphi)$ be the standard tree encoding of φ, where each node is labeled by either a modality, or a boolean connective, or an atomic proposition. The *weak refinement quantifier alternation depth* $\Upsilon_w(\varphi)$ of a RML formula φ is the maximum of the alternation lengths $\ell(\chi)$ where χ is the sequence of refinement quantifiers along a path of $T(\exists_r\varphi)$ (note that we consider $T(\exists_r\varphi)$ and not $T(\varphi)$).

As an example, for $\varphi = (\forall_r\exists_r p) \vee \Box(\exists_r(p \wedge \forall_r q))$, $\Upsilon_w(\varphi) = 3$. Note that $\mathsf{RML}^{\exists} = \mathsf{RML}^1$ and $\mathsf{RML}^{\forall} \subseteq \mathsf{RML}^2$. Moreover, for each RML formula φ, $\Upsilon_w(\forall_r\varphi) = \Upsilon_w(\widetilde{\forall_r\varphi}) + 1$. The following example illustrates the succinctness of $\mathsf{RML}^{\exists}$ w.r.t. ML.

Example 1. For $n \geq 1$, a n-block is a sequence $b_1, \ldots, b_{n+1}$ of $n+1$ bits. The following $\mathsf{RML}^{\exists}$ formula φ_n is satisfied by a tree structure iff there are two paths from the root encoding two n-blocks of the form $b_1, \ldots, b_n, b_{n+1}$ and $b_1, \ldots, b_n, b'_{n+1}$ s.t. $b_{n+1} \neq b'_{n+1}$:

$$\varphi_n := \exists_r\big(\Diamond^{n+1}(0 \wedge \neg 1) \wedge \Diamond^{n+1}(1 \wedge \neg 0) \wedge \bigwedge_{i=1}^{n} \bigvee_{b\in\{0,1\}} \Box^i(b \wedge \neg(1-b))\big)$$

By using the approach in Section 6.2 of [3], one can easily show that any ML formula which is equivalent to φ_n has size singly exponential in n.

Investigated problems. For each RML-fragment $\mathfrak{F}$, let $\mathsf{SAT}(\mathfrak{F})$ be the set of satisfiable $\mathfrak{F}$ formulas. In this paper, we investigate the complexity of $\mathsf{SAT}(\mathfrak{F})$ for any $\mathfrak{F} \in \{\mathsf{RML}, \mathsf{RML}^{\exists}, \mathsf{RML}^{\forall}, \mathsf{RML}^2, \ldots\}$. Figure 1 depicts our complexity results.

Assumption. Since RML is bisimulation invariant [20, 3], by Remark 1, w.l.o.g. we can assume that the semantics of RML is restricted to tree structures.

Since RML and ML have the same expressiveness [20, 3], we easily obtain the following (for details, see Appendix A).

Proposition 1 (Finite Model Property). *Let φ be a RML formula and $\langle T,V \rangle$ be a tree structure satisfying φ. Then, there is a* finite refinement *$\langle T_r,V_r \rangle$ of $\langle T,V \rangle$ satisfying φ.*

3 Upper bounds

In this section, we provide the upper bounds illustrated in Figure 1. Our approach consists of two steps. First, in Section 3.1, we show that RML enjoys a singly exponential size model property. Then, by using this result, we show in Section 3.2 that SAT(RML) can be decided by a singly exponential-time bounded ATM whose number of alternations on an input φ is at most $\Upsilon_w(\varphi)-1$ and whose initial state is existential. We fix a finite set P of atomic propositions and consider RML formulas and tree structures over P.

3.1 Exponential Size Model Property

In this section, we prove the following result.

Theorem 1 (Exponential Size Model Property). *For all satisfiable RML formulas φ and tree structures $\langle T,V\rangle$ such that $\langle T,V\rangle$ satisfies φ, the following holds: there exists a* finite refinement *$\langle T',V'\rangle$ of $\langle T,V\rangle$ such that $\langle T',V'\rangle$ satisfies φ and $|T'| \leq |\varphi|^{3|\varphi|^2}$.*

First, we summarize the main steps in the proof of Theorem 1. Given a RML formula φ, we associate with φ tableaux-based *finite* objects called *constraints systems for* φ (Definition 2). Essentially, a constraint system $\mathcal{S}$ for φ is a tuple of hierarchically ordered finite tree structures which intuitively represents an *extended model* of φ: (1) each node x in a tree structure of $\mathcal{S}$ is additionally labeled by a set of subformulas of φ which hold at the tree substructure rooted at node x, and, in particular, the first tree structure, called *main structure*, represents a model of φ, and (2) the other tree structures of $\mathcal{S}$ are used to manage the $\exists_r$-subformulas of φ. In fact, in order to be an *extended model* of φ, $\mathcal{S}$ has to satisfy additional structural requirements which capture the semantics of the boolean connectives and all the modalities except the universal refinement quantifier $\forall_r$, the latter being only semantically captured. Let $\mathcal{C}(\varphi)$ be the set of these constraints systems for φ, which are said to be *well-formed, saturated, and semantically $\forall_r$-consistent*. We individuate a subclass $\mathcal{C}_{min}(\varphi)$ of $\mathcal{C}(\varphi)$ consisting of '*minimal*' constraints systems for φ whose sizes are *singly exponential* in the size of φ, and which can be obtained from φ by applying structural *completion rules* (Definitions 3 and 4). Furthermore, we introduce a notion of 'refinement' between constraint systems for φ (Definition 5) which preserves the semantic $\forall_r$-consistency requirement. Then, given a *finite* tree structure $\langle T,V\rangle$ satisfying φ, we show that: (1) there is a constraint system $\mathcal{S} \in \mathcal{C}(\varphi)$ whose main structure is $\langle T,V\rangle$ (Lemma 1), and (2) starting from $\mathcal{S}$, it is possible to construct a minimal constraint system $\mathcal{S}_{min} \in \mathcal{C}_{min}(\varphi)$ which is a *refinement* of $\mathcal{S}$ (Lemma 3). This entails that the main structure of $\mathcal{S}_{min}$ is a refinement of $\langle T,V\rangle$ satisfying φ and having a single exponential size. Hence, by Proposition 1, Theorem 1 follows. Now, we proceed with the details of the proof of Theorem 1.

We denote by $\overline{P}$ the set of negations of propositions in P, i.e. $\overline{P} = \{\neg p \mid p \in P\}$. A set χ of RML formulas is *complete* if for each $p \in P$, either $p \in \chi$ or $\neg p \in \chi$. In the following, we fix a RML formula φ. The *closure* $\mathsf{cl}(\varphi)$ of φ is the set containing all the subformulas of φ and the formulas in $P \cup \overline{P}$. Moreover, $\mathsf{d}_{\Diamond,\Box}(\varphi)$ denotes the nesting depth of modalities $\Diamond$ and $\Box$ (in φ), and $\mathsf{d}_{\exists}(\varphi)$ denotes the nesting depth of modality $\exists_r$.

Definition 2. A *constraint system* for φ is a tuple $\mathcal{S} = \langle\langle T_1, L_1, \leftarrow_1\rangle, \ldots, \langle T_n, L_n, \leftarrow_n\rangle\rangle$, where for all $1 \leq i \leq n$, T_i is a *finite* tree and L_i and $\leftarrow_i$ are two T_i-labelings such that:

1. for each $x \in T_i$, $L_i(x)$ is a *complete* subset of $\mathsf{cl}(\varphi)$; moreover, $\varphi \in L_i(\varepsilon)$ if $i = 1$;
2. $\leftarrow_i : T_i \mapsto \{\bot\}$ if $i = 1$ ($\bot$ is for undefined), and $\leftarrow_i : T_i \mapsto \{j\} \times T_j$ for some $1 \leq j < i$ otherwise (note that $j < i$); moreover, for $i > 1$ and $x, x' \in T_i$ with $\leftarrow_i(x) = \langle j, y\rangle$ and $\leftarrow_i(x') = \langle j, y'\rangle$, if x' is a successor of x in T_i, then y' is a successor of y in T_j.

We denote by $\widetilde{L}_i$ the P-valuation over T_i defined as $\widetilde{L}_i(x) := L_i(x) \cap P$ for all $x \in T_i$, by $\mathcal{S}(i)$ the ith component of $\mathcal{S}$, i.e. $\langle T_i, L_i, \leftarrow_i\rangle$, and by $\mathsf{dim}(\mathcal{S})$ the number of $\mathcal{S}$ components, i.e., n. The (tree) structure $\langle T_1, \widetilde{L}_1\rangle$ is called the *main structure* of $\mathcal{S}$.
Let $1 \leq i, j \leq n$, $x \in T_i$, $y \in T_j$ and $\psi \in \mathsf{cl}(\varphi)$. We write $\langle j, y\rangle \leftarrow_{\mathcal{S}} \langle i, x\rangle$ to mean that $\leftarrow_i(x) = \langle j, y\rangle$, and $\mathcal{S} \vdash \langle i, x, \psi\rangle$ (resp., $\mathcal{S} \nvdash \langle i, x, \psi\rangle$) to mean that $\psi \in L_i(x)$ (resp., $\psi \notin L_i(x)$). If $\mathcal{S} \vdash \langle i, x, \psi\rangle$, we say that $\langle i, x, \psi\rangle$ is a $\mathcal{S}$-*constraint*. $\mathcal{S}$ contains a *clash* if $\mathcal{S} \vdash \langle i, x, p\rangle$ and $\mathcal{S} \vdash \langle i, x, \neg p\rangle$ for some $1 \leq i \leq n$, $x \in T_i$, and $p \in P$. Otherwise, $\mathcal{S}$ is called *clash-free*. Moreover, $\mathcal{S}$ is said to be *well-formed* if $\mathcal{S}$ is clash-free and whenever $\langle j, y\rangle \leftarrow_{\mathcal{S}} \langle i, x\rangle$, then $\widetilde{L}_j(y) = \widetilde{L}_i(x)$. Furthermore, $\mathcal{S}$ is said to be *semantically* $\forall_r$-*consistent* if whenever $\mathcal{S} \vdash (i, x, \forall_r \psi)$ then the tree structure $\langle T_i, \widetilde{L}_i\rangle_x$ satisfies $\forall_r \psi$.

If $\mathcal{S}$ is well-formed, then the labeling $\leftarrow_i$ induce a refinement hierarchy. More precisely:

Remark 2. Let $\mathcal{S} = \langle\langle T_1, L_1, \leftarrow_1\rangle, \ldots, \langle T_n, L_n, \leftarrow_n\rangle\rangle$ be a *well-formed* constraint system for φ. Then, $\langle j, y\rangle \leftarrow_{\mathcal{S}} \langle i, x\rangle$ implies that $\langle T_i, \widetilde{L}_i\rangle_x$ is a refinement of $\langle T_j, \widetilde{L}_j\rangle_y$.

Definition 3 (Saturated Constraint Systems). A constraint system $\mathcal{S}$ for φ is *saturated* if none of the following *completion rules* are applicable to $\mathcal{S}$.

$\wedge$-rule: `if` $\mathcal{S} \vdash \langle i, x, \psi_1 \wedge \psi_2\rangle$, $\mathcal{S}(i) = \langle T, L, \leftarrow\rangle$, and $\{\psi_1, \psi_2\} \not\subseteq L(x)$
`then` *update* $L(x) := L(x) \cup \{\psi_1, \psi_2\}$

$\vee$-rule: `if` $\mathcal{S} \vdash \langle i, x, \psi_1 \vee \psi_2\rangle$, $\mathcal{S}(i) = \langle T, L, \leftarrow\rangle$, and $\{\psi_1, \psi_2\} \cap L(x) = \emptyset$
`then` *update* $L(x) := L(x) \cup \{\psi_k\}$ for some $k \in \{1, 2\}$

$\exists_r$-rule: `if` $\mathcal{S} \vdash \langle i, x, \exists_r \psi\rangle$, $\mathcal{S} := \langle\langle T_1, L_1, \leftarrow_1\rangle, \ldots, \langle T_n, L_n, \leftarrow_n\rangle\rangle$, and
$\mathcal{S} \nvdash \langle h, \varepsilon, \psi\rangle$ for each $h \leq \mathsf{dim}(\mathcal{S})$ such that $\leftarrow_h(\varepsilon) = \langle i, x\rangle$
`then` *update* $\mathcal{S} := \langle\langle T_1, L_1, \leftarrow_1\rangle, \ldots, \langle T_{n+1}, L_{n+1}, \leftarrow_{n+1}\rangle\rangle$, where
$T_{n+1} := \{\varepsilon\}$, $L_{n+1}(\varepsilon) := \{\psi\} \cup (L_i(x) \cap (P \cup \overline{P}))$, and $\leftarrow_{n+1}(\varepsilon) := \langle i, x\rangle$

$\Box$-rule: `if` $\mathcal{S} \vdash \langle i, x, \Box\psi\rangle$ and $\mathcal{S} \nvdash \langle i, x', \psi\rangle$ for some successor x' of x in $\mathcal{S}(i)$
`then` *let* $\mathcal{S}(i) = \langle T, L, \leftarrow\rangle$
update $L(x') := L(x') \cup \{\psi\}$ for each successor x' of x in T

$\Diamond$-rule: `if` $\mathcal{S} \vdash \langle i, x, \Diamond\psi\rangle$ and $\mathcal{S} \nvdash \langle i, x', \psi\rangle$ for each successor x' of x in $\mathcal{S}(i)$
`then` let $\langle i_0, x_0\rangle \leftarrow_{\mathcal{S}} \ldots \leftarrow_{\mathcal{S}} \langle i_k, x_k\rangle$ with $i_0 = 1$ and $\langle i_k, x_k\rangle = \langle i, x\rangle$
guess some *complete* set $\chi \subseteq P \cup \overline{P}$,
`for each` $q = k, k-1, \ldots, 0$ with $\mathcal{S}(i_q) = \langle T_q, L_q, \leftarrow_q\rangle$ `do`
update $T_q := T_q \cup \{x_q \cdot h_q\}$ for some $h_q \in \mathbb{N}$ such that $x_q \cdot h_q \notin T_q$
`if` $q < k$ `then` $L_q(x_q \cdot h_q) := \chi$ and $\leftarrow_{i_{q+1}}(x_{q+1} \cdot h_{q+1}) := \langle i_q, x_q \cdot h_q\rangle$
`else` $L_q(x_q \cdot h_q) := \{\psi\} \cup \chi$

Remark 3. Let $\mathcal{S}$ be a constraint system for φ. Then, applying any rule of Definition 3 to $\mathcal{S}$ yields a constraint system for φ.

The $\vee$-rule, the $\wedge$-rule, and the $\Box$-rule of Definition 3 are standard. The $\exists_r$-rule and the $\Diamond$-rule are the unique rules which add new nodes to the given constraint system

$\mathcal{S}$ for φ. The $\exists_r$-rule is applicable to a $\mathcal{S}$-constraint $\xi = \langle i,x,\exists_r\psi\rangle$ if there are no $\mathcal{S}$-constraints (ξ-witnesses) of the form $\langle h,\varepsilon,\psi\rangle$ such that $\langle i,x\rangle \leftarrow_{\mathcal{S}} \langle h,\varepsilon\rangle$. The rule then adds a ξ-witness $\langle n+1,\varepsilon,\psi\rangle$ to $\mathcal{S}$ by extending $\mathcal{S}$ with a new component containing a single node (the root) whose label is propositionally consistent with the label of x. The $\Diamond$-rule is applicable to a $\mathcal{S}$-constraint $\xi = \langle i,x,\Diamond\psi\rangle$ if there are no $\mathcal{S}$-constraints (ξ-witnesses) of the form $\langle i,x',\psi\rangle$ where x' is a successor of x. Let $\langle i_0,x_0\rangle \leftarrow_{\mathcal{S}} \ldots \leftarrow_{\mathcal{S}} \langle i_k,x_k\rangle$ be the maximal chain of 'backward links' from $\langle i_k,x_k\rangle = \langle i,x\rangle$. The rule then adds a ξ-witness $\langle i_k,x'_k,\psi\rangle$ to $\mathcal{S}$ (x'_k being a new successor of $x_k = x$), a complete set $\chi \subseteq P \cup \overline{P}$ is guessed, and the hierarchical structure of $\mathcal{S}$ is restored as follows: the rule adds the new constraints $\langle i_0,x'_0,\chi\rangle,\ldots,\langle i_k,x'_k,\chi\rangle$, where $x'_0,\ldots,x'_{k-1}$ are new successors of $x_0,\ldots,x_{k-1}$ respectively, and the new chain of 'backward links' $\langle i_0,x'_0\rangle \leftarrow_{\mathcal{S}} \ldots \leftarrow_{\mathcal{S}} \langle i_k,x'_k\rangle$. The proof of the following lemma is given in Appendix B.1.

Lemma 1 (Soundness & Completeness). *Let $\langle T,V\rangle$ be a finite tree structure. Then, $\langle T,V\rangle$ satisfies φ* if and only if *there is a well-formed, saturated, and semantically $\forall_r$-consistent constraint system $\mathcal{S}$ for φ whose main structure is $\langle T,V\rangle$.*

Definition 4 (Minimal Constraint Systems). A constraint system $\mathcal{S}$ for φ is *initial* if $\mathcal{S} = \langle\langle\{\varepsilon\},L,\leftarrow\rangle\rangle$, and for all $\psi \in L(\varepsilon)$, either $\psi = \varphi$ or $\psi \in P \cup \overline{P}$. A *minimal* constraint system $\mathcal{S}$ for φ is a constraint system for φ which can be obtained from some *initial* constraint system for φ by a sequence of applications of the rules of Definition 3.

The following lemma (whose proof is in Appendix B.2) shows that every *minimal* constraint system for φ has a 'size' singly exponential in the size of φ.

Lemma 2. *Each* minimal *constraint system $\mathcal{S}$ for φ satisfies the following invariant: (i) each tree in $\mathcal{S}$ has height at most $\mathsf{d}_{\Diamond,\Box}(\varphi)$ and branching degree at most $|\varphi|^{(2\mathsf{d}_{\exists}(\varphi)+1)}$, and (ii) $\mathsf{dim}(\mathcal{S})$ is at most $|\varphi|^{4\cdot(\mathsf{d}_{\exists}(\varphi))^2\cdot(\mathsf{d}_{\Diamond,\Box}(\varphi)+1)}$. Moreover, any sequence of applications of the rules of Definition 3 starting from an* initial *constraint system for φ is finite.*

We introduce a notion of 'refinement' over constraint systems for φ, which generalizes the refinement preorder over finite structures. Moreover, this notion crucially preserves both well-formedness and the semantic $\forall_r$-consistency requirement (Lemma 3).

