The Fixed-Parameter Tractability of Model Checking Concurrent Systems *

Stefan Göller

LIAFA/CNRS, Paris 7, France
Universität Bremen, Fachbereich Mathematik und Informatik, Germany
goeller@informatik.uni-bremen.de

Abstract

We study the fixed-parameter complexity of model checking temporal logics on concurrent systems that are modeled as the product of finite systems and where the size of the formula is the parameter. We distinguish between asynchronous product and synchronous product. Sometimes it is possible to show that there is an algorithm for this with running time $(\sum_i |\mathcal{T}_i|)^{O(1)} \cdot f(|\varphi|)$, where the $\mathcal{T}_i$ are the component systems and φ is the formula and f is computable function, thus model checking is fixed-parameter tractable when the size of the formula is the parameter.

In this paper we concern ourselves with the question, provided fixed-parameter tractability is known, whether it holds for an elementary function f. Negative answers to this question are provided for modal logic and EF logic: Depending on the mode of synchronization we show the non-existence of such an elementary function f under different assumptions from (parameterized) complexity theory.

1998 ACM Subject Classification F.4.1 Mathematical Logic, F.2.0 Analysis of Algorithms and Problem Complexity: General

Keywords and phrases Model Checking, Concurrent Systems, Parameterized Complexity

Digital Object Identifier 10.4230/LIPIcs.CSL.2013.332

1 Introduction

Model checking is one of the most successful approaches in formal verification; it asks to verify if a given specification is satisfied in a given system [5, 2]. The computational complexity of model checking finite systems is very well understood (beginning with the pioneer work [15]), but model checking the modal μ-calculus is an exception: The best-known upper bound in $\mathsf{UP} \cap \mathsf{co\text{-}UP}$ and the lower bound is P. In general it turns out that typically the source of hardness of model checking lies in the size of the formula and not in the size of the input system. Indeed, the complexity of model checking the modal μ-calculus lies in P for every fixed formula. Another such famous example is model checking of linear temporal logic LTL that is generally PSPACE-complete but which can be solved in time $|\mathcal{T}|^{O(1)} \cdot 2^{|\varphi|}$, where $\mathcal{T}$ is the (transition) system and φ is the formula [12]. Measuring the complexity only in the size of the system and not in the size of the specification is indeed justified since typically the specification is small and the system is big.

Yet, model checking tools have to deal with a combinatorial blowup of the state space of the input system, commonly known as the *state explosion problem*, that can be seen as one of the biggest challenges in real-world problems. Different sources of explosion arise, for

* The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 259454.

Computer Science Logic 2013 (CSL'13).
Editor: Simona Ronchi Della Rocca; pp. 332–347

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

instance the number of program variables or clocks, the number of concurrently running components, or the number of different subroutines, just to mention few of them. Technically speaking, this can be seen that the input consists of systems $\mathcal{T}_1, \ldots, \mathcal{T}_d$ (the *components*), but the actual system of interest is the product of these systems; independent on the underlying synchronization mechanism, we refer to such systems as *concurrent systems* in the following.

Parameterized complexity theory allows for a fine-grained complexity investigation of problems where certain sizes of the input are declared as a parameter. A central class in this theory is the class FPT consisting of all problems that can be solved in time $n^{O(1)} \cdot f(k)$ for some computable function f, where n is the input size and k is the parameter. For instance, the above-mentioned model checking problem of LTL is in FPT when the size of the formula is the parameter. Although their exact definitions are not relevant here, we list the parameterized complexity classes that are subject of this paper

$$\mathsf{FPT} \subseteq \mathsf{AW}[*] \subseteq \mathsf{AW}[\mathsf{SAT}],$$

where none of the inclusions is known but believed to be strict. The class AW[$*$] can be seen as fixed parameter tractability plus parameter bounded alternating nondeterminism [4] and AW[SAT] is the set of all problems that are (FPT)-reducible to a variant of QBF, where quantification is restricted to a partition of the variables and in which the number of variables that are assigned to true in each quantifier block plus the number of partitions of the variables is the parameter. We refer to [8] for more details on parameterized complexity theory.