Definition 5 (Refinement for constraint systems). *Let $\mathcal{S} = \langle\langle T_1,L_1,\leftarrow_1\rangle,\ldots,\langle T_n,L_n,\leftarrow_n\rangle\rangle$ and $\mathcal{S}' = \langle\langle T'_1,L'_1,\leftarrow'_1\rangle,\ldots,\langle T'_m,L'_m,\leftarrow'_m\rangle\rangle$ be constraint systems for φ. $\mathcal{S}$ is a* refinement *of $\mathcal{S}'$ if there is a tuple $\mathcal{T} = \langle\uparrow_1,\ldots,\uparrow_n\rangle$ such that for all $1 \le i \le n$, there is $1 \le j \le m$ so that $\uparrow_i$: $T_i \mapsto \{j\} \times T'_j$ and for all $x \in T_i$ with $\uparrow_i(x) = \langle j,y\rangle$:*

1. *$L_i(x) \subseteq L'_j(y)$, $j = 1$ iff $i = 1$, and $y = \varepsilon$ iff $x = \varepsilon$;*
2. *for each successor x' of x in T_i, $\uparrow_i(x') = \langle j,y'\rangle$ where y' is a successor of y in T'_j.*
3. *if $\leftarrow_i(x) = \langle i',x'\rangle$, then $\leftarrow'_j(y) = \langle j',y'\rangle$ and $\uparrow_{i'}(x') = \langle j',y'\rangle$.*

Lemma 3 (Minimalization). *Let $\mathcal{S}'$ be a constraint system for φ which is well-formed and semantically $\forall_r$-consistent. Then, any constraint system $\mathcal{S}$ for φ which is a refinement of $\mathcal{S}'$ is well-formed and semantically $\forall_r$-consistent too, and the main structure of $\mathcal{S}$ is a refinement of the main structure of $\mathcal{S}'$. Moreover, if $\mathcal{S}'$ is additionally saturated, there is a* minimal *and saturated constraint system $\mathcal{S}$ for φ which is a refinement of $\mathcal{S}'$.*

Sketched proof. (Details are in Appendix B.3). The first part of Lemma 3 follows from Definition 5 and the following crucial observation: if $\langle T_r, V_r\rangle$ is a refinement of a tree structure $\langle T, V\rangle$, then for each $\forall_r$-formula $\forall_r\psi$, $\langle T, V\rangle \models \forall_r\psi$ implies $\langle T_r, V_r\rangle \models \forall_r\psi$. For the second part of Lemma 3, let $\mathcal{S}'$ be a well-formed, saturated, and semantically $\forall_r$-consistent constraint system for φ. Since $\mathcal{S}'$ is well-formed, there is a unique *initial* constraint system $\mathcal{S}_0$ for φ s.t. $\mathcal{S}_0$ is a refinement of $\mathcal{S}'$. For each rule of Definition 3, we define an extension of such a rule which has the same *precondition* and the same *effect* (w.r.t. a given constraint system $\mathcal{S}$) with the difference that the nondeterministic choices are guided by $\mathcal{S}'$. By Lemma 2, it follows that any sequence of applications of the *new* rules starting from $\mathcal{S}_0$ is *finite*. Moreover, the application of these new rules preserves the property of a constraint system to be a refinement of $\mathcal{S}'$. Hence, we deduce that there is a *minimal* and saturated constraint system $\mathcal{S}$ for φ which is a refinement of $\mathcal{S}'$. □

Proof of Theorem 1. Let $\langle T, V\rangle$ be a tree structure satisfying φ. By Proposition 1, there is a finite refinement $\langle T_r, V_r\rangle$ of $\langle T, V\rangle$ satisfying φ. By Lemma 1, there is a well-formed, saturated, and semantically $\forall_r$-consistent constraint system $\mathcal{S}$ for φ whose main structure is $\langle T_r, V_r\rangle$. Thus, by Lemma 3, there is a *minimal*, well-formed, saturated, and semantically $\forall_r$-consistent constraint system $\mathcal{S}_{min}$ for φ whose main structure $\langle T_{min}, V_{min}\rangle$ is a refinement of $\langle T_r, V_r\rangle$. Hence, $\langle T_{min}, V_{min}\rangle$ is a refinement of $\langle T, V\rangle$ as well, and by Lemmata 1 and 2, $\langle T_{min}, V_{min}\rangle$ satisfies φ and $|T_{min}| \leq |\varphi|^{3|\varphi|^2}$. Hence, the result follows.

3.2 Checking satisfiability

For a RML formula, the set $\mathsf{FL}(\varphi)$ of *first-level* subformulas of φ is defined as follows: if $\varphi = \varphi_1 \vee \varphi_2$ or $\varphi = \varphi_1 \wedge \varphi_2$, then $\mathsf{FL}(\varphi) = \mathsf{FL}(\varphi_1) \cup \mathsf{FL}(\varphi_2)$; otherwise $\mathsf{FL}(\varphi) = \{\varphi\}$.

Theorem 2. *$SAT(RML) \in AEXP_{pol}$ and $SAT(RML^k) \in \Sigma_k^{EXP}$ for each $k \geq 1$.*

Proof. For a RML formula φ, a *certificate* of φ is a finite tree structure $\langle T, V\rangle$ such that $|T| \leq |\varphi|^{3\cdot|\varphi|^2}$. Define:

$\widehat{\mathsf{SAT}}(\mathsf{RML}) := \{(\varphi, \langle T, V\rangle) \mid \varphi \in \mathsf{RML}$ *and* $\langle T, V\rangle$ is a certificate of φ satisfying $\varphi\}$

By Theorem 1, $\varphi \in \mathsf{SAT}(\mathsf{RML})$ iff $(\varphi, \langle T, V\rangle) \in \widehat{\mathsf{SAT}}(\mathsf{RML})$ for some certificate $\langle T, V\rangle$ of φ. Since $\langle T, V\rangle$ has size singly exponential in the size of φ, it suffices to show that $\widehat{\mathsf{SAT}}(\mathsf{RML})$ can be decided by a *polynomial-time* bounded ATM whose number of alternations on an input $(\varphi, \langle T, V\rangle)$ is at most $\Upsilon_w(\varphi) - 1$ and whose initial state is existential. For this, in turn, we show that $\widehat{\mathsf{SAT}}(\mathsf{RML})$ can be decided by a *nondeterministic polynomial-time bounded* procedure "*check*" that given an input $(\varphi, \langle T, V\rangle)$, uses in case $\Upsilon_w(\varphi) > 1$ as an *oracle* the same language $\widehat{\mathsf{SAT}}(\mathsf{RML})$ but with input queries of the form $(\psi, \langle T', V'\rangle)$, where $\Upsilon_w(\psi) < \Upsilon_w(\varphi)$ and $\psi \in \mathsf{cl}(\varphi)$. Hence, by standard arguments in complexity theory [12, 14], the result follows. Procedure *check* is defined as follows.

check$(\varphi, \langle T, V\rangle)$ /** $\varphi \in \mathsf{RML}$ and $\langle T, V\rangle$ is a certificate of φ **/

$\mathcal{K} \leftarrow \{(\varphi, \langle T, V\rangle)\}$;
`while` $\mathcal{K} \neq \emptyset$ `do`
  `select` $(\psi, \langle T', V'\rangle) \in \mathcal{K}$; `update` $\mathcal{K} \leftarrow \mathcal{K} \setminus \{(\psi, \langle T', V'\rangle)\}$;
  `guess` $\mathcal{F} \subseteq \mathsf{FL}(\psi)$ and let $\psi_{\mathcal{F}}$ be the *boolean* formula obtained from ψ by replac-

ing each first-level subformula θ of ψ with `true` if $\theta \in \mathcal{F}$, and `false` otherwise;
`if` $\psi_{\mathcal{F}}$ evaluates to `false` `then` *reject the input*;
`for each` $\theta \in \mathcal{F}$ `do`
`case` $\theta = p$ with $p \in P$: `if` $p \notin V'(\varepsilon)$ `then` *reject the input*;
`case` $\theta = \neg p$ with $p \in P$: `if` $p \in V'(\varepsilon)$ `then` *reject the input*;
`case` $\theta = \Diamond\theta'$: `guess` a child x of the T'-root, `update` $\mathcal{K} \leftarrow \mathcal{K} \cup \{(\theta', \langle T', V'\rangle_x)\}$;
`case` $\theta = \Box\theta'$: `update` $\mathcal{K} \leftarrow \mathcal{K} \cup \{(\theta', \langle T', V'\rangle_{x_1}), \ldots, (\theta', \langle T', V'\rangle_{x_k})\}$
where $x_1, \ldots, x_k$ are the children of the root of T';
`case` $\theta = \exists_r\theta'$: `guess` a *certificate* $\langle T'', V''\rangle$ of θ' which is a refinement of $\langle T', V'\rangle$ and `update` $\mathcal{K} \leftarrow \mathcal{K} \cup \{(\theta', \langle T'', V''\rangle)\}$;[1]
`case` $\theta = \forall_r\theta'$: `query` the oracle for $\widetilde{\mathsf{SAT}}(\mathsf{RML})$ with input $(\widetilde{\forall_r\theta'}, \langle T', V'\rangle)$;
/** note that $\Upsilon_w(\widetilde{\forall_r\theta'}) < \Upsilon_w(\forall_r\theta') \leq \Upsilon_w(\varphi)$ **/
`if` the oracle answers *YES* `then` *reject the input*;
`end for`
`end while`
accept the input.

Correctness of the procedure *check* easily follows from Theorem 1. □

4 Lower bounds

In this section, we provide the lower bounds illustrated in Figure 1. The main contribution is $\mathsf{AEXP}_{\mathsf{pol}}$-hardness of $\mathsf{SAT}(\mathsf{RML})$, which is proved by a polynomial-time reduction from a suitable $\mathsf{AEXP}_{\mathsf{pol}}$-complete problem. First, we define this problem.

Let $k \geq 1$. A *k-ary deterministic Turing Machine* is a deterministic Turing machine $\mathcal{M} = \langle k, I, A, Q, \{q_{acc}, q_{rej}\}, q_0, \delta\rangle$ operating on k ordered semi-infinite tapes and having just one read/write head, where: I (resp., $A \supset I$) is the input (resp., work) alphabet, A contains the blank symbol #, Q is the set of states, q_{acc} (resp., q_{rej}) is the terminal accepting (resp., rejecting) state, q_0 is the initial state, and $\delta : (Q \setminus \{q_{acc}, q_{rej}\}) \times A \to (Q \times A \times \{-1, +1\}) \cup \{1, \ldots, k\}$ is the transition function. In each non-terminal step, if the read/write head scans a cell of the ℓth tape ($1 \leq \ell \leq k$) and $(q, a) \in (Q \setminus \{q_{acc}, q_{rej}\}) \times A$ is the current pair state/ scanned cell content, the following occurs:

- $\delta(q, a) \in Q \times A \times \{-1, +1\}$ (*ordinary moves*): $\mathcal{M}$ overwrites the tape cell being scanned, there is a change of state, and the read/write head moves one position to the left (-1) or right (+1) in accordance with $\delta(q, a)$.
- $\delta(q, a) = h \in \{1, \ldots, k\}$ (*jump moves*): the read/write head jumps to the left-most cell of the hth tape and the state remains unchanged.

$\mathcal{M}$ *accepts* a k-ary input $(w_1, \ldots, w_k) \in (I^*)^k$, written $\mathcal{M}(w_1, \ldots, w_k)$, if the computation of $\mathcal{M}$ from $(w_1, \ldots, w_k)$ (initially, the ℓth tape contains the word w_ℓ, and the head points to the left-most cell of the first tape) is accepting. We consider the following problem.

[1] By Theorem 1, if there is a refinement of $\langle T', V'\rangle$ which satisfies θ', there is also a refinement of $\langle T', V'\rangle$ satisfying θ' which is a certificate of θ'. Moreover, checking for two given *finite* tree structures $\langle T, V\rangle$ and $\langle T', V'\rangle$, whether $\langle T, V\rangle$ is a refinement of $\langle T', V'\rangle$ (or, equivalently, $\langle T', V'\rangle$ is a simulation of $\langle T, V\rangle$) can be done in polynomial time (see, e.g., [13]).

Alternation Problem. An instance of the problem is a triple $(k,n,\mathcal{M})$, where $k \geq 1$, $n > 1$, and a $\mathcal{M}$ is a *polynomial-time bounded* k-ary deterministic Turing Machine with input alphabet I. The instance $(k,n,\mathcal{M})$ is *positive* iff the following holds, where $\mathsf{Q}_\ell = \exists$ if ℓ is odd, and $\mathsf{Q}_\ell = \forall$ otherwise (for all $1 \leq \ell \leq k$),

$$\mathsf{Q}_1 x_1 \in I^{2^n}.\, \mathsf{Q}_2 x_2 \in I^{2^n}. \ldots \mathsf{Q}_k x_k \in I^{2^n}.\, \mathcal{M}(x_1,\ldots,x_k)$$

Note that the quantifications Q_i are restricted to words over I of length 2^n.

For $k \geq 1$, the *k-Alternation Problem* is the Alternation Problem restricted to instances of the form $(k,n,\mathcal{M})$ (i.e., the first input parameter is fixed to k), and the *Linear Alternation Problem* is the Alternation Problem restricted to instances of the form $(n,n,\mathcal{M})$. The proof of the following result is standard (a proof is given in Appendix C).

Proposition 2. *The Linear Alternation Problem is AEXP_{pol}-complete and for all $k \geq 1$, the k-Alternation Problem is Σ_k^{EXP}-complete.*

Fix an instance $(k,n,\mathcal{M})$ of the Alternation Problem with $\mathcal{M} = \langle k, I, A, Q, \{q_{acc}, q_{rej}\}, q_0, \delta\rangle$. Since $\mathcal{M}$ is polynomial-time bounded, there is an integer constant $c \geq 1$ such that when started on a k-ary input $(w_1,\ldots,w_k)$, $\mathcal{M}$ reaches a terminal configuration in at most $(|w_1| + \ldots + |w_k|)^c$ steps. A (k,n)-input is a k-ary input $(w_1,\ldots,w_k)$ such that $w_i \in I^{2^n}$ for all $1 \leq i \leq k$. Let $\mathsf{c}(k,n) := c \cdot (n + \lceil \log k \rceil)$, where $\lceil \log k \rceil$ denotes the smallest $i \in \mathbb{N}$ such that $i \geq \log k$. Note that a configuration of $\mathcal{M}$ reachable from a (k,n)-input, called (k,n)-*configuration*, can be described as a tuple $\overrightarrow{C} = (C_1,\ldots,C_k)$ of k words $C_1,\ldots,C_k$ over $A \cup (Q \times A)$ of length exactly $2^{\mathsf{c}(k,n)}$ such that for some $1 \leq \ell \leq k$, C_ℓ is of the form $w \cdot (q,a) \cdot w' \in A^* \times (Q \times A) \times A^*$, and for $i \neq \ell$, $C_i \in A^{2^{\mathsf{c}(k,n)}}$. For a (k,n)-input $(a \cdot w_1,\ldots,w_k)$, the associated *initial* (k,n)-*configuration* is $((q_0,a) \cdot w_1 \cdot \#^{2^{\mathsf{c}(k,n)}-2^n},\ldots,w_k \cdot \#^{2^{\mathsf{c}(k,n)}-2^n})$. Thus, the computations of $\mathcal{M}$ from (k,n)-inputs, called (k,n)-*computations*, can be described by sequences π of at most $2^{\mathsf{c}(k,n)}$ (k,n)-configurations. In fact, w.l.o.g., we can assume that π has length *exactly* $2^{\mathsf{c}(k,n)}$. In the rest of this section, we prove the following result.

Theorem 3. *One can construct a RML^{k+1} formula φ in time polynomial in n, k, and the size of the TM $\mathcal{M}$ such that (i) φ is a $\mathsf{RML}^{\forall}$ formula if $k = 1$, and (ii) φ is satisfiable* if and only if *the instance $(k,n,\mathcal{M})$ of the Alternation Problem is positive.*

By Proposition 2 and Theorem 3, we obtain the following.

Corollary 1. *$\mathsf{SAT}(\mathsf{RML})$ is AEXP_{pol}-hard, $\mathsf{SAT}(\mathsf{RML}^{\forall})$ is NEXPTIME-hard, and for all $k \geq 1$, $\mathsf{SAT}(\mathsf{RML}^{k+1})$ is Σ_k^{EXP}-hard.*

Tree encoding of (k,n)-computations. In order to prove Theorem 3, first, we define an encoding of (k,n)-computations by suitable tree structures over P, where P is given by

$$P = \{0, 1, arg_1, \ldots, arg_k\} \cup \Lambda$$

and Λ is the set of triples (u_-, u, u_+) s.t. $u \in A \cup (Q \times A)$ and $u_-, u_+ \in A \cup (Q \times A) \cup \{\bot\}$ ($\bot$ is for undefined). An *extended TM block ext_bl* is a word over 2^P of length $2\mathsf{c}(k,n)+2$ of the form $ext\_bl = \{bit_1\} \cdot \ldots \cdot \{bit_{\mathsf{c}(k,n)}\} \cdot bl$, where bl, called *TM block*, is of the form $bl = \{bit'_1\} \cdot \ldots \cdot \{bit'_{\mathsf{c}(k,n)}\} \cdot \{arg_\ell\} \cdot \{t\}$ with $1 \leq \ell \leq k$ and $t \in \Lambda$. The *content* $\mathsf{CON}(ext\_bl)$ (resp., $\mathsf{CON}(bl)$) of ext_bl (resp., bl) is bl (resp., t), the *component number* of ext_bl and bl is ℓ, and the *position number* of ext_bl (resp., bl) is the integer in

$[0, 2^{\mathsf{c}(k,n)} - 1]$ whose binary code is $bit_1, \ldots, bit_{\mathsf{c}(k,n)}$ (resp., $bit'_1, \ldots, bit'_{\mathsf{c}(k,n)}$). Intuitively, *ext_bl* encodes the triple $t = (C_\ell(i-1), C_\ell(i), C_\ell(i))$ with $i = \mathsf{ID}(bl)$ (where $C_\ell(i-1) = \bot$ if $i = 0$, and $C_\ell(i+1) = \bot$ if $i = 2^{\mathsf{c}(k,n)} - 1$) of the ℓth component C_ℓ of some (k,n)-configuration, the latter being the $(\mathsf{ID}(ext\_bl))$-th (k,n)-configuration of some (k,n)-computation. For a sequence $\pi = \overrightarrow{C}_0, \ldots, \overrightarrow{C}_{2^{\mathsf{c}(k,n)}-1}$ of $2^{\mathsf{c}(k,n)}$ (k,n)-configurations, we can encode π by the set $S_{ext\_bl}(\pi)$ of extended blocks defined as: $ext\_bl \in S_{ext\_bl}(\pi)$ iff there are $0 \leq i, j \leq 2^{\mathsf{c}(k,n)} - 1$ and $0 \leq \ell \leq k$ such that $\mathsf{ID}(ext\_bl) = i$, $\mathsf{CON}(ext\_bl) = bl$, $\mathsf{ID}(bl) = j$ and bl is the TM block associated with the jth symbol of the ℓth component of $\overrightarrow{C}_i$. The tree representation of the set $S_{ext\_bl}(\pi)$ is defined as follows.

Definition 6. A (k,n)-*computation tree code is* a tree structure $\langle T, V\rangle$ over P such that:

1. Each path of $\langle T, V\rangle$ from the root has length $2\mathsf{c}(k,n)+2$, and each node is labeled exactly by a proposition in P. Moreover, $\langle T, V\rangle$ satisfies the ML-formula
$$\bigwedge_{i=1}^{2\mathsf{c}(k,n)} \Box^{i-1}((\Diamond 0 \wedge \Diamond 1) \wedge \Box(0 \vee 1)) \wedge \Box^{2\mathsf{c}(k,n)}(\bigwedge_{\ell=1}^{k} \Diamond arg_\ell \wedge \Box \bigvee_{\ell=1}^{k} arg_\ell) \wedge \Box^{2\mathsf{c}(k,n)+2} \bigvee_{t \in \Lambda} t$$
This requirement implies, in particular, that each path ν of $\langle T, V\rangle$ from the root is labeled by a word of the form $V(\varepsilon) \cdot ext\_bl$, where *ext_bl* is an extended TM block.
2. There is a sequence $\pi = \overrightarrow{C}_0, \ldots, \overrightarrow{C}_{2^{\mathsf{c}(k,n)}-1}$ of $2^{\mathsf{c}(k,n)}$ (k,n)-configurations such that the set of extended TM blocks of $\langle T, V\rangle$ corresponds to the set $S_{ext\_bl}(\pi)$.

We also need to encode existential and universal quantification on the different components of a (k,n)-input of the TM $\mathcal{M}$. This leads to the following definition.

Definition 7 (Initialized full (k,n)-computation tree codes). Let $1 \leq \ell \leq k$. A ℓ-*initialized full* (k,n)-*computation tree code* is a tree structure $\langle T, V\rangle$ over P such that:

1. *Fullness requirement*. $\langle T, V\rangle$ satisfies Property 1 of Definition 6. Moreover, let $\nu = z_0, \ldots, z_{2c(n)+1}, z_{2c(n)+2}$ be a path of $\langle T, V\rangle$ (from the root) encoding an extended TM block *ext_bl* with component number h such that either $h > \ell$ or $\mathsf{ID}(ext\_bl) > 0$. Then, for each $t \in \Lambda$, there is a child z of $z_{2c(n)+1}$ which is labeled by $\{t\}$.
2. ℓ-*initialization requirement*. There are $w_1, \ldots, w_\ell \in I^{2^n}$ s.t. for each extended block *ext_bl* of $\langle T, V\rangle$ with component number $1 \leq h \leq \ell$ and position number 0, $bl = \mathsf{CON}(ext\_bl)$ encodes the $\mathsf{ID}(bl)$th symbol of the hth component of any *initial* (k,n)-configuration associated with a (k,n)-input of the form $(w_1, \ldots, w_\ell, w'_{\ell+1}, \ldots, w'_k)$ for some $w'_{\ell+1}, \ldots, w'_k \in I^{2^n}$. We say that $w_1, \ldots, w_\ell \in I^{2^n}$ is the ℓ-ary input (which is uniquely determined) associated with $\langle T, V\rangle$ and we write $\langle T, V\rangle(w_1, \ldots, w_\ell)$.

Intuitively, a ℓ-initialized full (k,n)-computation tree code $\langle T, V\rangle$ associated with a ℓ-ary input $w_1, \ldots, w_\ell \in I^{2^n}$ encodes all the possible (k,n)-computations from (k,n)-inputs of the form $(w_1, \ldots, w_\ell, w'_{\ell+1}, \ldots, w'_k)$ for arbitrary words $w'_{\ell+1}, \ldots, w'_k \in I^{2^n}$. More precisely, by construction, the following holds.

Proposition 3. *Let $1 \leq \ell \leq k$, $w_1, \ldots, w_\ell \in I^{2^n}$, and $\langle T, V\rangle$ be a ℓ-initialized full (k,n)-computation tree code such that $\langle T, V\rangle(w_1, \ldots, w_\ell)$ holds. Then, the following holds:*

1. *case $\ell < k$: for each $w \in I^{2^n}$, there is a refinement $\langle T_r, V_r\rangle$ of $\langle T, V\rangle$ which is a $\ell+1$-initialized full (k,n)-computation tree code satisfying $\langle T_r, V_r\rangle(w_1, \ldots, w_\ell, w)$. Moreover, for each refinement $\langle T_r, V_r\rangle$ of $\langle T, V\rangle$ which is a $\ell+1$-initialized full (k,n)-computation tree code, there is $w \in I^{2^n}$ such that $\langle T_r, V_r\rangle(w_1, \ldots, w_\ell, w)$ holds.*

2. *case $\ell = k$: the set of refinements of $\langle T,V\rangle$ which are (k,n)-computation tree codes encoding (k,n)-computations is non-empty, and each of such refinements encodes the (k,n)-computation from the (k,n)-input $(w_1,\ldots,w_k)$.*

The proof of the following Lemma 4 is given in Appendix C.2.

Lemma 4. *One can construct in time polynomial in n, k, and the size of the TM $\mathcal{M}$,*

1. *a $\mathsf{RML}^{\forall}$ formula φ^1_{init} over P such that given a tree structure $\langle T,V\rangle$, $\langle T,V\rangle$ satisfies φ^1_{init} if and only if $\langle T,V\rangle$ is a 1-initialized full (k,n)-computation tree code;*
2. *a $\mathsf{RML}^{\forall}$ formula φ^{ℓ}_{init} over P (for each $2 \le \ell \le k$) such that given a* refinement *$\langle T_r,V_r\rangle$ of a $\ell-1$-initialized full (k,n)-computation tree code, $\langle T_r,V_r\rangle$ satisfies φ^{ℓ}_{init}* if and only if *$\langle T_r,V_r\rangle$ is a ℓ-initialized full (k,n)-computation tree code;*
3. *a $\mathsf{RML}^{\forall}$ formula φ_{comp} over P such that given a* refinement *$\langle T_r,V_r\rangle$ of a k-initialized full (k,n)-computation tree code, $\langle T_r,V_r\rangle$ satisfies φ_{comp}* iff *$\langle T_r,V_r\rangle$ is a (k,n)-computation tree code encoding a (k,n)-computation;*
4. *a ML formula φ_{acc} over P such that given a (k,n)-computation tree code $\langle T,V\rangle$, $\langle T,V\rangle$ satisfies φ_{acc}* iff *the (k,n)-configuration with position number $2^{c(k,n)-1}$ encoded by $\langle T,V\rangle$ is accepting.*

Theorem 3 directly follows from the following two results (Theorems 4 and 5).

Theorem 4. *One can construct a RML^{k+1} formula φ in time polynomial in n, k, and the size of the TM $\mathcal{M}$ such that φ is satisfiable* if and only if

$$\mathsf{Q}_1 x_1 \in I^{2^n}.\, \mathsf{Q}_2 x_2 \in I^{2^n} \ldots \mathsf{Q}_k x_k \in I^{2^n}.\, \mathcal{M}(x_1,\ldots,x_k)$$

where $\mathsf{Q}_\ell = \exists$ if ℓ is odd, and $\mathsf{Q}_\ell = \forall$ otherwise (for all $1 \le \ell \le k$).

Proof. Let $\varphi^1_{init},\ldots,\varphi^k_{init}$, and φ_{comp} be the $\mathsf{RML}^{\forall}$ formulas satisfying Properties 1–3 of Lemma 4, and φ_{acc} be the ML formula satisfying Property 4 of Lemma 4. Then, the RML^{k+1} formula φ is defined as follows, where $\widetilde{\mathsf{Q}}_\ell = \exists_r$ and $\mathrm{op}_\ell = \wedge$ if ℓ is odd, and $\widetilde{\mathsf{Q}}_\ell = \forall_r$ and $\mathrm{op}_\ell = \rightarrow$ otherwise (for all $2 \le \ell \le k$):[2]

$$\varphi := \varphi^1_{init} \wedge \widetilde{\mathsf{Q}}_2(\varphi^2_{init}\ \mathrm{op}_2\ \widetilde{\mathsf{Q}}_3(\varphi^3_{init}\ \mathrm{op}_3\ \ldots\ \mathrm{op}_{k-1}\ \widetilde{\mathsf{Q}}_k(\varphi^k_{init}\ \mathrm{op}_k\ \widetilde{\mathsf{Q}}_k(\varphi_{comp}\ \mathrm{op}_k\ \varphi_{acc}))\ldots))$$

By construction and Lemma 4, it easily follows that φ is RML^{k+1} formula which can be constructed in time polynomial in n, k, and the size of the TM $\mathcal{M}$. Let $\varphi_1 := \varphi$, $\varphi_{k+1} := \widetilde{\mathsf{Q}}_k(\varphi_{comp}\ \mathrm{op}_k\ \varphi_{acc})$, and for each $2 \le \ell \le k$,

$$\varphi_\ell := \widetilde{\mathsf{Q}}_\ell(\varphi^\ell_{init}\ \mathrm{op}_\ell\ \widetilde{\mathsf{Q}}_{\ell+1}(\varphi^{\ell+1}_{init}\ \mathrm{op}_{\ell+1}\ \ldots\ \mathrm{op}_{k-1}\ \widetilde{\mathsf{Q}}_k(\varphi^k_{init}\ \mathrm{op}_k\ \widetilde{\mathsf{Q}}_k(\varphi_{comp}\ \mathrm{op}_k\ \varphi_{acc}))\ldots))$$

Correctness of the construction directly follows from the following claim, where a 0-initialized full (k,n)-computation tree code is an arbitrary tree structure.

Claim: let $0 \le \ell \le k$, $w_1,\ldots,w_\ell \in I^{2^n}$, and $\langle T,V\rangle$ be a ℓ-initialized full (k,n)-computation tree code such that $\langle T,V\rangle(w_1,\ldots,w_\ell)$ holds. Then, $\langle T,V\rangle$ satisfies $\varphi_{\ell+1}$ if and only if

$$\mathsf{Q}_{\ell+1} x_{\ell+1} \in I^{2^n} \ldots \mathsf{Q}_k x_k \in I^{2^n}.\, \mathcal{M}(w_1,\ldots,w_\ell,x_{\ell+1},\ldots,x_k)$$

The claim follows from Proposition 3 and Lemma 4 (details are in Appendix C.3). □

The proof of the following theorem is given in Appendix C.4.

Theorem 5. *Let $k = 1$. Then, one can construct a $\mathsf{RML}^{\forall}$ formula $\varphi^{\forall}$ in time polynomial in n and the size of the TM $\mathcal{M}$ such that $\varphi^{\forall}$ is satisfiable* if and only if $\exists x \in I^{2^n}.\, \mathcal{M}(x)$.

[2] For RML formulas φ and ψ, $\varphi \rightarrow \psi$ is an abbreviation for $\widetilde{\varphi} \vee \psi$.

5 Concluding remarks

An intriguing question left open is the complexity of satisfiability for *multi-agent* refinement modal logic [20, 3]: we conjecture that this problem remains in $\mathsf{AEXP}_{\mathsf{pol}}$. Moreover, it would be interesting to investigate the exact complexity of the fragments $\mathsf{RML}^{\exists}$, $\mathsf{RML}^{\forall}$, and RML^{k}, and the succinctness gap between RML^{k} and RML^{k+1} for each $k \geq 1$. Furthermore, since the modal μ-calculus extended with refinement quantifiers (RML^{μ}, for short) is non-elementarily decidable [3], an other interesting direction is to individuate weak forms of interactions between fixed-points and refinement quantifiers, which may lead to elementarily decidable and interesting RML^{μ}-fragments.

References

1. P. Balbiani, A. Baltag, H. van Ditmarsch, A. Herzig, T. Hoshi, and T. De Lima. 'Knowable' as 'known after an announcement'. *Review of Symbolic Logic*, 1(3):305–334, 2008.
2. P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, Cambridge, 2001. Cambridge Tracts in Theoretical Computer Science 53.
3. L. Bozzelli, H. van Ditmarsch, T. French, J. Hales, and S. Pinchinat. Refinement modal logic. Available at http://arxiv.org/abs/1202.3538, 2012.
4. A.K. Chandra, D.C. Kozen, and L.J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981.
5. E.M. Clarke and E.A. Emerson. Design and verification of synchronization skeletons using branching time temporal logic. In *Proc. of Workshop on Logic of Programs*, LNCS 131, pages 52–71. Springer-Verlag, 1981.
6. E.A. Emerson and C.S. Jutla. The complexity of tree automata and logics of programs. In *Proc. 29th FOCS*, pages 328–337, 1988.
7. J. Ferrante and C. Rackoff. A decision procedure for the first order theory of real addition with order. *SIAM Journal on Computing*, 4(1):69–76, 1975.
8. K. Fine. Propositional quantifiers in modal logic. *Theoria*, 36(3):336–346, 1970.
9. T. French. *Bisimulation quantifiers for modal logic*. PhD thesis, University of Western Australia, 2006.
10. D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. MIT Press, 2000.
11. M. Hollenberg. *Logic and bisimulation*. PhD thesis, University of Utrecht, 1998.
12. D.S. Johnson. A catalog of complexity classes. In *Handbook of Theoretical Computer Science*, pages A:67–161. MIT Press, 1990.
13. O. Kupferman and M.Y. Vardi. Verification of Fair Transisiton Systems. In *Proc. 8th CAV*, LNCS 1102, pages 372–382. Springer, 1996.
14. C.H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.
15. R. Parikh, L. Moss, and C. Steinsvold. Topology and epistemic logic. In *Handbook of Spatial Logics*, pages 299–341. Springer Verlag, 2007.
16. Sophie Pinchinat. A generic constructive solution for concurrent games with expressive constraints on strategies. In *Proc. 5th ATVA*, LNCS 4762, pages 253–267. Springer, 2007.
17. T.A. Henzinger R. Alur and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.
18. T. Rybina and A. Voronkov. Upper bounds for a theory of queues. In *Proc. 30th ICALP*, LNCS 2719, pages 714–724. Springer, 2003.
19. H. van Ditmarsch and T. French. Simulation and information. In *Proc. Knowledge Representation for Agents and Multi-Agent Systems*, LNAI 5605, pages 51–65. Springer, 2009.
20. H. van Ditmarsch, T. French, and S. Pinchinat. Future event logic - axioms and complexity. In *Proc. 7th Advances in Modal Logic*, volume 8, pages 77–99. College Publications, 2010.
21. A. Visser. Bisimulations, model descriptions and propositional quantifiers, 1996. Logic Group Preprint Series 161, Department of Philosophy, Utrecht University.

Appendix

A Proof of Proposition 1

Proposition 1 (Finite Model Property). *Let* φ *be a* RML *formula and* $\langle T,V\rangle$ *be a tree structure satisfying* φ*. Then, there is a* finite refinement $\langle T_r,V_r\rangle$ *of* $\langle T,V\rangle$ *satisfying* φ*.*

Proof. Since for each RML formula, there is an equivalent ML formula [20, 3], it suffices to show that the result holds for ML. Let φ be a ML formula and $\langle T,V\rangle$ be a tree structure satisfying φ. We denote by $\mathsf{d}_{\Diamond,\Box}(\varphi)$ the nesting depth of modalities $\Diamond$ and $\Box$ in φ. Let $\langle T_{pr},V_{pr}\rangle$ be the tree structure obtained from $\langle T,V\rangle$ by pruning all the subtrees rooted at nodes $x\in T\subseteq\mathbb{N}^*$ such that $|x|>\mathsf{d}_{\Diamond,\Box}(\varphi)$. By a straightforward induction on $\mathsf{d}_{\Diamond,\Box}(\varphi)$, we deduce that $\langle T_{pr},V_{pr}\rangle$ satisfies φ as well. Now, we observe that $\langle T_{pr},V_{pr}\rangle$ is a *refinement* of $\langle T,V\rangle$ having *finite* height (bounded by $\mathsf{d}_{\Diamond,\Box}(\varphi)$). Since $\langle T_{pr},V_{pr}\rangle$ has finite height, we easily deduce that there is a *finite* tree structure $\langle T_r,V_r\rangle$ such that $\langle T_r,V_r\rangle$ and $\langle T_{pr},V_{pr}\rangle$ are bisimilar. Hence, $\langle T_r,V_r\rangle$ is a refinement of $\langle T,V\rangle$. Moreover, since ML is bisimulation invariant, $\langle T_r,V_r\rangle$ satisfies φ as well, and we are done. □

B Proofs from Section 3

B.1 Proof of Lemma 1

In order to prove Lemma 1, we need additional definitions. For a RML formula φ and a tree structure $\langle T,V\rangle$, the φ-*completion* of $\langle T,V\rangle$ is the labeled tree $\langle T,L\rangle$, where for each $x\in T$, $L(x)$ is the set of formulas $\psi\in\mathsf{cl}(\varphi)$ such that $\langle T,V\rangle_x$ (the tree substructure of $\langle T,V\rangle$ rooted at x) satisfies ψ. Note that $L(x)$ is a complete subset of $\mathsf{cl}(\varphi)$ and $L(x)\cap P=V(x)$. A constraint system $\mathcal{S}=\langle\langle T_1,L_1,\leftarrow_1\rangle,\ldots,\langle T_n,L_n,\leftarrow_n\rangle\rangle$ for φ is said to be a *model* for φ if for each $1\leq i\leq n$, there is a formula $\psi_i\in\mathsf{cl}(\varphi)$ such that $\langle T_i,L_i\rangle$ is the ψ_i-completion of $\langle T_i,\widetilde{L}_i\rangle$ (note that $\psi_1=\varphi$).

Lemma 1 (Soundness & Completeness). *Let* $\langle T,V\rangle$ *be a finite tree structure. Then,* $\langle T,V\rangle$ *satisfies* φ if and only if *there is a well-formed, saturated, and semantically* $\forall_r$*-consistent constraint system* $\mathcal{S}$ *for* φ *whose main structure is* $\langle T,V\rangle$*.*

Proof. **Soundness:** Let $\mathcal{S}=\langle\langle T_1,L_1,\leftarrow_1\rangle,\ldots,\langle T_n,L_n,\leftarrow_n\rangle\rangle$ be a constraint system for φ such that $\mathcal{S}$ is well-formed, saturated, and semantically $\forall_r$-consistent. We need to show that the main structure of $\mathcal{S}$ satisfies φ. For this, it suffices to show that for all $1\leq i\leq n$, $x\in T_i$, and $\psi\in L_i(x)$, the tree structure $\langle T_i,\widetilde{L}_i\rangle_x$ satisfies ψ (recall that $\varphi\in L_1(\varepsilon)$). The proof is by structural induction on ψ:

- $\psi=p\in P$. Since $p\in L_i(x)$, $p\in\widetilde{L}_i(x)$ as well. Hence, the result follows.
- $\psi=\neg p$ and $p\in P$. Since $\neg p\in L_i(x)$ and $\mathcal{S}$ is well-formed (hence, clash-free), it holds that $p\notin L_i(x)$. Hence, $p\notin\widetilde{L}_i(x)$ and the result follows.
- $\psi=\psi_1\wedge\psi_2$ (resp., $\psi=\psi_1\vee\psi_2$). Since $\psi\in L_i(x)$ and $\mathcal{S}$ is saturated, by the precondition of the $\wedge$-rule (resp., $\vee$-rule), we obtain that $\psi_1,\psi_2\in L_i(x)$ (resp., $\psi_k\in L_i(x)$ for some $k\in\{1,2\}$). Hence, by the induction hypothesis, the result follows.

– $\psi = \Box\psi'$ (resp., $\psi = \Diamond\psi'$). Since $\psi \in L_i(x)$ and $\mathcal{S}$ is saturated, by the precondition of the $\Box$-rule (resp., $\Diamond$-rule), we obtain that $\psi' \in L_i(x')$ for each (resp., for some) successor x' of x in T_i. Hence, by the induction hypothesis, the result follows.
– $\psi = \exists_r\psi'$. Since $\psi \in L_i(x)$ and $\mathcal{S}$ is saturated, by the precondition of the $\exists_r$-rule, there is some $h \leq \mathsf{dim}(\mathcal{S})$ such that $\leftarrow_h(\varepsilon) = \langle i,x\rangle$ and $\psi' \in L_h(\varepsilon)$. By the induction hypothesis, $\langle T_h, \widetilde{L}_h\rangle$ satisfies ψ'. Moreover, since $\leftarrow_h(\varepsilon) = \langle i,x\rangle$ and $\mathcal{S}$ is well-formed, by Remark 2, it holds that $\langle T_h, \widetilde{L}_h\rangle$ is a refinement of $\langle T_i, \widetilde{L}_i\rangle_x$. Hence, $\langle T_i, \widetilde{L}_i\rangle_x$ satisfies $\exists_r\psi'$ and the result follows.
– $\psi = \forall_r\psi'$. Since $\psi \in L_i(x)$ and $\mathcal{S}$ is semantically $\forall_r$-consistent, the result follows.