The parameterized complexity of model checking synchronous concurrent systems has intensively been studied by Demri, Laroussinie and Schnoebelen [7]. Among their results it is shown that already very simple questions like reachability or model checking modal logic are not fixed-parameter tractable when the number of components and the size of the formula is the parameter, respectively. More precisely in [7] it is shown that (i) already when the number of components d is the parameter it is AWT[SAT]-hard to decide reachability of a synchronous product of d transition systems, already when transitions can only synchronize over a binary alphabet and (ii) when the size of the formula is the parameter, model checking the synchronous product of systems with respect to modal logic is AW[$*$]-complete. Since the above-mentioned parameterized complexity classes are believed to be strict, we have that (i) and (ii) show that already these two basic model checking problems are *not* fixed-parameter tractable unless FPT coincides with them.

It is important to mention that in [7] the technical reason for AW[$*$]-hardness for model checking modal logic on the synchronous product lies in the fact the transitions can synchronize over a transition alphabet that is independent of the size of the formula. Thus, the question arises whether fixed-parameter tractability can be gained when the transition alphabet is bounded by the size of the formula (which is natural if one considers multi-modal logic, for instance). Too, the question arises whether fixed-parameter tractability can be gained when we restrict the synchronization mechanism to be the *asynchronous* product.

Unfortunately still, in both cases under the assumption $\mathsf{FPT} \neq \mathsf{AWT}[\mathsf{SAT}]$, already for powerful branching-time logics like CTL, the answer to both questions is negative since reachability of synchronous product [7] can easily be encoded. However, the compositional method à la Feferman and Vaught, which has recently been developed for the fragments modal logic and EF logic by Rabinovich [14], gives positive answers to the above questions: In certain cases, one can show that for model checking the product of given systems $\mathcal{T}_1, \ldots, \mathcal{T}_d$ against a given formula φ there is an algorithm with running time $(\sum_i |\mathcal{T}_i|)^{O(1)} \cdot f(|\varphi|)$ for a primitive recursive function f, thus being fixed-parameter tractable. Unfortunately, the

compositional method yields an algorithm whose running time is bounded by a nonelementary function f; moreover this is not avoidable as recently proven [11, 10].

In this paper we concern ourselves with the question, provided fixed-parameter tractability for model checking concurrent systems is known, whether one can hope for *any* algorithm witnessing fixed-parameter tractability with an elementarily growing function f. The main results of this paper answer this question negatively under different assumptions from (parameterized) complexity theory depending on the logic and the synchronization mode under consideration.

Our contribution. We revisit briefly that the compositional method allows to model check given systems $\mathcal{T}_1, \ldots, \mathcal{T}_d$ and a formula φ in time $(\sum_i |\mathcal{T}_i|)^{O(1)} \cdot f(|\varphi|)$ in case either (i) φ is a formula of modal logic and the asynchronous product of the $\mathcal{T}_1, \ldots, \mathcal{T}_d$ is considered, (ii) φ is a formula of EF logic and the asynchronous product of the $\mathcal{T}_1, \ldots, \mathcal{T}_d$ is considered, or (iii) φ is a formula of modal logic and the synchronous product of the $\mathcal{T}_1, \ldots, \mathcal{T}_d$ is considered. For (i) we show that there is no algorithm for this that runs in time $(\sum_i |\mathcal{T}_i|)^{O(1)} \cdot f(|\varphi|)$ for any elementary function f unless $\mathsf{FPT} = \mathsf{AW}[*]$. For (ii) and (iii) we prove that there is no algorithm for this that runs in time $(\sum_i |\mathcal{T}_i|)^{O(1)} \cdot f(|\varphi|)$ for any elementary function f unless $\mathsf{P} = \mathsf{NP}$. We remark that the assumption $\mathsf{FPT} \neq \mathsf{AW}[*]$ is a stronger assumption than $\mathsf{P} \neq \mathsf{NP}$. The overall picture of the fixed-parameter tractability of model checking modal logic and EF logic on concurrent systems is summarized in Table 1.