Completeness: let $\langle T,V\rangle$ be a finite tree structure satisfying φ. We need to show that there is a well-formed, saturated, and semantically $\forall_r$-consistent constraint system $\mathcal{S}$ for φ whose main structure is $\langle T,V\rangle$. Let $\mathcal{S}_0 = \langle T, L, \leftarrow\rangle$, where $\langle T,L\rangle$ is the φ-completion of $\langle T,V\rangle$ and $\leftarrow(x) = \bot$ for each $x \in T$. Evidently, $\mathcal{S}_0$ is a model for φ which is well-formed and whose main structure is $\langle T,V\rangle$. We extend $\mathcal{S}_0$ by repeated applications of the following rule:

Semantic $\exists_r$-rule:
`if` $\mathcal{S} \vdash (i,x,\exists_r\psi)$, $\mathcal{S} = \langle\langle T_1, L_1, \leftarrow_1\rangle, \ldots, \langle T_n, L_n, \leftarrow_n\rangle\rangle$ is *a model* for φ, and
$\mathcal{S} \nvdash (h, \varepsilon, \psi)$ for each $h \leq \mathsf{dim}(\mathcal{S})$ such that $\leftarrow_h(\varepsilon) = \langle i,x\rangle$
`then` *update* $\mathcal{S} := \langle\langle T_1, L_1, \leftarrow_1\rangle, \ldots, \langle T_{n+1}, L_{n+1}, \leftarrow_{n+1}\rangle\rangle$ where
$\langle T_{n+1}, L_{n+1}\rangle$ is the ψ-completion of some *finite* refinement $\langle T_{n+1}, \widetilde{L}_{n+1}\rangle$ of $\langle T_i, \widetilde{L}_i\rangle_x$ satisfying ψ, and $\leftarrow_{n+1}$ is some labeling ensuring that $\leftarrow_{n+1}(\varepsilon) = \langle i,x\rangle$, $\mathcal{S}$ is still a constraint system for φ, and for each $y \in T_{n+1}$ such that $\leftarrow_{n+1}(y) = \langle i,z\rangle$, $\widetilde{L}_{n+1}(y) = \widetilde{L}_i(z)$.
/** Since $\exists_r\psi \in L_i(x)$ and $\mathcal{S}$ is a model of φ, there is a refinement $\langle T_{n+1}, V_{n+1}\rangle$ of $\langle T_i, \widetilde{L}_i\rangle_x$ satisfying ψ. Thus, since refinement is a preorder, by Proposition 1, there is a *finite* refinement $\langle T_{n+1}, \widetilde{L}_{n+1}\rangle$ of $\langle T_i, \widetilde{L}_i\rangle_x$ satisfying ψ. **/

We show that by a finite number of applications of the semantic $\exists_r$-rule starting from the initial model $\mathcal{S}_0$ for φ (which is also well-formed), we obtain a well-formed, saturated, and semantically $\forall_r$-consistent constraint system for φ. Hence, since the main structure of $\mathcal{S}_0$ is $\langle T,V\rangle$ and applications of the semantic $\exists_r$-rule do not modify the main structure, the result follows. Note that if $\mathcal{S}$ is a model for φ, then $\mathcal{S}$ is semantically $\forall_r$-consistent, and at most the $\exists_r$-rule of Definition 3 is applicable to $\mathcal{S}$. Moreover, the application of the semantic $\exists_r$-rule preserves the property of a constraint system to be well-formed and a model for φ. Since the precondition of the semantic $\exists_r$-rule corresponds to the precondition of the $\exists_r$-rule in Definition 3, it suffices to show the following:

Claim: any sequence of applications of the semantic $\exists_r$-rule starting from $\mathcal{S}_0$ is *finite*.

Proof of the claim: fix an ordering $\varphi_1, \ldots, \varphi_k$ of the formulas in $\mathsf{cl}(\varphi)$ such that for all $i \neq j$, $\mathsf{d}_\exists(\varphi_i) > \mathsf{d}_\exists(\varphi_j)$ implies $i < j$. For each model $\mathcal{S}$ for φ which can be obtained from $\mathcal{S}_0$ by a sequence of applications of the semantic $\exists_r$-rule, we associate to $\mathcal{S}$ a k-tuple of natural numbers $< \mathcal{S} >$ as follows. Let $\mathcal{S} = \langle\langle T_1, L_1, \leftarrow_1\rangle, \ldots, \langle T_n, L_n, \leftarrow_n\rangle\rangle$. For each $1 < i \leq \mathsf{dim}(\mathcal{S})$, let $\psi_i \in \mathsf{cl}(\varphi)$ be the formula such that $\exists_r\psi_i$ is in the precondition of the semantic $\exists_r$-rule instance whose application has generated the ith component $\mathcal{S}(i)$ of $\mathcal{S}$. Moreover, let $\psi_1 = \varphi$. Note that for all $1 \leq i \leq \mathsf{dim}(\mathcal{S})$, $\langle T_i, L_i\rangle$ is the ψ_i-completion of $\langle T_i, \widetilde{L}_i\rangle$. Then, for each $\psi \in \mathsf{cl}(\varphi)$, define

$$I(\mathcal{S}, \psi) = \{1 \leq i \leq dim(\mathcal{S}) \mid \psi_i = \psi\}$$

Moreover, for all $1 \leq i \leq \mathsf{dim}(\mathcal{S})$ and $\exists_r\psi \in \mathsf{cl}(\varphi)$, let $T(\mathcal{S},i,\exists_r\psi)$ be the set of nodes $x \in T_i$ such that the semantic $\exists_r$-rule is applicable to node x with respect to formula $\exists_r\psi$, i.e.

$T(\mathcal{S},i,\exists_r\psi) = \{x \in T_i \mid \exists_r\psi \in L_i(x)$ and there is no $j \leq \mathsf{dim}(\mathcal{S})$ s.t. $\leftarrow_j(\varepsilon) = \langle i,x\rangle$ and $\psi \in L_j(\varepsilon)\}$

Then, $< S >= \langle n_1,\ldots,n_k\rangle$ where for all $1 \leq j \leq k$

$$n_j = \sum_{i \in I(\mathcal{S},\varphi_j)} \sum_{\exists_r\psi \in cl(\varphi_j)} |T(\mathcal{S},i,\exists_r\psi)|$$

Let $<_k$ be the lexicographic ordering over $\mathbb{N}^k$. Then, by construction, for each model $\mathcal{S}'$ for φ obtained form $\mathcal{S}$ by an application of the semantic $\exists_r$-rule, it easily follows that $< \mathcal{S}' ><_k < \mathcal{S} >$. Thus, since $<_k$ is well-founded, the claim follows. □

B.2 Proof of Lemma 2

Lemma 2. *Each* minimal *constraint system $\mathcal{S}$ for φ satisfies the following invariant: (i) each tree in $\mathcal{S}$ has height at most $\mathsf{d}_{\diamond,\Box}(\varphi)$ and branching degree at most $|\varphi|^{(2\mathsf{d}_\exists(\varphi)+1)}$, and (ii) $\mathsf{dim}(\mathcal{S})$ is at most $|\varphi|^{4\cdot(\mathsf{d}_\exists(\varphi))^2\cdot(\mathsf{d}_{\diamond,\Box}(\varphi)+1)}$. Moreover, any sequence of applications of the rules of Definition 3 starting from an* initial *constraint system for φ is finite.*

Proof. Recall that the labeling of a node in a constraint system $\mathcal{S}$ for φ contains only formulas in $\mathsf{cl}(\varphi)$, and every application to $\mathcal{S}$ of a rule in Definition 3 either adds a new node or a new component with a single node, or add a new formula in $\mathsf{cl}(\varphi)$ to a label of a node. Thus, since $\mathsf{cl}(\varphi)$ is finite, Properties (i) and (ii) in the statement of the lemma entail that any sequence of applications of the rules of Definition 3 starting from an *initial* constraint system for φ is finite. Thus, we need to prove Properties (i) and (ii). Let $\mathcal{S} = \langle\langle T_1,L_1,\leftarrow_1\rangle,\ldots,\langle T_n,L_n,\leftarrow_n\rangle\rangle$ be a *minimal* constraint system for φ. For all $1 \leq i \leq \mathsf{dim}(\mathcal{S})$ and $x \in T_i$, x satisfies one of the following:

- x is the root of T_i. Moreover, if $i > 1$, then the root of T_i is generated by an application of the $\exists_r$-rule. The formula ψ in the body of the $\exists_r$-rule which is added to the label of the root of T_i (when such a root is generated) is called *main formula* of $\mathcal{S}(i)$. The *main formula* of $\mathcal{S}(1)$ is φ.
- x is not the root of T_i, x is generated be executing the body of the $\diamond$-rule, and in particular, x corresponds to node $x_k \cdot h_k$ in the body of the $\diamond$-rule (see Definition 3).
- x is not the root of T_i, x is generated be executing the body of the $\diamond$-rule, and in particular, x corresponds to some node $x_q \cdot h_q$ with $q < k$ in the body of the $\diamond$-rule (see Definition 3).

In the first two cases, we say that x is a main node. Otherwise, x is said to be a secondary node. For each $1 \leq i \leq \mathsf{dim}(\mathcal{S})$, the main formula of $\mathcal{S}(i)$ is denoted by φ_i (note that $\varphi_1 = \varphi$). For all $1 \leq i \leq \mathsf{dim}(\mathcal{S})$ and $x \in T_i$, define $\rightarrow_i(x) = \{1 \leq h \leq \mathsf{dim}(\mathcal{S}) \mid \leftarrow_h(\varepsilon) = \langle i,x\rangle\}$. Note that for each $h \in \rightarrow_i(x)$, $h > i$ and the root of T_h is generated by executing the $\exists_r$-rule applied to the constraint $\langle i,x,\exists_r\varphi_h\rangle$. Hence, in particular, $\exists_r\varphi_h \in L_i(x)$. Since $\varphi = \varphi_1$ and $\varphi_i \in \mathsf{cl}(\varphi)$ for each $1 \leq i \leq \mathsf{dim}(\mathcal{S})$, Properties (i) and (ii) in the lemma directly follow from the following claims.

Claim 1: For all $1 \leq i \leq \mathsf{dim}(\mathcal{S})$, $x \in T_i$, and $h \in \rightarrow_i(x)$, it holds that $L_i(x) \subseteq \mathsf{cl}(\varphi_i)$, $|\rightarrow_i(x)| \leq |\varphi_i|$, $\mathsf{d}_{\diamond,\Box}(\varphi_h) \leq \mathsf{d}_{\diamond,\Box}(\varphi_i)$, $\mathsf{d}_\exists(\varphi_h) < \mathsf{d}_\exists(\varphi_i)$, and φ_h is a subformula of φ_i.

Proof of Claim 1: let $1 \leq i \leq \mathsf{dim}(\mathcal{S})$ and $x \in T_i$. First, we show that $L_i(x) \subseteq \mathsf{cl}(\varphi_i)$. Evidently, $L_i(\varepsilon) \subseteq \mathsf{cl}(\varphi_i)$. Thus, it suffices to show that for all nodes y and z of T_i such that z is a successor of y and $L_i(y) \subseteq \mathsf{cl}(\varphi_i)$, $L_i(z) \subseteq \mathsf{cl}(\varphi_i)$ holds as well. This follows from the fact that for each $\psi \in L_i(z)$

such that $\psi \notin P \cup \overline{P}$, either $\Diamond\psi \in L_i(y)$ or $\Box\psi \in L_i(y)$ (i.e., ψ has been added to the label of z by a $\Diamond$-rule or a $\Box$-rule), or ψ has been added to $L_i(z)$ by an application of the $\vee$-rule or $\wedge$-rule starting from a formula $\psi' \in L_i(z)$ such that ψ is strict subformula of ψ'. Now, let us show that $|{\rightarrow_i}(x)| \leq |\varphi_i|$. Since ${\rightarrow_i}(x)$ can be implicitly updated only by a $\exists_r$-rule application, it easily follows that ${\rightarrow_i}(x) = \{i_1, \ldots, i_k\}$, $\exists_r\varphi_{i_1}, \ldots, \exists_r\varphi_{i_k} \in L_i(x)$, and $\exists_r\varphi_{i_1}, \ldots, \exists_r\varphi_{i_k}$ are pairwise distinct. Since $L_i(x) \subseteq \mathsf{cl}(\varphi_i)$, the result follows. Moreover, we obtain that φ_h is a subformula of φ_i, $\mathsf{d}_{\Diamond,\Box}(\varphi_h) \leq \mathsf{d}_{\Diamond,\Box}(\varphi_i)$, and $\mathsf{d}_{\exists}(\varphi_h) < \mathsf{d}_{\exists}(\varphi_i)$ for all $h \in {\rightarrow_i}(x)$. □

Claim 2: For all $1 \leq i \leq \mathsf{dim}(\mathcal{S})$, the height of T_i is bounded by $\mathsf{d}_{\Diamond,\Box}(\varphi_i)$.

Proof of Claim 2: For a finite set χ of RML formulas, define $\mathsf{d}_{\Diamond,\Box}(\chi) := max(\{\mathsf{d}_{\Diamond,\Box}(\psi) \mid \psi \in \chi\})$. The proof is by induction on $\mathsf{dim}(\mathcal{S}) - i$. We consider the induction step (the base case $i = \mathsf{dim}(\mathcal{S})$ being simpler). Thus, let $i < \mathsf{dim}(\mathcal{S})$. A node x of T_i is said to be *propositional* if $L_i(x) \subseteq P \cup \overline{P}$. Let $\pi = x_0, \ldots, x_k$ be a path of T_i from the root. We need to show that the length of π is bounded by $\mathsf{d}_{\Diamond,\Box}(\varphi_i)$. Observe that since $x_0 = \varepsilon$, $\varphi_i \in L_i(\varepsilon)$, and $L_i(\varepsilon) \subseteq \mathsf{cl}(\varphi_i)$ (Claim 1), it holds that $\mathsf{d}_{\Diamond,\Box}(L_i(x_0)) = \mathsf{d}_{\Diamond,\Box}(\varphi_i)$. We distinguish two cases:

- $\pi = x_0 \ldots, x_k$ does *not* visit nodes which are both propositional and secondary. We show that for all $0 < j \leq k$, $\mathsf{d}_{\Diamond,\Box}(L_i(x_j)) < \mathsf{d}_{\Diamond,\Box}(L_i(x_{j-1}))$, hence, since $\mathsf{d}_{\Diamond,\Box}(L_i(x_0)) = \mathsf{d}_{\Diamond,\Box}(\varphi_i)$, the result follows. We assume the contrary and derive a contradiction. Then, there is $0 < j \leq k$ such that $\mathsf{d}_{\Diamond,\Box}(L_i(x_j)) \geq \mathsf{d}_{\Diamond,\Box}(L_i(x_{j-1}))$. There are two subcases:
  - x_j is a main node. Hence, there must be $\Diamond\psi \in L_i(x_{j-1})$ such that $\psi \in L_i(x_j)$. Moreover, for each $\psi' \in L_i(x_j) \setminus (P \cup \overline{P})$, either $\Diamond\psi' \in L_i(x_{j-1})$, or $\Box\psi' \in L_i(x_{j-1})$, or ψ' has been added to $L_i(x_j)$ by an application of the $\vee$-rule or $\wedge$-rule starting from a formula $\psi'' \in L_i(x_j)$ such that ψ' is a strict subformula of ψ''. It follows that $\mathsf{d}_{\Diamond,\Box}(L_i(x_j)) < \mathsf{d}_{\Diamond,\Box}(L_i(x_{j-1}))$, which is a contradiction.
  - x_j is a secondary node. By hypothesis, x_j is *not* a propositional node. Since when a secondary node is created, its label contains only atomic propositions, by the $\Box$-rule, there must be $\Box\psi \in L_i(x_{j-1})$ such that $\psi \in L_i(x_j)$. Moreover, for each $\psi' \in L_i(x_j) \setminus (P \cup \overline{P})$, either $\Box\psi' \in L_i(x_{j-1})$, or ψ' has been added to $L_i(x_j)$ by an application of the $\vee$-rule or $\wedge$-rule starting from a formula $\psi'' \in L_i(x_j)$ such that ψ' is a strict subformula of ψ''. It follows that $\mathsf{d}_{\Diamond,\Box}(L_i(x_j)) < \mathsf{d}_{\Diamond,\Box}(L_i(x_{j-1}))$, which is a contradiction.
- $\pi = x_0 \ldots, x_k$ visits some node which is both propositional and secondary. It easily follows that node x_k is both propositional and secondary. Hence, there is exactly one pair $\langle i', x'_k \rangle$ such that ${\leftarrow_{i'}}(x'_k) = \langle i, x_k \rangle$. Moreover, $x'_k \neq \varepsilon$. Let π' be the partial path of $T_{i'}$ from the root to node x'_k. Since $\mathcal{S}$ is a constraint system for φ, by Property 2 in Definition 2, there is $0 \leq q < k$ such that π' can be written in the form $\pi' = x'_q, x'_{q+1}, \ldots, x'_k$, where $x'_q = \varepsilon$ and ${\leftarrow_{i'}}(x'_l) = \langle i, x_l \rangle$ for all $q \leq l \leq k$. In particular, $i' \in {\rightarrow_i}(x_q)$ and $\exists_r\varphi_{i'} \in L_i(x_q)$. Since $i' > i$, by the induction hypothesis, the length of π' is bounded by $\mathsf{d}_{\Diamond,\Box}(\varphi_{i'})$. Since $\mathsf{d}_{\Diamond,\Box}(\varphi_{i'}) \leq \mathsf{d}_{\Diamond,\Box}(L_i(x_q))$ ($\exists_r\varphi_{i'} \in L_i(x_q)$), we obtain that the suffix $x_q, \ldots, x_k$ of π has length bounded by $\mathsf{d}_{\Diamond,\Box}(L_i(x_q))$. Since the prefix $x_0, \ldots, x_q$ of π does not visit nodes which are both propositional and secondary (x_q is not propositional), by the first case, it holds that for all $0 < j \leq q$, $\mathsf{d}_{\Diamond,\Box}(L_i(x_j)) < \mathsf{d}_{\Diamond,\Box}(L_i(x_{j-1}))$. Hence, the length of π is bounded by $\mathsf{d}_{\Diamond,\Box}(L_i(x_0)) = \mathsf{d}_{\Diamond,\Box}(\varphi_i)$ which concludes.

□

Claim 3: For all $1 \leq i \leq \mathsf{dim}(\mathcal{S})$ and $x \in T_i$, the out-degree of x in T_i (i.e., the number of successors of x in T_i) is at most $|\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)+1)}$.

Proof of Claim 3: the proof is by induction on $\mathsf{dim}(\mathcal{S}) - i$. For the base case ($i = \mathsf{dim}(\mathcal{S})$), it holds that every node of T_i is a main node. Let $x_1, \ldots, x_k$ be the successors of x in T_i (which are created by applications of the $\Diamond$-rule). By the precondition of the $\Diamond$-rule, it follows that there are pairwise distinct k formulas $\Diamond\psi_1, \ldots, \Diamond\psi_k \in L_i(x)$. Since $L_i(x) \subseteq \mathsf{cl}(\varphi_i)$ (Claim 1), it follows that the out-degree of x in T_i is bounded by $|\varphi_i|$. Hence, in this case the result holds. Now, assume that $i < \mathsf{dim}(\mathcal{S})$. Assume that $|\varphi_i| \geq 2$ (otherwise, the result is obvious) and x is a secondary node (the other case being simpler). Then, there is exactly one pair $\langle j, z\rangle$ such that $z \neq \varepsilon$ and $\leftarrow_j(z) = \langle i, x\rangle$ (in particular, x and z are created by the same application of the $\Diamond$-rule). Thus, since $\mathcal{S}$ is constraint system, there is an ancestor x_a of x in T_i such that $\leftarrow_j(\varepsilon) = \langle i, x_a\rangle$, hence $j \in \rightarrow_i(x_a)$. Let $N_{main}(x)$ be the number of successors of x which are main nodes. By reasoning as in the base case, $N_{main}(x) \leq |\varphi_i|$. Then, by construction it follows that:

$$\text{out-degree of } x = N_{main}(x) + N_j(z) + \sum_{l \in \rightarrow_i(x)} N_l(\varepsilon)$$

where $N_j(z)$ is the out-degree of z in T_j, and for all $l \in \rightarrow_i(x)$, $N_l(\varepsilon)$ is the out-degree of the T_l-root. Since $j > i$ and $l > i$ for all $l \in \rightarrow_i(x)$, by the induction hypothesis we obtain (note that $\mathsf{d}_\exists(\varphi_i) > 1$)

$$\text{out-degree of } x \leq |\varphi_i| + |\varphi_j|^{(2\mathsf{d}_\exists(\varphi_j)+1)} + \sum_{l \in \rightarrow_i(x)} |\varphi_l|^{(2\mathsf{d}_\exists(\varphi_l)+1)} \leq \text{ (by Claim 1 and since } j \in \rightarrow_i(x_a))$$

$$|\varphi_i| + |\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)-1)} + \sum_{l \in \rightarrow_i(x)} |\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)-1)} \leq \qquad \text{(by Claim 1)}$$

$$|\varphi_i| + |\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)-1)} + |\varphi_i| \cdot |\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)-1)} \leq 2 \cdot |\varphi_i| \cdot |\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)-1)} \leq \qquad (|\varphi_i| \geq 2)$$

$$|\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)+1)}$$

which concludes. □

For each $1 \leq i \leq \mathsf{dim}(\mathcal{S})$, let $H(\mathcal{S}, i)$ be the set of *i-descendants* in $\mathcal{S}$ defined as $H(\mathcal{S}, i) := \{i\} \cup \{j \in H(\mathcal{S}, h) \mid h \in \rightarrow_i(x) \text{ for some } x \in T_i\}$. Note that $H(\mathcal{S}, i)$ is well defined since for all $x \in T_i$ and $h \in \rightarrow_i(x)$, $h > i$. Moreover, $\mathsf{dim}(\mathcal{S}) = |H(\mathcal{S}, 1)|$.

Claim 4: $|H(\mathcal{S}, i)| \leq |\varphi_i|^{4 \cdot (\mathsf{d}_\exists(\varphi_i))^2 \cdot (\mathsf{d}_{\Diamond,\Box}(\varphi_i)+1)}$ for all $1 \leq i \leq \mathsf{dim}(\mathcal{S})$.

Proof of Claim 4: the proof is by induction on $\mathsf{dim}(\mathcal{S}) - i$. For the base case ($i = \mathsf{dim}(\mathcal{S})$), $H(\mathcal{S}, i) = \{i\}$. Hence, the result follows. Now, assume that $i < \mathsf{dim}(\mathcal{S})$. If $\mathsf{d}_\exists(\varphi_i) = 0$, then $H(\mathcal{S}, i) = \{i\}$ and the result holds. Now, let $\mathsf{d}_\exists(\varphi_i) \geq 1$. By Claims 2 and 3, $|T_i| \leq |\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)+1) \cdot \mathsf{d}_{\Diamond,\Box}(\varphi_i)}$ and by Claim 1, for each $x \in T_i$, $|\rightarrow_i(x)| \leq |\varphi_i|$. Then:

$$|H(\mathcal{S}, i)| = 1 + \sum_{x \in T_i, h \in \rightarrow_i(x)} |H(\mathcal{S}, h)| \leq \qquad \text{(by the induction hypothesis)}$$

$$1 + \sum_{x \in T_i, h \in \rightarrow_i(x)} |\varphi_h|^{4 \cdot (\mathsf{d}_\exists(\varphi_h))^2 \cdot (\mathsf{d}_{\Diamond,\Box}(\varphi_h)+1)} \leq \qquad \text{(by Claim 1)}$$

$$1 + \sum_{x \in T_i, h \in \rightarrow_i(x)} |\varphi_i|^{4 \cdot (\mathsf{d}_\exists(\varphi_i)-1)^2 \cdot (\mathsf{d}_{\Diamond,\Box}(\varphi_i)+1)} \leq \qquad (|T_i| \leq |\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)+1) \cdot \mathsf{d}_{\Diamond,\Box}(\varphi_i)} \text{ and } |\rightarrow_i(x)| \leq |\varphi_i|)$$

$$1 + |\varphi_i| \cdot |\varphi_i|^{(2\mathsf{d}_\exists(\varphi_i)+1)\cdot \mathsf{d}_{\Diamond,\Box}(\varphi_i)} \cdot |\varphi_i|^{4\cdot(\mathsf{d}_\exists(\varphi_i)-1)^2\cdot(\mathsf{d}_{\Diamond,\Box}(\varphi_i)+1)} \leq$$
$$1 + |\varphi_i|^{(\mathsf{d}_{\Diamond,\Box}(\varphi_i)+1)\cdot(4\cdot(\mathsf{d}_\exists(\varphi_i))^2+5-6\mathsf{d}_\exists(\varphi_i))} \leq \qquad (\text{since } \mathsf{d}_\exists(\varphi_i) \geq 1, \text{ hence, } |\varphi_i| > 1)$$
$$|\varphi_i|^{4\cdot(\mathsf{d}_\exists(\varphi_i))^2\cdot(\mathsf{d}_{\Diamond,\Box}(\varphi_i)+1)}.$$

Hence, the result follows. □

B.3 Detailed proof of Lemma 3

Let $\mathcal{S} = \langle\langle T_1, L_1, \leftarrow_1\rangle, \ldots, \langle T_n, Ł_n, \leftarrow_n\rangle\rangle$, $\mathcal{S}' = \langle\langle T'_1, L'_1, \leftarrow'_1\rangle, \ldots, \langle T'_m, L'_m, \leftarrow'_m\rangle\rangle$, and $\mathcal{T} = \langle\uparrow_1, \ldots, \uparrow_n\rangle$ as in Definition 5. We also say that $\mathcal{S}$ is a refinement of $\mathcal{S}'$ w.r.t. $\mathcal{T}$. Lemma 3 directly follows from the following two lemmata.

Lemma 5. *Let $\mathcal{S}$ and $\mathcal{S}'$ be two constraint systems for φ such that $\mathcal{S}'$ is well-formed and $\mathcal{S}$ is a refinement of $\mathcal{S}'$. Then, $\mathcal{S}$ is well-formed as well, and the main structure of $\mathcal{S}$ is a refinement of the main structure of $\mathcal{S}'$. Moreover, if $\mathcal{S}'$ is additionally semantically $\forall_r$-consistent, then $\mathcal{S}$ is semantically $\forall_r$-consistent too.*

Proof. Let $\mathcal{S} = \langle\langle T_1, L_1, \leftarrow_1\rangle, \ldots, \langle T_n, L_n, \leftarrow_n\rangle\rangle$ and $\mathcal{S}' = \langle\langle T'_1, L'_1, \leftarrow'_1\rangle, \ldots, \langle T'_m, L'_m, \leftarrow'_m\rangle\rangle$ as in the statement of the lemma, and $\mathcal{T} = \langle\uparrow_n, \ldots, \uparrow_n\rangle$ such that $\mathcal{S}$ is a refinement of $\mathcal{S}'$ w.r.t. $\mathcal{T}$. First, we show that $\mathcal{S}$ is well-formed as well. Let $i \leq \mathsf{dim}(\mathcal{S})$ and $x \in T_i$ such that $\uparrow_i(x) = \langle j, y\rangle$. By Property 1 of Definition 5, $L_i(x) \subseteq L'_j(y)$. Since $\mathcal{S}'$ is clash-free ($\mathcal{S}'$ is well-formed) and i and x are arbitrary, it follows that $\mathcal{S}$ is clash-free as well. Moreover, since $L_i(x)$ is complete, we also obtain that $\widetilde{L}_i(x) = \widetilde{L}'_j(y)$. Now, assume that $i > 1$ and $\leftarrow_i(x) = \langle i', x'\rangle$. We show that $\widetilde{L}_i(x) = \widetilde{L}_{i'}(x')$, hence $\mathcal{S}$ is well-formed. By Property 3 of Definition 5, $\leftarrow'_j(y) = \langle j', y'\rangle$ and $\uparrow_{i'}(x') = \langle j', y'\rangle$. Then, by the above considerations, $\widetilde{L}_{i'}(x') = \widetilde{L}'_{j'}(y')$. Moreover, since $\mathcal{S}'$ is well-formed, $\widetilde{L}'_j(y) = \widetilde{L}'_{j'}(y')$. Thus, since $\widetilde{L}_i(x) = \widetilde{L}'_j(y)$, it follows that $\widetilde{L}_i(x) = \widetilde{L}_{i'}(x')$, and the result holds. Moreover, by Property 2 of Definition 5, we obtain the following.

Claim: for all $i \leq \mathsf{dim}(\mathcal{S})$ and $x \in T_i$ such that $\uparrow_i(x) = \langle j, y\rangle$, $\langle T_i, \widetilde{L}_i\rangle_x$ is a refinement of $\langle T'_j, \widetilde{L}'_j\rangle_y$.

By Property 1 of Definition 5, $\uparrow_1(\varepsilon) = \langle 1, \varepsilon\rangle$. Thus, by the claim above, it follows that $\langle T_1, \widetilde{L}_1\rangle$ (the main structure of $\mathcal{S}$) is a refinement of $\langle T'_1, \widetilde{L}'_1\rangle$ (the main structure of $\mathcal{S}'$), which concludes the proof of the first part of the lemma. For the second part of the lemma, assume that $\mathcal{S}'$ is additionally semantically $\forall_r$-consistent. Let $1 \leq i \leq n$, $x \in T_i$, and $\forall_r\psi \in L_i(x)$. We need to show that the tree structure $\langle T_i, \widetilde{L}_i\rangle_x$ satisfies $\forall_r\psi$. Let $\uparrow_i(x) = \langle j, y\rangle$. By Property 1 of Definition 5, $\forall_r\psi \in L'_j(y)$. Thus, since $\mathcal{S}'$ is semantically $\forall_r$-consistent, it holds that the tree structure $\langle T'_j, \widetilde{L}'_j\rangle_y$ satisfies $\forall_r\psi$. By the claim above, $\langle T_i, \widetilde{L}_i\rangle_x$ is a refinement of $\langle T'_j, \widetilde{L}'_j\rangle_y$. It follows that $\langle T_i, \widetilde{L}_i\rangle_x$ satisfies $\forall_r\psi$ as well (recall that refinement is a preorder), which concludes the proof of the lemma. □

Lemma 6. *Let $\mathcal{S}'$ be a constraint system for φ which is well-formed and saturated. Then, there exists a* minimal *and saturated constraint system $\mathcal{S}$ for φ which is a refinement of $\mathcal{S}'$.*

Proof. Let $\mathcal{S}' = \langle\langle T'_1, L'_1, \leftarrow'_1\rangle, \ldots, \langle T'_m, L'_m, \leftarrow'_m\rangle\rangle$ as in the statement of the lemma. First, for each rule of Definition 3, we define an extension of such a rule which has the same precondition and the same effect (w.r.t. a given constraint system $\mathcal{S}$ for φ) with the difference that the nondeterministic

choices are guided by $\mathcal{S}'$. Moreover, these new rules are applicable to pairs $(\mathcal{S},\mathcal{T})$ such that $\mathcal{S}$ is a refinement of $\mathcal{S}'$ w.r.t. $\mathcal{T}$, and their application preserves this last condition. These new rules are defined as follows, where for each rule, the parts which are not present in the corresponding rule of Definition 3 are underlined (for the rest, the two rules are identical).

Assumption: $\mathcal{S}$ is a refinement of $\mathcal{S}'$ w.r.t. $\mathcal{T}$.

$\wedge$-rule: `if` $\mathcal{S} \vdash \langle i,x,\psi_1 \wedge \psi_2\rangle$, $\mathcal{S}(i) = \langle T,L,\leftarrow\rangle$, and $\{\psi_1,\psi_2\} \not\subseteq L(x)$
`then` *update* $L(x) := L(x) \cup \{\psi_1,\psi_2\}$

$\vee$-rule: `if` $\mathcal{S} \vdash \langle i,x,\psi_1 \vee \psi_2\rangle$, $\mathcal{S}(i) = \langle T,L,\leftarrow\rangle$, and $\{\psi_1,\psi_2\} \cap L(x) = \emptyset$
`then` *update* $L(x) := L(x) \cup \{\psi_k\}$ for some $k \in \{1,2\}$
such that $\psi_k \in L'_j(y)$ where $\uparrow_i(x) = \langle j,y\rangle$ and $\mathcal{T} = \langle\uparrow_1,\ldots,\uparrow_n\rangle$
/** note that such a k exists since $L(x) \subseteq L'_j(y)$ and $\mathcal{S}'$ is saturated **/

$\exists_r$-rule: `if` $\mathcal{S} \vdash \langle i,x,\exists_r\psi\rangle$, $\mathcal{S} := \langle\langle T_1,L_1,\leftarrow_1\rangle,\ldots,\langle T_n,L_n,\leftarrow_n\rangle\rangle$, and
$\mathcal{S} \nvdash \langle h,\varepsilon,\psi\rangle$ for each $h \leq \mathsf{dim}(\mathcal{S})$ such that $\leftarrow_h(\varepsilon) = \langle i,x\rangle$
`then` let $\mathcal{T} = \langle\uparrow_1,\ldots,\uparrow_n\rangle$, $\uparrow_i(x) = \langle j,y\rangle$, and $h \leq \mathsf{dim}(\mathcal{S}')$ s.t. $\leftarrow'_h(\varepsilon) = \langle j,y\rangle$ and $\psi \in L'_h(\varepsilon)$
/** note that such a h exists since $L_i(x) \subseteq L'_j(y)$ and $\mathcal{S}'$ is saturated **/
update $\mathcal{S} := \langle\langle T_1,L_1,\leftarrow_1\rangle,\ldots,\langle T_{n+1},L_{n+1},\leftarrow_{n+1}\rangle\rangle$, $\mathcal{T} := \langle\uparrow_1,\ldots,\uparrow_{n+1}\rangle$,
where $T_{n+1} := \{\varepsilon\}$, $L_{n+1}(\varepsilon) := \{\psi\} \cup (L_i(x) \cap (P \cup \overline{P}))$, $\leftarrow_{n+1}(\varepsilon) := \langle i,x\rangle$,
and $\uparrow_{n+1}(\varepsilon) := \langle h,\varepsilon\rangle$.
/** since $\mathcal{S}'$ is well-formed and $L_i(x) \subseteq L'_j(y)$, $L_{n+1}(\varepsilon) \subseteq L'_h(\varepsilon)$ **/

$\Box$-rule: `if` $\mathcal{S} \vdash \langle i,x,\Box\psi\rangle$ and $\mathcal{S} \nvdash \langle i,x',\psi\rangle$ for some successor x' of x in $\mathcal{S}(i)$
`then` *let* $\mathcal{S}(i) = \langle T,L,\leftarrow\rangle$
update $L(x') := L(x') \cup \{\psi\}$ for each successor x' of x in T

$\Diamond$-rule: `if` $\mathcal{S} \vdash \langle i,x,\Diamond\psi\rangle$ and $\mathcal{S} \nvdash \langle i,x',\psi\rangle$ for each successor x' of x in $\mathcal{S}(i)$
`then` let $\langle i_0,x_0\rangle \leftarrow_{\mathcal{S}} \ldots \leftarrow_{\mathcal{S}} \langle i_k,x_k\rangle$ with $i_0 = 1$ and $\langle i_k,x_k\rangle = \langle i,x\rangle$
let $\mathcal{T} = \langle\uparrow_1,\ldots,\uparrow_n\rangle$, $\uparrow_{i_0}(x_0) = \langle j_0,y_0\rangle,\ldots,\uparrow_{i_k}(x_k) = \langle j_k,y_k\rangle$
/** note that $j_0 = 1$ and $\langle j_0,y_0\rangle \leftarrow_{\mathcal{S}'} \ldots \leftarrow_{\mathcal{S}'} \langle j_k,y_k\rangle$ **/
let y'_k be a successor of y_k in T'_{j_k} such that $\psi \in L'_{j_k}(y'_k)$
/** such a y'_k exists since $L_{i_k}(x_k) \subseteq L'_{j_k}(y_k)$ (L_{i_k} is the labeling of $\mathcal{S}(i_k)$) and
$\mathcal{S}'$ is saturated **/
let $\langle j_0,y'_0\rangle \leftarrow_{\mathcal{S}'} \ldots \leftarrow_{\mathcal{S}'} \langle j_k,y'_k\rangle$
/** note that y'_q is a successor of y_q in T'_{j_q} for all $1 \leq q \leq k$ **/
guess some *complete* set $\chi \subseteq P \cup \overline{P}$ such that $\chi = L'_{j_k}(y'_k) \cap (P \cup \overline{P})$
/** $L'_{j_0}(y'_0) \cap (P \cup \overline{P}) = \ldots = L'_{j_k}(y'_k) \cap (P \cup \overline{P}) = \chi$ since $\mathcal{S}'$ is well-formed **/
`for each` $q = k,k-1,\ldots,0$ with $\mathcal{S}(i_q) = \langle T_q,L_q,\leftarrow_q\rangle$ `do`
update $T_q := T_q \cup \{x_q \cdot h_q\}$ for some $h_q \in \mathbb{N}$ such that $x_q \cdot h_q \notin T_q$,
$\uparrow_{i_q}(x_q \cdot h_q) := \langle j_q,y'_q\rangle$
`if` $q < k$ `then` $L_q(x_q \cdot h_q) := \chi$ and $\leftarrow_{i_{q+1}}(x_{q+1} \cdot h_{q+1}) := \langle i_q,x_q \cdot h_q\rangle$
`else` $L_q(x_q \cdot h_q) := \{\psi\} \cup \chi$

By construction, we easily obtain the following.

Claim: Let $(\mathcal{S},\mathcal{T})$ be such that $\mathcal{S}$ is a refinement of $\mathcal{S}'$ w.r.t. $\mathcal{T}$, R be any of the rules defined above, and $(\mathcal{S}'',\mathcal{T}'')$ obtained from $(\mathcal{S},\mathcal{T})$ by applying rule R. Then, $\mathcal{S}''$ is a refinement of $\mathcal{S}'$ w.r.t. $\mathcal{T}''$.