Related work. The parameterized complexity of various problems in formal verification has been investigated in [7, 13] and we refer to the reference therein and to [8] for more information on parameterized complexity. The parameterized complexity of model checking first-order logic and monadic second-order logic over words has been studied in [9] and in fact we give a reduction from model checking first-order logic over words to model checking modal logic on the asynchronous product of systems in this paper. The parameterized complexity of satisfiability of modal logic under various parameters of the input formulas has been investigated in [1].

Organization of this paper. Preliminaries, the compositional method and upper bounds are content of Section 2. Section 3 provides technical tools that allow us to compare trees that encode large numbers with small formulas. In Section 4 we discuss the proof strategies of our main results. In Section 5 we show that fixed-parameter tractability is not possible with an elementary running time in the size of the formula for model checking modal logic on the asynchronous product unless $\mathsf{FPT} = \mathsf{AW}[*]$. We show in Section 6 that for model checking modal logic on the synchronous product and for model checking EF on the asynchronous product fixed-parameter tractability is not possible with an elementary running time in the size of the formula unless $\mathsf{P} = \mathsf{NP}$. We conclude in Section 7. Missing proofs due to space restrictions can be found in the appendix.

2 Preliminaries

The exact definitions of the parameterized complexity classes that appear in this paper are not important, we refer the reader to [8] for more details. For two integers i and j, we define the interval $[i, j] \stackrel{\text{def}}{=} \{i, i+1, \ldots, j\}$. By $\mathbb{N} \stackrel{\text{def}}{=} \{0, 1, 2, \ldots, \}$ we denote the set of *non-negative integers*. The *tower function* $\mathsf{Tower} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is defined as $\mathsf{Tower}(0, n) \stackrel{\text{def}}{=} n$ and $\mathsf{Tower}(\ell + 1, n) \stackrel{\text{def}}{=} 2^{\mathsf{Tower}(\ell, n)}$ for each $\ell \in \mathbb{N}$ and each $n \in \mathbb{N}$. We also introduce the tower function in one parameter as $\mathsf{Tower}(\ell) \stackrel{\text{def}}{=} \mathsf{Tower}(\ell, 2)$ for each $\ell \in \mathbb{N}$. Define the

Table 1 The fixed-parameter tractability of model checking the product of finite systems $\mathcal{T}_1, \ldots, \mathcal{T}_d$ against a formula φ of modal logic or of EF logic, where $|\varphi|$ is the parameter.

		Asynchronous product ($\bigotimes$)	**Synchronous Product ($\prod$)**
ML	**upper**	$(\sum_{i\in[1,d]} \lvert\mathcal{T}_i\rvert)^{O(1)} \cdot f(\lvert\varphi\rvert)$ for some primitive recursive f by Theorem 2 ([14])	$(\sum_{i\in[1,d]} \lvert\mathcal{T}_i\rvert)^{O(1)} \cdot f(\lvert\varphi\rvert)$ for some primitive recursive f by Theorem 2 ([14])
	lower	**not in** $(\sum_{i\in[1,d]} \lvert\mathcal{T}_i\rvert)^{O(1)} \cdot f(\lvert\varphi\rvert)$ for any elementary f unless $\mathsf{FPT} = \mathsf{AW}[*]$ by Theorem 8	**not in** $(\sum_{i\in[1,d]} \lvert\mathcal{T}_i\rvert)^{O(1)} \cdot f(\lvert\varphi\rvert)$ for any elementary f unless $\mathsf{P} = \mathsf{NP}$ by Theorem 9
EF	**upper**	$(\sum_{i\in[1,d]} \lvert\mathcal{T}_i\rvert)^{O(1)} \cdot f(\lvert\varphi\rvert)$ for some primitive recursive f by Theorem 2 ([14])	**not in** FPT unless $\mathsf{FPT} = \mathsf{AW}[\mathsf{SAT}]$ by Theorem 3 (Theorem 5.1 in [7])
	lower	**not in** $(\sum_{i\in[1,d]} \lvert\mathcal{T}_i\rvert)^{O(1)} \cdot f(\lvert\varphi\rvert)$ for any elementary f unless $\mathsf{P} = \mathsf{NP}$ by Theorem 10	