Since $\mathcal{S}'$ is well-formed, there is a unique *initial* constraint system $\mathcal{S}_0 = \langle\{\varepsilon\}, L, \leftarrow\rangle$ for φ s.t. $\mathcal{S}_0$ is a refinement of $\mathcal{S}'$. Let $\mathcal{T}_0 = \langle\uparrow_1\rangle$ with $\uparrow_1$: $\{\varepsilon\} \mapsto \{1\} \times \{\varepsilon\}$. Note that $\mathcal{S}_0$ is a refinement of $\mathcal{S}'$ w.r.t. $\mathcal{T}_0$. Now, every rule defined above has the same *precondition* and the same *effect* (w.r.t. a constraint system $\mathcal{S}$ for φ) of the corresponding rule in Definition 3. Hence, by the claim above and Lemma 2, any sequence of applications of the rules defined above starting from $(\mathcal{S}_0,\mathcal{T}_0)$ is *finite* and leads to a pair $(\mathcal{S},\mathcal{T})$ s.t.: (i) $\mathcal{S}$ is a refinement of $\mathcal{S}'$ w.r.t. $\mathcal{T}$, and (ii) $\mathcal{S}'$ can be obtained from $\mathcal{S}_0$ by a sequence of applications of the rules of Definition 3. Hence, there is a *minimal* and saturated constraint system $\mathcal{S}$ for φ which is a refinement of $\mathcal{S}'$. This concludes the proof of the lemma. □

C Proofs from Section 4

In this section, for RML formulas φ and ψ, we use $\varphi \to \psi$ as an abbreviation for $\widetilde{\varphi} \vee \psi$.

C.1 Proof of Proposition 2

First, we briefly recall the framework of Alternating Turing Machines (ATM), see [4] for more details. An ATM $\mathcal{M}$ allows both existential choices (nondeterminism) and universal choices (parallelism). The set of $\mathcal{M}$-states is partitioned into a set of existential states and a set of universal states. This partition induces an analogous partition on the set of configurations in accordance with the associated state. The acceptance criterium of $\mathcal{M}$ can be defined via a reachability two-player turn-based game between player Eve and player Adam, where existential (resp., universal) configurations are controlled by Eve (resp., Adam). A configuration C of $\mathcal{M}$ *leads to acceptance* iff there is a strategy of Eve from C which allows to select only computations (from C) ending in an accepting configuration. An input word α is *accepted* by $\mathcal{M}$ iff the initial configuration associated with α leads to acceptance. The ATM $\mathcal{M}$ is singly exponential-time bounded if there is an integer constant $c \geq 1$ such that for each input α, when started on α, no matter what are the universal and existential choices, $\mathcal{M}$ halts in at most $2^{|\alpha|^c}$ steps. The ATM $\mathcal{M}$ has a polynomial-bounded number of alternations if there is an integer constant $c \geq 1$ s.t. for all inputs α and computations π from α, the number of alternations of existential and universal configurations along π is at most $|\alpha|^c$.

Now, we prove Proposition 2. The upper bounds are easy, while the lower bounds directly follows from the following two lemmata.

Lemma 7. *Let $k \geq 1$ and $\mathcal{M}_{\mathcal{A}}$ be a singly exponential-time bounded ATM with at most $k-1$ alternations and such that the initial state is existential. Moreover, let $c \geq 1$ be an integer constant such that for each input α, when started on α, $\mathcal{M}_{\mathcal{A}}$ reaches a terminal configuration in at most $2^{|\alpha|^c}$ steps. Then, given an input α, one can construct in time polynomial in α and the size of $\mathcal{M}_{\mathcal{A}}$ an instance $(k, 2|\alpha|^c, \mathcal{M})$ of the* Alternation Problem *s.t. the instance is positive* iff *$\mathcal{M}_{\mathcal{A}}$ accepts α.*

Lemma 8. *Let $\mathcal{M}_{\mathcal{A}}$ be a singly exponential-time bounded ATM with a polynomial-time bounded number alternations. Moreover, let $c \geq 1$ and $c_a \geq 1$ be integer constants such that for each input α, when started on α, $\mathcal{M}_{\mathcal{A}}$ has at most $|\alpha|^{c_a}$ alternations and $\mathcal{M}_{\mathcal{A}}$ reaches a terminal configuration*

in at most $2^{|\alpha|^c}$ *steps. Then, given an input* α*, one can construct in time polynomial in* α *and the size of* $\mathcal{M}_{\mathcal{A}}$ *an instance* $(2|\alpha|^{max\{c,c_a\}}, 2|\alpha|^{max\{c,c_a\}}, \mathcal{M})$ *of the* Alternation Problem *such that the instance is positive* iff $\mathcal{M}_{\mathcal{A}}$ *accepts* α*.*

The proofs of Lemmata 7 and 8 are very similar, so that we prove only Lemma 7.

Proof of Lemma 7. Let $\mathcal{M}_{\mathcal{A}}$, c, and k as in the statement of Lemma 7. Let $I_{\mathcal{A}}$ (resp., $A_{\mathcal{A}}$) be the input (resp., work) alphabet of $\mathcal{M}_{\mathcal{A}}$, where $I_{\mathcal{A}} \subset A_{\mathcal{A}}$, and Q be the set of $\mathcal{M}_{\mathcal{A}}$-states. Fix an input $\alpha \in I_{\mathcal{A}}^*$. An α-configuration is a word over $I_{\mathcal{A}} \times (Q \times I_{\mathcal{A}}) \times I_{\mathcal{A}}$ of length exactly $2^{|\alpha|^c}$. Note that any configuration of $\mathcal{M}_{\mathcal{A}}$ reachable from the input α can be encoded by an α-configuration. We denote by C_α the initial (existential) α-configuration associated with the input α. A *partial computation* of $\mathcal{M}_{\mathcal{A}}$ is a finite sequence $\pi = C_1, \ldots, C_p$ of α-configurations such that $p \leq 2^{|\alpha|^c}$ and for each $1 \leq i < p$, C_{i+1} is a $\mathcal{M}_{\mathcal{A}}$-successor of C_i (note that a computation of $\mathcal{M}_{\mathcal{A}}$ over α is a partial computation). We say that π is *uniform* if additionally, one of the following holds:

- C_p is terminal and π visits only existential n-configurations;
- C_p is terminal and π visits only universal n-configurations;
- $p > 1$, C_p is existential and for each $1 \leq h < p$, C_h is universal;
- $p > 1$, C_p is universal and for each $1 \leq h < p$, C_h is existential.

Let $\diamondsuit$ be a fresh symbol and $I = A_{\mathcal{A}} \cup \{\diamondsuit\}$. The *code* of a partial computation $\pi = C_1, \ldots, C_p$ is the word over I of length exactly $2^{2|\alpha|^c}$ given by $C_1, \ldots, C_p, C^0_{p+1}, \ldots, C^0_{2^{|\alpha|^c}}$, where $C^0_h \in \{\diamondsuit\}^{2^{|\alpha|^c}}$ for all $p+1 \leq h \leq 2^{|\alpha|^c}$. We construct a polynomial-time bounded k-ary deterministic Turing Machine $\mathcal{M}$, which satisfies Lemma 7 for the given input α of $\mathcal{M}_{\mathcal{A}}$, as follows. The input alphabet of $\mathcal{M}$ is I. Given a k-ary input $(w_1, \ldots, w_k) \in (I^*)^k$, $\mathcal{M}$ operates in k-steps. At step ith ($1 \leq i \leq k$), the behavior of $\mathcal{M}$ is as follows, where $n = 2|\alpha|^c$ and for a partial computation $\pi = C_1, \ldots, C_p$, $\mathsf{first}(\pi) = C_1$ and $\mathsf{last}(\pi) = C_p$:

- *First step:* $i = 1$.
  1. If $w_1 \in I^{2^n}$ and w_1 encodes a uniform partial computation π_1 of $\mathcal{M}_{\mathcal{A}}$ from C_α, then the behavior is as follows. If $\mathsf{last}(\pi_1)$ is accepting (resp., rejecting), then $\mathcal{M}$ accepts (resp., rejects) the input . If instead $\mathsf{last}(\pi_1)$ is not a terminal configuration, then $\mathcal{M}$ goes to step $i+1$.
  2. Otherwise, $\mathcal{M}$ *rejects* the input.
- $i > 1$.
  1. If $w_i \in I^{2^n}$ and w_i encodes a uniform partial computation π_i of $\mathcal{M}_{\mathcal{A}}$ such that $\mathsf{first}(\pi_i) = \mathsf{last}(\pi_{i-1})$, where π_{i-1} is the uniform partial computation encoded by w_{i-1}, then the behavior is as follows. If $\mathsf{last}(\pi_i)$ is accepting (resp., rejecting), then $\mathcal{M}$ accepts (resp., rejects) the input. If instead $\mathsf{last}(\pi_i)$ is not a terminal configuration, then $\mathcal{M}$ goes to step $i+1$.
  2. Otherwise, if i is odd (resp., even), then $\mathcal{M}$ *rejects* (resp., *accepts*) the input.

Note that Conditions 1 in the steps above can be checked by $\mathcal{M}$ in polynomial time (in the size of the input) by using the transition function of $\mathcal{M}_{\mathcal{A}}$ and n-bit counters. Hence, $\mathcal{M}$ is a polynomial-time bounded k-ary deterministic Turing Machine which, evidently, can be constructed in time polynomial in n and the size of $\mathcal{M}_{\mathcal{A}}$. Now, we prove that the construction is correct, i.e. $(k, n, \mathcal{M})$ is a positive instance of the Alternation Problem iff $\mathcal{M}_{\mathcal{A}}$ accepts α. For each $1 \leq \ell \leq k$, let $\mathsf{Q}_\ell = \exists$ if ℓ is odd, and $\mathsf{Q}_\ell = \forall$ otherwise. Since C_α is existential, $\mathcal{M}_{\mathcal{A}}$ accepts α iff there is a non-rejecting uniform partial computation π_1 of $\mathcal{M}_{\mathcal{A}}$ from C_α such that $\mathsf{last}(\pi_1)$ leads to acceptance. Moreover, for

each $w_1 \in I^{2^n}$, $\mathcal{M}$ accepts an input of the form $(w_1, w'_2, \ldots, w'_k)$ *only if* w_1 encodes a non-rejecting uniform partial computation of $\mathcal{M}_{\mathcal{A}}$ from C_α. Thus, since $\mathsf{Q}_1 = \exists$, correctness of the construction directly follows from the following claim.

Claim: let $1 \leq \ell \leq k$ and $\pi = \pi_1 \ldots \pi_\ell$ be a partial computation of $\mathcal{M}_{\mathcal{A}}$ from C_α such that π_ℓ is uniform and for each $1 \leq h < \ell$, π_h is non-empty and $\pi_h \cdot \mathsf{first}(\pi_{h+1})$ is uniform as well. Let $w_\ell \in I^{2^n}$ be the word encoding π_ℓ and for each $1 \leq h < l$, $w_h \in I^{2^n}$ be the word encoding $\pi_h \cdot \mathsf{first}(\pi_{h+1})$. Then, $\mathsf{last}(\pi_\ell)$ leads to acceptance if and only if

$$\mathsf{Q}_{\ell+1} x_{\ell+1} \in I^{2^n}. \ldots \mathsf{Q}_k x_k \in I^{2^n}. \mathcal{M}(w_1, \ldots, w_\ell, x_{\ell+1}, \ldots, x_k) \tag{1}$$

Proof of the claim: the proof is by induction on $k - \ell$.

Base Step: $\ell = k$. Note that in this case $\mathsf{last}(\pi_k)$ is a terminal configuration (otherwise, the number of alternations of existential and universal configurations along π would be greater than $k-1$). Thus, we need to show that $\mathsf{last}(\pi_k)$ is accepting iff $\mathcal{M}(w_1, \ldots, w_k)$. By construction, when started on the input $(w_1, \ldots, w_k)$, $\mathcal{M}$ reaches the kth step and Condition 1 in this step is satisfied. Moreover, either $\mathsf{last}(\pi_k)$ is accepting and $\mathcal{M}$ accepts the input $(w_1, \ldots, w_k)$, or $\mathsf{last}(\pi_k)$ is rejecting and $\mathcal{M}$ rejects the input $(w_1, \ldots, w_k)$. Hence, the result follows.

Induction Step: $\ell < k$. First, assume that $\mathsf{last}(\pi_\ell)$ is a terminal configuration. By construction on any input of the form $(w_1, \ldots, w_\ell, w'_{\ell+1}, \ldots, w'_k)$, $\mathcal{M}$ reaches the ℓth step and Condition 1 in this step is satisfied. Moreover, either $\mathsf{last}(\pi_\ell)$ is accepting and $\mathcal{M}$ accepts the input $(w_1, \ldots, w_\ell, w'_{\ell+1}, \ldots, w'_k)$, or $\mathsf{last}(\pi_\ell)$ is rejecting and $\mathcal{M}$ rejects the input $(w_1, \ldots, w_\ell, w'_{\ell+1}, \ldots, w'_k)$. Hence, in this case the result holds. Now, assume that $\mathsf{last}(\pi_\ell)$ is not terminal. We distinguish two cases:

- $\ell + 1$ is even: hence $\mathsf{Q}_{\ell+1} = \forall$. Since C_α is existential and $\mathsf{last}(\pi_\ell)$ is not terminal, by hypothesis, $\mathsf{last}(\pi_\ell)$ must be an universal configuration. First, assume that $\mathsf{last}(\pi_\ell)$ leads to acceptance. Let $w_{\ell+1} \in I^{2^n}$. By construction on any input of the form $(w_1, \ldots, w_\ell, w_{\ell+1}, w'_{\ell+2} \ldots, w'_k)$, $\mathcal{M}$ reaches the $(\ell+1)$th step. If $w_{\ell+1}$ satisfies Condition 2 in this step, then since $\ell + 1$ is even, $\mathcal{M}$ accepts the input. Hence, $\mathsf{Q}_{\ell+2} x_{\ell+2} \in I^{2^n}. \ldots \mathsf{Q}_k x_k \in I^{2^n}. \mathcal{M}(w_1, \ldots, w_\ell, w_{\ell+1}, x_{\ell+2}, \ldots, x_k)$. Otherwise, $w_{\ell+1}$ encodes a uniform partial computation $\pi_{\ell+1}$ of $\mathcal{M}_{\mathcal{A}}$ from $\mathsf{last}(\pi_\ell)$. Since $\mathsf{last}(\pi_\ell)$ leads to acceptance and $\mathsf{last}(\pi_\ell)$ is universal, $\mathsf{last}(\pi_{\ell+1})$ leads to acceptance as well. Thus, applying the induction hypothesis to the partial computation $\pi_1 \ldots \pi_{\ell-1} \pi'_\ell \pi_{\ell+1}$ (where π'_ℓ is obtained from π_ℓ by removing $\mathsf{last}(\pi_\ell)$), it follows that $\mathsf{Q}_{\ell+2} x_{\ell+2} \in I^{2^n}. \ldots \mathsf{Q}_k x_k \in I^{2^n}. \mathcal{M}(w_1, \ldots, w_\ell, w_{\ell+1}, x_{\ell+2}, \ldots, x_k)$. Thus, the previous condition holds for each $w_{\ell+1} \in I^{2^n}$. Since $Q_{\ell+1} = \forall$, it follows that Condition (1) holds. For the converse direction, assume that Condition (1) holds. Let $\pi_{\ell+1}$ be any uniform partial computation of $\mathcal{M}_{\mathcal{A}}$ from $\mathsf{last}(\pi_\ell)$. We need to show that $\mathsf{last}(\pi_{\ell+1})$ leads to acceptance. Since Condition (1) holds and $\mathsf{Q}_{\ell+1} = \forall$, we can apply the induction hypothesis to the partial computation $\pi_1 \ldots \pi_{\ell-1} \pi'_\ell \pi_{\ell+1}$ (where π'_ℓ is obtained from π_ℓ by removing $\mathsf{last}(\pi_\ell)$). Hence, the result follows.
- $\ell + 1$ is odd: hence $\mathsf{Q}_{\ell+1} = \exists$. Since C_α is existential and $\mathsf{last}(\pi_\ell)$ is not terminal, by hypothesis, $\mathsf{last}(\pi_\ell)$ must be an existential configuration. First, assume that $\mathsf{last}(\pi_\ell)$ leads to acceptance. Hence, there is an uniform partial computation $\pi_{\ell+1}$ of $\mathcal{M}_{\mathcal{A}}$ from $\mathsf{last}(\pi_\ell)$ such that $\mathsf{last}(\pi_{\ell+1})$ leads to acceptance. Let $w_{\ell+1} \in I^{2^n}$ be the word encoding $\pi_{\ell+1}$. Applying the induc-

tion hypothesis to the partial computation $\pi_1 \dots \pi_{\ell-1}\pi'_\ell\pi_{\ell+1}$ (where π'_ℓ is obtained from π_ℓ by removing $\mathsf{last}(\pi_\ell)$), it follows that $\mathsf{Q}_{\ell+2}x_{\ell+2} \in I^{2^n}. \dots \mathsf{Q}_k x_k \in I^{2^n}. \mathcal{M}(w_1, \dots, w_\ell, w_{\ell+1}, x_{\ell+2}, \dots, x_k)$. Thus, since $\mathsf{Q}_{\ell+1} = \exists$, it follows that Condition (1) holds. For the converse direction, assume that Condition (1) holds. Hence, there must be $w_{\ell+1} \in I^{2^n}$ such that (*) $\mathsf{Q}_{\ell+2}x_{\ell+2} \in I^{2^n}. \dots \mathsf{Q}_k x_k \in I^{2^n}. \mathcal{M}(w_1, \dots, w_\ell, w_{\ell+1}, x_{\ell+2}, \dots, x_k)$. By construction on any input of the form $(w_1, \dots, w_\ell, w_{\ell+1}, w'_{\ell+2} \dots, w'_k)$, $\mathcal{M}$ reaches the $(\ell+1)$th step. Note that since $\ell+1$ is odd, Condition 2 in the $(\ell+1)$th step cannot be satisfied (otherwise for all words $w'_{\ell+2}, \dots, w'_k$, $(w_1, \dots, w_\ell,\ w_{\ell+1}, w'_{\ell+2} \dots, w'_k)$ would be rejected by contradicting Condition (*)). Thus, by construction, $w_{\ell+1}$ encodes a uniform partial computation $\pi_{\ell+1}$ of $\mathcal{M}_\mathcal{A}$ from $\mathsf{last}(\pi_\ell)$. By applying the induction hypothesis to the partial computation $\pi_1 \dots \pi_{\ell-1}\pi'_\ell\pi_{\ell+1}$ (where π'_ℓ is obtained from π_ℓ by removing $\mathsf{last}(\pi_\ell)$), it follows that $\mathsf{last}(\pi_{\ell+1})$ leads to acceptance. Since $\mathsf{last}(\pi_\ell)$ is an existential configuration, we obtain that $\mathsf{last}(\pi_\ell)$ leads to acceptance as well, which concludes. □

C.2 Proof of Lemma 4

For each $1 \leq \ell \leq k$, a *ℓ-extended TM block* (resp., *ℓ-TM block*) is an extended TM block (resp., TM block) with component number ℓ. A (k,n)-computation tree code $\langle T, V\rangle$ is said to be *well-initialized* if the (k,n)-configuration with position number 0 encoded by $\langle T, V\rangle$ is initial. In the following, we use the following ML formula $\Phi_{complete}$ characterizing the tree structures such that each path from the root has length $2\mathsf{c}(k,n)+2$.

$$\Phi_{complete} ::= \Box^{2\mathsf{c}(k,n)+3}\mathtt{false} \wedge \bigwedge_{i=0}^{2\mathsf{c}(k,n)+1} \Box^i \Diamond \mathtt{true}$$

Lemma 4. *One can construct in time polynomial in n, k, and the size of the TM $\mathcal{M}$,*

1. *a RML$^\forall$ formula φ^1_{init} over P such that given a tree structure $\langle T, V\rangle$, $\langle T, V\rangle$ satisfies φ^1_{init} if and only if $\langle T, V\rangle$ is a 1-initialized full (k,n)-computation tree code;*
2. *a RML$^\forall$ formula φ^ℓ_{init} over P (for each $2 \leq \ell \leq k$) such that given a* refinement *$\langle T_r, V_r\rangle$ of a $\ell-1$-initialized full (k,n)-computation tree code, $\langle T_r, V_r\rangle$ satisfies φ^ℓ_{init}* if and only if *$\langle T_r, V_r\rangle$ is a ℓ-initialized full (k,n)-computation tree code;*
3. *a RML$^\forall$ formula φ_{comp} over P such that given a* refinement *$\langle T_r, V_r\rangle$ of a k-initialized full (k,n)-computation tree code, $\langle T_r, V_r\rangle$ satisfies φ_{comp}* iff *$\langle T_r, V_r\rangle$ is a (k,n)-computation tree code encoding a (k,n)-computation;*
4. *a ML formula φ_{acc} over P such that given a (k,n)-computation tree code $\langle T, V\rangle$, $\langle T, V\rangle$ satisfies φ_{acc}* iff *the (k,n)-configuration with position number $2^{\mathsf{c}(k,n)-1}$ encoded by $\langle T, V\rangle$ is accepting.*

Proof. Since the proof of Property 1 is very similar to the proof of Property 2 and Property 4 is trivial, we only prove Property 2 and Property 3.

Proof of Property 2. For $2 \leq \ell \leq k$, the RML$^\forall$ formula φ^ℓ_{init} satisfying Property 2 is defined as:

$$\varphi^\ell_{init} := \varphi^\ell_{full} \wedge \varphi^\ell_{<2^n} \wedge \varphi^\ell_{\geq 2^n} \wedge \varphi_{conf}$$

where φ^{ℓ}_{full}, $\varphi^{\ell}_{<2^n}$, $\varphi^{\ell}_{\geq 2^n}$, and φ_{conf} satisfy the following:

- φ^{ℓ}_{full} is a ML formula ensuring that the given refinement of a $\ell-1$-initialized full (k,n)-computation tree code satisfies the fullness requirement in Definition 7.
- $\varphi^{\ell}_{<2^n}$, $\varphi^{\ell}_{\geq 2^n}$, and φ_{conf} are $\mathsf{RML}^{\forall}$ formulas ensuring that the given refinement of a $\ell-1$-initialized full (k,n)-computation tree code satisfies the ℓ-initialization requirement in Definition 7:
  - $\varphi^{\ell}_{<2^n}$ (resp., $\varphi^{\ell}_{\geq 2^n}$) is a $\mathsf{RML}^{\forall}$ formula ensuring that for all extended ℓ-TM blocks *ext_bl* and *ext_bl'* such that $\mathsf{ID}(ext\_bl) = \mathsf{ID}(ext\_bl') = 0$, $\mathsf{CON}(ext\_bl) = bl$, $\mathsf{CON}(ext\_bl') = bl'$, and $\mathsf{ID}(bl) = \mathsf{ID}(bl') < 2^n$ (resp., $\mathsf{ID}(bl) = \mathsf{ID}(bl') \geq 2^n$), the following holds: $\mathsf{CON}(bl) = \mathsf{CON}(bl')$ and $\mathsf{CON}(bl)$ is of the form (u_-, a, u_+) with $a \in I$ (resp., $(u_-, \#, u_+)$).
  - φ_{conf} is a $\mathsf{RML}^{\forall}$ formula ensuring that for all extended ℓ-TM blocks *ext_bl* and *ext_bl'* such that $\mathsf{ID}(ext\_bl) = \mathsf{ID}(ext\_bl') = 0$, $\mathsf{CON}(ext\_bl) = bl$, $\mathsf{CON}(ext\_bl') = bl'$, and $\mathsf{ID}(bl') = \mathsf{ID}(bl) + 1$ (consecutive ℓ-TM blocks), the following holds: $\mathsf{CON}(bl)$ and $\mathsf{CON}(bl')$ are of the form (u_-, u, u_+) and (u, u_+, u_{++}), respectively. Moreover, if $\mathsf{ID}(bl) = 0$ (resp., $\mathsf{ID}(bl') = 2^{\mathsf{c}(k,n)} - 1$), then $\mathsf{CON}(bl)$ (resp., $\mathsf{CON}(bl')$) is of the form $(\bot, u, u_+)$ (resp., $(u_-, u, \bot)$)

$$\varphi^{\ell}_{full} := \Big(\bigwedge_{i=1}^{2\mathsf{c}(k,n)} \Box^{i-1}(\Diamond 0 \wedge \Diamond 1) \Big) \wedge \Big(\Box^{2\mathsf{c}(k,n)} \bigwedge_{i=1}^{k} \Diamond arg_i \Big) \wedge \Big(\Box^{2\mathsf{c}(k,n)+1} \bigwedge_{i=\ell+1}^{k} (arg_i \rightarrow \bigwedge_{t\in\Lambda} \Diamond t) \Big) \wedge$$
$$\Big(\bigwedge_{i=1}^{\mathsf{c}(k,n)} \Box^{i}(1 \rightarrow \Box^{2\mathsf{c}(k,n)-i+1} \bigwedge_{t\in\Lambda} \Diamond t) \Big) \wedge \Phi_{complete}$$

The constructions of formulas $\varphi^{\ell}_{<2^n}$ and $\varphi^{\ell}_{\geq 2^n}$ are similar. So, we just give the definition of $\varphi^{\ell}_{\geq 2^n}$.

$$\varphi^{\ell}_{\geq 2^n} := \forall_r \Big(\underbrace{\Big(\Phi_{complete} \wedge \bigwedge_{i=1}^{\mathsf{c}(k,n)} \Box^{i} 0 \wedge \neg \bigvee_{i=1}^{\mathsf{c}(k,n)} (\Diamond^{\mathsf{c}(k,n)+i} 0 \wedge \Diamond^{\mathsf{c}(k,n)+i} 1) \wedge \bigvee_{i=n+1}^{\mathsf{c}(k,n)} \Diamond^{\mathsf{c}(k,n)+i} 1 \wedge \Box^{2\mathsf{c}(k,n)+1} arg_{\ell} \Big)}_{\text{select } \ell\text{-TM blocks with the same position number } \geq 2^n \text{ associated with extended TM blocks of position number 0}}$$
$$\longrightarrow \bigvee_{(u_-,\#,u_+)\in\Lambda} \Box^{2\mathsf{c}(k,n)+2}(u_-, \#, u_+) \Big)$$

$$\varphi_{conf} := \forall_r \Big(\underbrace{\Big(\Phi_{complete} \wedge \bigwedge_{i=1}^{\mathsf{c}(k,n)} \Box^{i} 0 \wedge \Box^{2\mathsf{c}(k,n)+1} arg_{\ell} \wedge \varphi_{inc} \Big)}_{\text{select two consecutive } \ell\text{-TM blocks associated with extended TM blocks of position number 0}} \longrightarrow \varphi_{check} \Big)$$

where φ_{inc} and φ_{check} are ML formulas defined below. In particular, φ_{inc} allows to "select" only TM blocks having position numbers i and $i+1$ for some $0 \leq i < 2^{\mathsf{c}(k,n)} - 1$, while φ_{check} checks the consistency of the selected consecutive ℓ-TM blocks associated with extended TM blocks of position number 0. Note that for two TM blocks $bl = \{bit_1\}, \ldots, \{bit_{\mathsf{c}(k,n)}\}, \ldots$ and $bl' = \{bit'_1\}, \ldots, \{bit'_{\mathsf{c}(k,n)}\}$, …, $\mathsf{ID}(bl') = \mathsf{ID}(bl) + 1$ iff there is $1 \leq i \leq \mathsf{c}(k,n)$ such that: (1) $bit_i = 0$ and $bit'_i = 1$, (2) for each $1 \leq j \leq i-1$, $bit_j = 1$ and $bit'_j = 0$, and (3) for each $i+1 \leq j \leq \mathsf{c}(k,n)$, $bit_j = bit'_j$.

$$\varphi_{inc} := \Diamond^{\mathsf{c}(k,n)+1} 0 \wedge \Diamond^{\mathsf{c}(k,n)+1} 1 \wedge \bigvee_{i=1}^{\mathsf{c}(k,n)} \Big(\varphi^{i}_{inc} \wedge \neg \bigvee_{j=i+1}^{\mathsf{c}(k,n)} (\Diamond^{\mathsf{c}(k,n)+j} 0 \wedge \Diamond^{\mathsf{c}(k,n)+j} 1) \Big)$$

where $\varphi^i_{inc} := \texttt{true}$ if $i = 1$, and φ^i_{inc} is defined as follows otherwise

$$\Box^{\mathsf{c}(k,n)+1}\big(0 \to (\Box^{i-1}1 \wedge \bigwedge_{j=1}^{i-2} \Box^j 0)\big) \wedge \Box^{\mathsf{c}(k,n)+1}\big(1 \to (\Box^{i-1}0 \wedge \bigwedge_{j=1}^{i-2} \Box^j 1)\big)$$

$$\varphi_{check} := \bigvee_{(u_-,u,u_+),(u,u_+,u_{++})\in\Lambda} \bigvee_{i=1}^{\mathsf{c}(k,n)} \Big((\neg \bigvee_{j=i+1}^{j=\mathsf{c}(k,n)} (\Diamond^{\mathsf{c}(k,n)+j}0 \wedge \Diamond^{\mathsf{c}(k,n)+j}1)) \wedge \Diamond^{\mathsf{c}(k,n)+i}0 \wedge \Diamond^{\mathsf{c}(k,n)+i}1 \wedge$$

$$\underbrace{\Box^{\mathsf{c}(k,n)+i}(0 \to \Box^{\mathsf{c}(k,n)-i+2}(u_-,u,u_+)) \wedge \Box^{\mathsf{c}(k,n)+i}(1 \to \Box^{\mathsf{c}(k,n)-i+2}(u,u_+,u_{++}))\Big)}_{\text{content consistency for consecutive TM blocks associated with the same component of a } (k,n)\text{-configuration}}$$

$$\wedge$$

$$\Big(\Diamond^{\mathsf{c}(k,n)}(\underbrace{\Diamond(0 \wedge \Diamond(0 \wedge \ldots \wedge \Diamond 0)\ldots)}_{\mathsf{c}(k,n) \text{ times}})) \longrightarrow \Diamond^{\mathsf{c}(k,n)}(\underbrace{\Diamond(0 \wedge \Diamond(0 \wedge \ldots \wedge \Diamond(0}_{\mathsf{c}(k,n) \text{ times}} \wedge \bigvee_{(\bot,u,u_+)\in\Lambda} \Diamond^2(\bot,u,u_+))\ldots))))\Big)$$

$$\wedge$$

$$\Big(\Diamond^{\mathsf{c}(k,n)}(\underbrace{\Diamond(1 \wedge \Diamond(1 \wedge \ldots \wedge \Diamond 1)\ldots)}_{\mathsf{c}(k,n) \text{ times}})) \longrightarrow \Diamond^{\mathsf{c}(k,n)}(\underbrace{\Diamond(1 \wedge \Diamond(1 \wedge \ldots \wedge \Diamond(1}_{\mathsf{c}(k,n) \text{ times}} \wedge \bigvee_{(u_-,u,\bot)\in\Lambda} \Diamond^2(u,u,\bot))\ldots))))\Big)$$

Proof of Property 3. Since a (k,n)-computation tree code which is a refinement of a k-initialized full (k,n)-computation tree code is well-initialized, Property 3 directly follows from the following claim.

Claim: One can construct in time polynomial in n, k, and the size of the TM $\mathcal{M}$,

- a $\mathsf{RML}^\forall$ formula φ_{tree_code} over P s.t. given a *refinement* $\langle T_r, V_r\rangle$ of a k-initialized full (k,n)-computation tree code, $\langle T_r, V_r\rangle$ satisfies φ_{tree_code} *iff* $\langle T_r, V_r\rangle$ is a (k,n)-computation tree code;
- a $\mathsf{RML}^\forall$ formula $\varphi_{faithful}$ over P such that given a well-initialized (k,n)-computation tree code $\langle T, V\rangle$, $\langle T, V\rangle$ satisfies $\varphi_{faithful}$ *iff* $\langle T, V\rangle$ encodes a (k,n)-computation.

Proof of the claim.

Construction of the $\mathsf{RML}^\forall$ formula φ_{tree_code}:

$$\varphi_{tree\_code} := \Phi_{complete} \wedge \bigwedge_{i=1}^{2\mathsf{c}(k,n)} \big(\Box^{i-1}(\Diamond 0 \wedge \Diamond 1)\big) \wedge \Box^{2\mathsf{c}(k,n)}(\bigwedge_{\ell=1}^{k} \Diamond arg_\ell) \wedge \varphi_{unique} \wedge \varphi_{control} \wedge \varphi'_{conf}$$

where φ_{unique}, $\varphi_{control}$, and φ'_{conf} ensure Property 2 in Definition 6. In particular, the following holds:

- φ_{unique} is a $\mathsf{RML}^\forall$ formula ensuring that for all extended TM blocks *ext_bl* and *ext_bl'* such that $\mathsf{ID}(ext\_bl) = \mathsf{ID}(ext\_bl')$, $\mathsf{CON}(ext\_bl) = bl$, $\mathsf{CON}(ext\_bl') = bl'$, and bl and bl' have the same component and position number, the following holds: $\mathsf{CON}(bl) = \mathsf{CON}(bl')$ (i.e., for each position number i, there is a unique (k,n)-configuration with position number i).

– $\varphi_{control}$ is a ML formula ensuring that for each position number i, there is a unique extended TM block *ext_bl* such that $\mathsf{ID}(ext\_bl) = i$ and $\mathsf{CON}(\mathsf{CON}(ext\_bl))$ is of the form $(u_-,(q,a),u_+)$.
– φ'_{conf} is a $\mathsf{RML}^{\forall}$ formula ensuring that for all extended TM blocks *ext_bl* and *ext_bl'* such that $\mathsf{ID}(ext\_bl) = \mathsf{ID}(ext\_bl')$, $\mathsf{CON}(ext\_bl) = bl$, $\mathsf{CON}(ext\_bl') = bl'$, bl and bl' have the same component number, and $\mathsf{ID}(bl') = \mathsf{ID}(bl) + 1$ (consecutive TM blocks), the following holds: $\mathsf{CON}(bl)$ and $\mathsf{CON}(bl')$ are of the form (u_-,u,u_+) and (u,u_+,u_{++}), respectively. Moreover, if $\mathsf{ID}(bl) = 0$ (resp., $\mathsf{ID}(bl') = 2^{\mathsf{c}(k,n)} - 1$), then $\mathsf{CON}(bl)$ (resp., $\mathsf{CON}(bl')$) is of the form $(\bot,u,u_+)$ (resp., $(u_-,u,\bot)$).

$$\varphi_{unique} := \forall_r\Big(\underbrace{\big(\Phi_{complete} \wedge \neg\big(\bigvee_{i=1}^{2\mathsf{c}(k,n)} \Diamond^i 0 \wedge \Diamond^i 1\big) \wedge \bigvee_{1\le\ell\le k} \Box^{2\mathsf{c}(k,n)+1} arg_\ell\big)}_{\text{select for some } \ell,\, i, \text{ and } j, \text{ extended } \ell\text{-TM blocks } ext\_bl \text{ with } \mathsf{ID}(ext\_bl)=i \text{ and } \mathsf{ID}(\mathsf{CON}(ext\_bl))=j} \longrightarrow \bigvee_{t\in\Lambda} \Box^{2\mathsf{c}(k,n)+2} t\Big)$$

$$\varphi_{control} := \Box^{\mathsf{c}(k,n)}\Big[\big(\bigvee_{(u_-,(q,a),u_+)\in\Lambda} \Diamond^{\mathsf{c}(k,n)+2}(u_-,(q,a),u_+)\big) \wedge \neg \bigvee_{(u_-,(q,a),u_+),(u'_-,(q',a'),u'_+)\in\Lambda}$$
$$\Big(\big\{\bigvee_{1\le\ell\ne h\le k} \Diamond^{\mathsf{c}(k,n)+1}(arg_\ell \wedge \Diamond(u_-,(q,a),u_+)) \wedge \Diamond^{\mathsf{c}(k,n)+1}(arg_h \wedge \Diamond(u'_-,(q',a'),u'_+))\big\} \vee$$
$$\big\{\bigvee_{i=1}^{\mathsf{c}(k,n)} \Diamond^i(0 \wedge \Diamond^{\mathsf{c}(k,n)-i+2}(u_-,(q,a),u_+)) \wedge \Diamond^i(1 \wedge \Diamond^{\mathsf{c}(k,n)-i+2}(u'_-,(q',a'),u'_+))\big\}\Big)\Big]$$

$$\varphi'_{conf} := \forall_r\Big(\underbrace{\big(\Phi_{complete} \wedge \neg\big(\bigvee_{i=1}^{\mathsf{c}(k,n)} \Diamond^i 0 \wedge \Diamond^i 1\big) \wedge \bigvee_{1\le\ell\le k} \Box^{2\mathsf{c}(k,n)+1} arg_\ell \wedge \varphi_{inc}\big)}_{\text{select two consecutive TM blocks of some component of some } (k,n)\text{-configuration}} \longrightarrow \varphi_{check}\Big)$$

where φ_{inc} and φ_{check} are the ML formulas defined at the end of the proof of Property 1.

Construction of the $\mathsf{RML}^{\forall}$ formula $\varphi_{faithful}$: First, we construct two ML formulas φ_{sc} and φ_{sb} that satisfy the following:

– given a refinement $\langle T_r,V_r\rangle$ of a (k,n)-computation tree code $\langle T,V\rangle$, $\langle T_r,V_r\rangle$ satisfies φ_{sc} iff $\langle T_r,V_r\rangle$ "selects" two consecutive (k,n)-configurations encoded by $\langle T,V\rangle$ (i.e., two (k,n) configurations having position numbers i and $i+1$ for some $0 \le i < 2^{\mathsf{c}(k,n)} - 1$);
– given a refinement $\langle T_r,V_r\rangle$ of a well-initialized (k,n)-computation tree code which "selects" two consecutive (k,n)-configurations $\overrightarrow{C}$ and $\overrightarrow{C}'$, it holds that: for each refinement $\langle T'_r,V'_r\rangle$ of $\langle T_r,V_r\rangle$, $\langle T'_r,V'_r\rangle$ satisfies φ_{sb} iff $\langle T'_r,V'_r\rangle$ "selects" two TM blocks bl and bl' which have the same position and component number, and are associated with $\overrightarrow{C}$ and $\overrightarrow{C}'$, respectively.

$$\varphi_{sc} := \Phi_{complete} \wedge \big(\bigwedge_{i=\mathsf{c}(k,n)+1}^{2\mathsf{c}(k,n)} \Box^{i-1}(\Diamond 0 \wedge \Diamond 1)\big) \wedge \big(\Box^{2\mathsf{c}(k,n)} \bigwedge_{1\le\ell\le k} \Diamond arg_\ell\big) \wedge \Diamond 0 \wedge \Diamond 1 \wedge$$
$$\bigvee_{i=1}^{\mathsf{c}(k,n)} \big(\psi^i_{inc} \wedge \neg \bigvee_{j=i+1}^{\mathsf{c}(k,n)} (\Diamond^j 0 \wedge \Diamond^j 1)\big)$$

where $\psi^i_{inc} ::= \mathtt{true}$ if $i = 1$, and ψ^i_{inc} is defined as follows otherwise

$$\Box\big(0 \to (\Box^{i-1}1 \wedge \bigwedge_{j=1}^{i-2} \Box^j 0)\big) \wedge \Box\big(1 \to (\Box^{i-1}0 \wedge \bigwedge_{j=1}^{i-2} \Box^j 1)\big)$$

$$\varphi_{sb} := \Phi_{complete} \wedge \bigvee_{j=1}^{\mathsf{c}(k,n)} (\Diamond^j 0 \wedge \Diamond^j 1) \wedge \neg\big(\bigvee_{i=1}^{\mathsf{c}(k,n)} \Diamond^{\mathsf{c}(k,n)+i} 0 \wedge \Diamond^{\mathsf{c}(k,n)+i} 1\big) \wedge \bigvee_{1 \leq \ell \leq k} \Box^{2\mathsf{c}(k,n)+1} arg_\ell$$

Now, we describe the construction of the $\mathsf{RML}^\forall$ formula $\varphi_{faithful}$:

$$\varphi_{faithful} ::= \forall_r\big(\varphi_{sc} \longrightarrow \bigvee_{(q,a) \in (Q \setminus \{q_{acc}, q_{rej}\}) \times A} (\psi_{q,a}^{check} \wedge \psi_{q,a}^{faithful})\big)$$

where for a refinement of a well-initialized (k,n)-computation tree code which selects two (k,n)-configurations $\overrightarrow{C}$ and $\overrightarrow{C}'$ with position numbers i and $i+1$, respectively, we have that: (1) $\psi_{q,a}^{check}$ is a ML formula that checks that $(q,a) \in (Q \setminus \{q_{acc}, q_{rej}\}) \times A$ is the pair state/scanned cell content associated with the (k,n)-configuration $\overrightarrow{C}$, and (2) $\psi_{q,a}^{faithful}$ is a $\mathsf{RML}^\forall$ formula that uses the ML formula φ_{sb} and checks that $\overrightarrow{C}'$ is the TM successor of $\overrightarrow{C}$.

$$\psi_{q,a}^{check} := \bigvee_{(u_-,(q,a),u_+) \in \Lambda} \bigvee_{i=1}^{\mathsf{c}(k,n)} \Big(\Diamond^i 1 \wedge \Diamond^i (0 \wedge \Diamond^{2\mathsf{c}(k,n)-i+2}(u_-,(q,a),u_+)) \wedge \neg \bigvee_{j=i+1}^{\mathsf{c}(k,n)} (\Diamond^j 0 \wedge \Diamond^j 1)\Big)$$

It remains to construct the $\mathsf{RML}^\forall$ formula $\psi_{q,a}^{faithful}$. We distinguish two cases.