inverse function $\log^*$ as $\log^*(n) \stackrel{\text{def}}{=} \min\{\ell \in \mathbb{N} \mid \mathsf{Tower}(\ell) \geq n\}$. Let $f, g : \mathbb{N} \to \mathbb{N}$ be functions. We say f *is bounded by* g if $f(n) \leq g(n)$ for all but finitely many $n \in \mathbb{N}$. A function $f : \mathbb{N} \to \mathbb{N}$ is *elementary* if it can be formed from the successor function, addition, subtraction, and multiplication using compositions, projections, bounded additions and bounded multiplications (of the form $\sum_{z\leq y} g(\overline{x}, z)$ and $\prod_{z\leq y} g(\overline{x}, z)$). For our purposes it will only be important that a function $f : \mathbb{N} \to \mathbb{N}$ is bounded by an elementary function if and only if there is some $\ell \in \mathbb{N}$ such that f is bounded by the function $n \mapsto \mathsf{Tower}(\ell, n)$.

Throughout the paper, let us fix a countable set of *atomic propositions* $\mathbb{P}$ and a countable set of *atomic actions* $\mathbb{A}$. A *signature* is a pair (P, A), where $\mathsf{P} \subseteq \mathbb{P}$ is a finite set of atomic propositions and where $\mathsf{A} \subseteq \mathbb{A}$ is a finite set of atomic actions. A *transition system* is a tuple $\mathcal{T} = (S, \{S_p \mid p \in \mathsf{P}\}, \{\xrightarrow{a} \mid a \in \mathsf{A}\})$, where (P, A) is some signature, S is a set of *states*, $S_p \subseteq S$ is a *valuation* of the atomic propositions for each $p \in \mathsf{P}$, and $\xrightarrow{a} \subseteq S \times S$ is a binary *transition relation* for each $a \in \mathsf{A}$. All transition systems that appear in this paper have finite state sets, thus we denote by $|\mathcal{T}| \stackrel{\text{def}}{=} |S| + |\mathsf{P}| + |\mathsf{A}|$ the *size* of $\mathcal{T}$. A *pointed transition system* is a pair $(s, \mathcal{T})$, where $\mathcal{T}$ is a transition system and s is a state of $\mathcal{T}$ that we also denote by the *point* of $\mathcal{T}$. We sometimes write $\mathcal{T}$ to denote $(s, \mathcal{T})$ whenever s has been fixed from the context.

Formulas φ of the logic EF are given by the following grammar, where $p \in \mathbb{P}$ and $a \in \mathbb{A}$:

$$\varphi \quad ::= \quad p \quad | \quad \neg\varphi \quad | \quad \varphi \wedge \varphi \quad | \quad \langle a \rangle \varphi \quad | \quad \mathsf{EF}\varphi$$

The *size* $|\varphi|$ of a formula φ is inductively defined as $|p| \stackrel{\text{def}}{=} 1$ for each $p \in \mathbb{P}$, $|\neg\varphi| \stackrel{\text{def}}{=} |\mathsf{EF}\varphi| \stackrel{\text{def}}{=} |\langle a \rangle \varphi| \stackrel{\text{def}}{=} |\varphi| + 1$, and $|\varphi_1 \wedge \varphi_2| \stackrel{\text{def}}{=} |\varphi_1| + |\varphi_2| + 1$. We introduce the usual abbreviations $\varphi_1 \vee \varphi_2 \stackrel{\text{def}}{=} \neg(\neg\varphi_1 \wedge \neg\varphi_2)$, $\varphi_1 \to \varphi_2 \stackrel{\text{def}}{=} \neg\varphi_1 \vee \varphi_2$, $\varphi_1 \leftrightarrow \varphi_2 \stackrel{\text{def}}{=} (\varphi_1 \to \varphi_2) \wedge (\varphi_2 \to \varphi_1)$, $[a]\varphi \stackrel{\text{def}}{=} \neg\langle a \rangle \neg\varphi$ for each $a \in \mathsf{A}$ and finally $\mathsf{AG}\varphi \stackrel{\text{def}}{=} \neg\mathsf{EF}\neg\varphi$. *(Multi-) Modal logic (*ML*)* is the fragment of EF, where the EF-operator does not occur. Given a signature (P, A) and an EF-formula φ, we say that φ is *defined over* (P, A) if P (resp. A) contains the set of atomic propositions (resp. atomic actions) that appear in φ.

Given a transition system $\mathcal{T} = (S, \{S_p \mid p \in \mathsf{P}\}, \{\xrightarrow{a} \mid a \in \mathsf{A}\})$, a state $s \in S$, and an EF-formula φ over (P, A), we define $(s, \mathcal{T}) \models \varphi$ by structural induction on the formula φ as follows (1) $(s, \mathcal{T}) \models p$ if and only if $s \in S_p$, (2) $(s, \mathcal{T}) \models \neg\varphi$ if and only if $(s, \mathcal{T}) \not\models \varphi$, (3) $(s, \mathcal{T}) \models \varphi_1 \wedge \varphi_2$ if and only if $(s, \mathcal{T}) \models \varphi_1$ and $(s, \mathcal{T}) \models \varphi_2$, (4) $(s, \mathcal{T}) \models \langle a \rangle \varphi$ if and

only if $(s', \mathcal{T}) \models \varphi$ for some $s' \in S$ with $s \xrightarrow{a} s'$ and finally (5) $(s, \mathcal{T}) \models \mathsf{EF}\varphi$ if and only if $(s', \mathcal{T}) \models \varphi$ for some $s' \in S$ with $s \to^* s'$, where $\to = \bigcup_{a \in \mathsf{A}} \xrightarrow{a}$.

Products and the compositional method. Let $d \geq 1$ and let us assume a transition system $\mathcal{T}_i = (S_i, \{S_{p,i} \mid p \in \mathsf{P}_i\}, \{\xrightarrow{a}_i \mid a \in \mathsf{A}_i\})$ over the signature $(\mathsf{P}_i, \mathsf{A}_i)$ for each $i \in [1, d]$, where P_i and P_j (resp. A_i and A_j) could possibly have non-empty intersection for different $i, j \in [1, d]$. Assume P to be the union of all the P_i and assume A to be the union of all the A_i.

We define their *asynchronous product* as $\bigotimes_{i \in [1,d]} \mathcal{T}_i \stackrel{\text{def}}{=} (\overline{S}, \{\overline{S}_p \mid p \in \mathsf{P}\}, \{\xrightarrow{a} \mid a \in \mathsf{A}\})$, where $\overline{S} \stackrel{\text{def}}{=} \prod_{i \in [1,d]} S_i$, for each $(s_1, \ldots, s_d) \in \overline{S}$ and each $p \in \mathsf{P}$ we have $(s_1, \ldots, s_d) \in \overline{S}_p$ if and only if $s_i \in S_{p,i}$ for some $i \in [1, d]$, for each $(s_1, \ldots, s_d)$, $(s'_1, \ldots, s'_d) \in \overline{S}$ and each $a \in \mathsf{A}$ we have $(s_1, \ldots, s_d) \xrightarrow{a} (s'_1, \ldots, s'_d)$ if and only if there is some $i \in [1, d]$ such that $s_i \xrightarrow{a}_i s'_i$ in $\mathcal{T}_i$ and $s_j = s'_j$ for each $j \in [1, d]$ with $j \neq i$.

We define their *synchronous product*[1] as $\prod_{i \in [1,d]} \mathcal{T}_i \stackrel{\text{def}}{=} (\overline{S}, \{\overline{S}_p \mid p \in \mathsf{P}\}, \{\xrightarrow{a} \mid a \in \mathsf{A}\})$, where $\overline{S}$ and the $\overline{S}_p$ are defined as for the asynchronous product, but where the transition relation is defined as follows: For each $(s_1, \ldots, s_d), (s'_1, \ldots, s'_d) \in \overline{S}$ and each $a \in \mathsf{A}$ we have $(s_1, \ldots, s_d) \xrightarrow{a} (s'_1, \ldots, s'_d)$ if and only if for all $i \in [1, d]$ we have $s_i \xrightarrow{a}_i s'_i$ in $\mathcal{T}_i$.