Case $\delta(q,a)$ is an ordinary move: let $\overrightarrow{C} = (C_1(0) \ldots C_1(2^{\mathsf{c}(k,n)-1}), \ldots, C_k(0) \ldots C_k(2^{\mathsf{c}(k,n)-1}))$ be a (k,n)-configuration whose pair state/scanned cell content is (q,a). In this case, for all $1 \leq \ell \leq k$ and $0 \leq j \leq 2^{\mathsf{c}(k,n)} - 1$, the 'value' $u_{\ell,j}$ of the j-th symbol of the ℓth component of the $\overrightarrow{C}$-successor is completely determined by the values $C_\ell(j-1)$, $C_\ell(j)$ and $C_\ell(j+1)$ (taking $C_\ell(j+1)$ for $j = 2^{\mathsf{c}(k,n)} - 1$ and $C_\ell(j-1)$ for $j = 0$ to be $\perp$). We denote by $next(C_\ell(j-1), C_\ell(j), C_\ell(j+1))$ our expectation for $u_{\ell,j}$ (this function can be trivially obtained from the transition function δ of $\mathcal{M}$). Thus, in this case, we have to check that given a refinement $\langle T_r, V_r \rangle$ of a well-initialized (k,n)-computation tree code which "selects" two (k,n)-configurations $\overrightarrow{C}$ and $\overrightarrow{C}'$ with position numbers i and $i+1$ (for some i) respectively (we say that $\overrightarrow{C}$ is the first configuration and $\overrightarrow{C}'$ is the second configuration), and such that (q,a) is the pair state/scanned cell content associated with $\overrightarrow{C}$, the following holds: for each refinement of $\langle T_r, V_r \rangle$ which selects two TM blocks bl and bl' such that bl and bl' have the same position and component number, and are associated with $\overrightarrow{C}$ and $\overrightarrow{C}'$, respectively, $\mathsf{CON}(bl')$ is of the form $(u_-, next(\mathsf{CON}(bl)), u_+)$. Hence, $\psi_{q,a}^{faithful}$ is defined as follows:

$$\psi_{q,a}^{faithful} ::= \forall_r\Big(\varphi_{sb} \longrightarrow \bigvee_{t,(u_-,next(t),u_+) \in \Lambda} \bigvee_{i=1}^{\mathsf{c}(k,n)} \big([\neg \bigvee_{j=i+1}^{\mathsf{c}(k,n)} (\Diamond^j 0 \wedge \Diamond^j 1)] \wedge \underbrace{[\Diamond^i (0 \wedge \Diamond^{2\mathsf{c}(k,n)-i+2} t)]}_{\text{select the TM block of the first } (k,n)\text{-configuration}} \wedge \underbrace{[\Diamond^i (1 \wedge \Diamond^{2\mathsf{c}(k,n)-i+2}(u_-, next(t), u_+))]}_{\text{select the TM block of the second } (k,n)\text{-configuration}}\big)\Big)$$

Case $\delta(q,a)$ is a jump move: for $s \in A \cup (Q \times A)$, let $A(s) := s$ if $s \in A$ and $A(s)$ be the A-component of s otherwise. Let $\overrightarrow{C} = (C_1(0)\ldots C_1(2^{\mathsf{c}(k,n)-1}),\ldots,C_k(0)\ldots C_k(2^{\mathsf{c}(k,n)-1}))$ be a (k,n)-configuration whose pair state/scanned cell content is (q,a). In this case, for all $1 \leq \ell \leq k$ and $0 \leq j \leq 2^{\mathsf{c}(k,n)}-1$, the 'value' $u_{\ell,j}$ of the j-th symbol of the ℓth component of the $\overrightarrow{C}$-successor satisfies the following: $u_{\ell,j} = (q, A(C_\ell(j)))$ if $\ell = \delta(q,a)$ and $j = 0$, and $u_{\ell,j} = A(C_\ell(j))$ otherwise. Let $\langle T_r, V_r\rangle$ be a refinement of a well-initialized (k,n)-computation tree code which "selects" two (k,n)-configurations $\overrightarrow{C}$ and $\overrightarrow{C}'$ with position numbers i and $i+1$ (for some i) respectively, and such that (q,a) is the pair state/scanned cell content associated with $\overrightarrow{C}$. We have to check that for each refinement of $\langle T_r, V_r\rangle$ which selects two TM blocks bl and bl' such that bl and bl' have the same position and component number, and are associated with $\overrightarrow{C}$ and $\overrightarrow{C}'$, respectively, the following holds:

- if bl and bl' have position number 0 and component number $\delta(q,a)$, then $\mathsf{CON}(bl) = (u_-, u, u_+)$ and $\mathsf{CON}(bl') = (u'_-, (q, A(u)), u'_+)$;
- otherwise, $\mathsf{CON}(bl) = (u_-, u, u_+)$ and $\mathsf{CON}(bl') = (u'_-, A(u), u'_+)$.

Hence, $\psi^{\mathit{faithful}}_{q,a}$ is defined as follows:

$$\psi^{\mathit{faithful}}_{q,a} ::= \forall_r \Big(\varphi_{sb} \longrightarrow \big(\underbrace{[(\Diamond^{2\mathsf{c}(k,n)+1} \mathit{arg}_{\delta(q,a)} \wedge \bigwedge_{j=1}^{\mathsf{c}(k,n)} \Box^{\mathsf{c}(k,n)+j} 0) \longrightarrow \psi_q]}_{\text{the two TM blocks have position number 0 and component number } \delta(q,a)} \quad \wedge$$

$$\underbrace{[\neg(\Diamond^{2\mathsf{c}(k,n)+1} \mathit{arg}_{\delta(q,a)} \wedge \bigwedge_{j=1}^{\mathsf{c}(k,n)} \Box^{\mathsf{c}(k,n)+j} 0) \rightarrow \psi]}_{\text{for the two selected TM blocks, either the position number is not 0 or the component number is not } \delta(q,a)} \big) \Big)$$

where ψ_q and ψ are ML-formulas. Below, we define ψ_q (the construction of ψ is very similar).

$$\psi_q ::= \bigvee_{(u_-,u,u_+),(u'_-,(q,A(u)),u'_+) \in \Lambda} \bigvee_{i=1}^{\mathsf{c}(k,n)} \Big([\neg \bigvee_{j=i+1}^{\mathsf{c}(k,n)} (\Diamond^j 0 \wedge \Diamond^j 1)] \wedge$$

$$\underbrace{[\Diamond^i (0 \wedge \Diamond^{2\mathsf{c}(k,n)-i+2} (u_-, u, u_+))]}_{\text{select the TM block of the first } (k,n)\text{-configuration}} \wedge \underbrace{[\Diamond^i (1 \wedge \Diamond^{2\mathsf{c}(k,n)-i+2} (u'_-, (q, A(u)), u'_+))]}_{\text{select the TM block of the second } (k,n)\text{-configuration}} \Big)$$

This concludes the proof of the claim. □

□

C.3 Detailed proof of Theorem 4

Theorem 4. *One can construct a* RML^{k+1} *formula* φ *in time polynomial in n, k, and the size of the TM* $\mathcal{M}$ *such that* φ *is satisfiable* if and only if

$$\mathsf{Q}_1 x_1 \in I^{2^n}. \mathsf{Q}_2 x_2 \in I^{2^n}. \ldots \mathsf{Q}_k x_k \in I^{2^n}. \mathcal{M}(x_1,\ldots,x_k)$$

where $\mathsf{Q}_\ell = \exists$ *if* ℓ *is odd, and* $\mathsf{Q}_\ell = \forall$ *otherwise (for all* $1 \leq \ell \leq k$*).*

Proof. Let $\varphi^1_{\mathit{init}},\ldots,\varphi^k_{\mathit{init}}$, and φ_{comp} be the $\mathsf{RML}^\forall$ formulas satisfying Properties 1–3 of Lemma 4, and φ_{acc} be the ML formula satisfying Property 4 of Lemma 4. Then, the RML^{k+1} formula φ is

defined as follows, where $\widetilde{\mathsf{Q}}_\ell = \exists_r$ and $\mathtt{op}_\ell = \wedge$ if ℓ is odd, and $\widetilde{\mathsf{Q}}_\ell = \forall_r$ and $\mathtt{op}_\ell = \rightarrow$ otherwise (for all $2 \leq \ell \leq k$):

$$\varphi := \varphi^1_{init} \wedge \widetilde{\mathsf{Q}}_2(\varphi^2_{init}\ \mathtt{op}_2\ \widetilde{\mathsf{Q}}_3(\varphi^3_{init}\ \mathtt{op}_3\ \ldots\ \mathtt{op}_{k-1}\ \widetilde{\mathsf{Q}}_k(\varphi^k_{init}\ \mathtt{op}_k\ \widetilde{\mathsf{Q}}_k(\varphi_{comp}\ \mathtt{op}_k\ \varphi_{acc}))\ldots))$$

By construction and Lemma 4, it easily follows that φ is RML^{k+1} formula which can be constructed in time polynomial in n, k, and the size of the TM $\mathcal{M}$. Let $\varphi_1 := \varphi$, $\varphi_{k+1} := \widetilde{\mathsf{Q}}_k(\varphi_{comp}\ \mathtt{op}_k\ \varphi_{acc})$, and for each $2 \leq \ell \leq k$,

$$\varphi_\ell := \widetilde{\mathsf{Q}}_\ell(\varphi^\ell_{init}\ \mathtt{op}_\ell\ \widetilde{\mathsf{Q}}_{\ell+1}(\varphi^{\ell+1}_{init}\ \mathtt{op}_{\ell+1}\ \ldots\ \mathtt{op}_{k-1}\ \widetilde{\mathsf{Q}}_k(\varphi^k_{init}\ \mathtt{op}_k\ \widetilde{\mathsf{Q}}_k(\varphi_{comp}\ \mathtt{op}_k\ \varphi_{acc}))\ldots))$$

Correctness of the construction directly follows from the following claim, where a 0-initialized full (k,n)-computation tree code is an arbitrary tree structure.

Claim: let $0 \leq \ell \leq k$, $w_1, \ldots, w_\ell \in I^{2^n}$, and $\langle T, V\rangle$ be a ℓ-initialized full (k,n)-computation tree code such that $\langle T,V\rangle(w_1,\ldots,w_\ell)$ holds. Then, $\langle T,V\rangle$ satisfies $\varphi_{\ell+1}$ if and only if

$$\mathsf{Q}_{\ell+1}x_{\ell+1} \in I^{2^n}. \ldots \mathsf{Q}_k x_k \in I^{2^n}. \mathcal{M}(w_1,\ldots,w_\ell,x_{\ell+1},\ldots,x_k) \tag{2}$$

Proof of the claim: the proof is by induction on $k-\ell$.

Base Step: $\ell = k$. We need to show that $\langle T,V\rangle$ satisfies $\varphi_{k+1} := \widetilde{\mathsf{Q}}_k(\varphi_{comp}\ \mathtt{op}_k\ \varphi_{acc})$ iff $\mathcal{M}(w_1,\ldots,w_k)$.

We assume that k is even (the other case being similar). Then, by construction, $\varphi_{k+1} = \forall_r(\varphi_{comp} \rightarrow \varphi_{acc})$. First, assume that $\langle T,V\rangle$ satisfies φ_{k+1}. Since $\langle T,V\rangle$ is a k-initialized full (k,n)-computation tree code such that $\langle T,V\rangle(w_1,\ldots,w_k)$ holds, by Proposition 3(2) there is a refinement $\langle T_r,V_r\rangle$ of $\langle T,V\rangle$ which is a (k,n)-computation tree code encoding the (k,n)-computation from the (k,n)-input $(w_1,\ldots,w_k)$. By Lemma 4(3), it follows that $\langle T_r,V_r\rangle$ satisfies φ_{comp}. Since $\langle T,V\rangle$ satisfies φ_{k+1}, we deduce that $\langle T_r,V_r\rangle \models \varphi_{acc}$. Thus, by Lemma 4(3), we obtain that the (k,n)-computation from the (k,n)-input $(w_1,\ldots,w_k)$ is accepting, i.e. $\mathcal{M}(w_1,\ldots,w_k)$.
For the converse implication assume that $\mathcal{M}(w_1,\ldots,w_k)$. Let $\langle T_r,V_r\rangle$ be any refinement of $\langle T,V\rangle$ satisfying φ_{comp}. We need to show that $\langle T_r,V_r\rangle$ satisfies φ_{acc}. By Lemma 4(3), it follows that $\langle T_r,V_r\rangle$ is a (k,n)-computation tree code encoding a (k,n)-computation. Moreover, since $\langle T_r,V_r\rangle$ is a refinement of $\langle T,V\rangle$ and $\langle T,V\rangle(w_1,\ldots,w_k)$ holds, by Proposition 3(2), $\langle T_r,V_r\rangle$ encodes the (accepting) (k,n)-computation from the (k,n)-input $(w_1,\ldots,w_k)$. Thus, by Lemma 4(4), the result follows.

Induction Step: let $\ell < k$. We assume that $\ell > 0$ and $\ell+1$ is even (the other cases being similar). Then, $\varphi_{\ell+1} = \forall_r(\varphi^{\ell+1}_{init} \rightarrow \varphi_{\ell+2})$ and $\mathsf{Q}_{\ell+1} = \forall$. First, assume that $\langle T,V\rangle$ satisfies $\varphi_{\ell+1}$. Let $w_{\ell+1} \in I^{2^n}$. Since $\langle T,V\rangle$ is a ℓ-initialized full (k,n)-computation tree code such that $\langle T,V\rangle(w_1,\ldots,w_\ell)$ holds, by Proposition 3(1) there must be a refinement $\langle T_r,V_r\rangle$ of $\langle T,V\rangle$ such that $\langle T_r,V_r\rangle$ is a $\ell+1$-initialized full (k,n)-computation tree code and $\langle T_r,V_r\rangle(w_1,\ldots,w_{\ell+1})$ holds. By Lemma 4(2), $\langle T_r,V_r\rangle \models \varphi^{\ell+1}_{init}$. Since $\langle T,V\rangle$ satisfies $\varphi_{\ell+1}$, we deduce that $\langle T_r,V_r\rangle \models \varphi_{\ell+2}$. Thus, by the induction hypothesis it follows that $\mathsf{Q}_{\ell+2}x_{\ell+2} \in I^{2^n}. \ldots \mathsf{Q}_k x_k \in I^{2^n}. \mathcal{M}(w_1,\ldots,w_{\ell+1},x_{\ell+2},\ldots,x_k)$. Since $\mathsf{Q}_{\ell+1} = \forall$ and $w_{\ell+1}$ is arbitrary, we obtain that Condition (2) in the claim holds. For the converse implication, assume that Condition (2) in the claim holds. Let $\langle T_r,V_r\rangle$ be a refinement of $\langle T,V\rangle$ satisfying $\varphi^{\ell+1}_{init}$. We need to show that $\langle T_r,V_r\rangle \models \varphi_{\ell+2}$. By Lemma 4(2), it holds that $\langle T_r,V_r\rangle$ is a $\ell+1$-initialized full (k,n)-computation tree code. Since $\langle T_r,V_r\rangle$ is a refinement of $\langle T,V\rangle$ and $\langle T,V\rangle(w_1,\ldots,w_\ell)$ holds, by Proposition 3(1) there must be $w_{\ell+1} \in I^{2^n}$ such that $\langle T_r,V_r\rangle(w_1,\ldots,w_{\ell+1})$ holds. Since $\mathsf{Q}_{\ell+1} = \forall$,

by hypothesis, it holds that $\mathsf{Q}_{\ell+2}x_{\ell+2} \in I^{2^n}. \ldots \mathsf{Q}_k x_k \in I^{2^n}.\ \mathcal{M}(w_1,\ldots,w_{\ell+1},x_{\ell+2},\ldots,x_k)$. Thus, by the induction hypothesis, $\langle T_r, V_r\rangle \models \varphi_{\ell+2}$ holds, and we are done. □

□

C.4 Proof of Theorem 5

Theorem 5. *Let $k = 1$. Then, one can construct a $\mathsf{RML}^{\forall}$ formula $\varphi^{\forall}$ in time polynomial in n and the size of the TM $\mathcal{M}$ such that $\varphi^{\forall}$ is satisfiable* if and only if $\exists x \in I^{2^n}.\, \mathcal{M}(x)$.

Proof. The $\mathsf{RML}^{\forall}$ formula $\varphi^{\forall}$ is given by $\psi_{tree\_code} \wedge \varphi_{faithful} \wedge \varphi_{acc}$, where $\varphi_{faithful}$ is the $\mathsf{RML}^{\forall}$ formula used in the proof of Lemma 4(3), φ_{acc} is the ML formula of Lemma 4(4), and ψ_{tree_code} is a $\mathsf{RML}^{\forall}$ formula over P which is satisfied by a tree structure $\langle T, V\rangle$ iff $\langle T, V\rangle$ is a $(1,n)$-computation tree code such that the $(1,n)$-configuration with position number 0 encoded by $\langle T, V\rangle$ is initial. The construction of the $\mathsf{RML}^{\forall}$ formula ψ_{tree_code} is similar to the construction of the $\mathsf{RML}^{\forall}$ formula φ_{tree_code} in the proof of Lemma 4(3). Thus, we omit the details here. □