Let us state the compositional method for ML and EF as proven in [14]. Here, we state it for EF logic and the asynchronous product. Also note that in [14] it has been proven for more general interpretations of the atomic propositions and more general products.

▶ **Theorem 1** ([14],Theorem 21). *The following is primitive recursive:*

INPUT: An EF *formula* φ *over* (P, A)*, where* $\mathsf{P} = \bigcup_{i=1}^{d} \mathsf{P}_i$ *and* $\mathsf{A} = \bigcup_{i=1}^{d} \mathsf{A}_i$ *for a* $d \geq 1$*.*

OUTPUT: A tuple $(\Psi_1, \ldots, \Psi_d, \beta)$*, where* $\Psi_i = \{\psi_i^j \mid j \in J_i\}$ *is a finite set of* EF *formulas over* $(\mathsf{P}_i, \mathsf{A}_i)$*, and a boolean formula* β *with variables from* $X \stackrel{def}{=} \{x_i^j \mid i \in [1, d], j \in J_i\}$ *such that for every transition system* $\mathcal{T}_i = (S_i, \{S_{p,i} \mid p \in \mathsf{P}_i\}, \{\xrightarrow{a}_i \mid a \in \mathsf{A}_i\})$ *over* $(\mathsf{P}_i, \mathsf{A}_i)$ *and every state* s_i *of* $\mathcal{T}_i$ *(*$i \in [1, d]$*) it holds:*

$$\left(\langle s_1, \ldots, s_d \rangle, \bigotimes_{i=1}^{d} \mathcal{T}_i\right) \models \varphi \qquad \Longleftrightarrow \qquad \mu \models \beta$$

Here, $\mu : X \to \{0, 1\}$ *is defined by* $\mu(x_i^j) = 1$ *if and only if* $(s_i, \mathcal{T}_i) \models \psi_i^j$*. Moreover, we have* $|\mathcal{D}(\varphi, d)| \leq g(d + |\varphi|)$ *for some primitive recursive function* g*, where* $\mathcal{D}(\varphi, d) \stackrel{def}{=} (\Psi_1, \ldots, \Psi_d, \beta)$ *is the* decomposition *of* φ *and its* size *is* $|\mathcal{D}(\varphi, d)| \stackrel{def}{=} |\beta| + \sum_{i \in [1,d]} \sum_{j \in J_i} |\psi_i^j|$*.*

▶ Remark. In [14] an analog of Theorem 1 has also been shown for the following cases: (i) φ and each formula in $\bigcup_{i \in [1,d]} \Psi_i$ is an ML formula, or (ii) φ and each formula in $\bigcup_{i \in [1,d]} \Psi_i$ is an ML formula and the asynchronous product $\bigotimes$ is replaced by the synchronous product $\prod$. Moreover, in [14] it is shown that for EF and the synchronous product such desirable decompositions as stated in Theorem 1 do *not* exist in general (even when the computability requirement is dropped).

Since model checking of EF and ML is decidable in polynomial time, Theorem 1 (whenever applicable) delivers a running time for model checking the product of finite systems in time

$$\Big(\sum_{i \in [1,d]} |\mathcal{T}_i|\Big)^{O(1)} \cdot f(d + |\varphi|)$$

[1] also known as *strong synchronization*, e.g. [7]

for a primitive recursive function f and is thus fixed-parameter tractable when $d + |\varphi|$ is the parameter. The following theorem states that it is even fixed-parameter tractable when only $|\varphi|$ is the parameter. Although not explictly stated in [14] (Corollary 22 of [14] requires $d + |\varphi|$ to be the parameter), its proof can be deduced from a refined analysis of Theorem 21 in [14] by using the notions of generalized product that were defined in [14].

▶ **Theorem 2** (A consequence from [14]). *Let* $\mathsf{op} \in \{\bigotimes, \prod\}$ *either stand for the asynchronous or synchronous product. Given a formula* φ *over* (P, A) *(where* $\mathsf{P} = \bigcup_{i=1}^{d} \mathsf{P}_i$ *and* $\mathsf{A} = \bigcup_{i=1}^{d} \mathsf{A}_i$*) and transition systems* $(\mathcal{T}_i)_{i\in[1,d]}$ *and a state* $\overline{s}$ *of* $\mathsf{op}_{i=1}^{d}\mathcal{T}_i$*, one can decide* $\left(\overline{s}, \mathsf{op}_{i=1}^{d}\mathcal{T}_i\right) \models \varphi$ *in time* $\left(\sum_{i\in[1,d]} |\mathcal{T}_i|\right)^{O(1)} \cdot f(|\varphi|)$ *for some primitive recursive function* f *in either of the following cases: (i)* φ *is some* ML *formula and* $\mathsf{op} = \prod$*, or (ii)* φ *is some* ML *formula and* $\mathsf{op} = \bigotimes$*, or (iii)* φ *is some* EF *formula and* $\mathsf{op} = \bigotimes$.

The question of fixed-parameter tractability of model checking EF for synchronous product has already been answered negatively for reachability in [7] (unless FPT = AWT[SAT]) and moreover reflects the non-decomposability of EF for synchronous product as mentioned in the end of Remark 2.

▶ **Theorem 3** (Theorem 5.1 in [7]). *Given transition systems* $\mathcal{T}_1, \ldots, \mathcal{T}_d$ *over a common signature* (A, P) *and two states* $\overline{s}, \overline{t}$ *in* $\prod_{i=1}^{d} \mathcal{T}_i$*, there is no algorithm that decides* $\overline{s} \to^* \overline{t}$ *in* $\prod_{i=1}^{d} \mathcal{T}_i$ *in time* $\left(\sum_{i=1}^{d} |\mathcal{T}_i|\right)^{O(1)} \cdot f(d)$ *for any computable function* f *unless* FPT = AWT[SAT].

3 Encoding huge numbers via sibling-ordered trees

In this section we show how one can represent numbers of size $O(n)$ by sibling-ordered trees of size $O(n)$ of depth $O(\log^*(n))$.

The idea of using wide trees of small height for proving nonelementary lower bounds has already been considered before in the literature: such wide trees, as discussed in [8], have been used in [1, 6, 9] to prove nonelementary lower bounds for parameterized complexity of satisfiability for modal logic, nonelementary lower bounds on the sizes of several normal forms of first-order logic formulas, and for the parameterized complexity of model checking first-order logic. More discussion on the particular choice of our sibling-ordered trees follows in Section 4.

In the following, let $\mathsf{P}_\ell \stackrel{\text{def}}{=} \{b_i \mid i \in [0, \ell+1]\}$ and $\mathsf{A}_\ell \stackrel{\text{def}}{=} \{a_i \mid i \in [0, \ell]\} \cup \{\Rightarrow, \Leftarrow\}$ for each $\ell \in \mathbb{N}$. For each $\ell \in \mathbb{N}$ a *pointed ℓ-sotree (for sibling-ordered tree)* that is either 0-pointed (i.e. the point does *not* satisfy proposition $b_{\ell+1}$) or 1-pointed (i.e. the point satisfies proposition $b_{\ell+1}$) and that has a *value* from $[0, \mathsf{Tower}(\ell) - 1]$ is a pointed transition system over $(\mathsf{P}_\ell, \mathsf{A}_\ell)$ that is defined inductively on ℓ:

- **Base case when $\ell = 0$:** A *pointed 0-sotree* is one of the following four pointed transition systems each with point ● over $(\mathsf{P}_0, \mathsf{A}_0)$:

The 0-pointed 0-sotree of value 0. The 0-pointed 0-sotree of value 1. The 1-pointed 0-sotree of value 0. The 1-pointed 0-sotree of value 1.

- **Inductive step for $\ell+1$:** A *pointed* $(\ell+1)$*-sotree* is a pointed transition system $(r_{\ell+1}, \mathcal{T}_{\ell+1})$ over $(\mathsf{P}_{\ell+1}, \mathsf{A}_{\ell+1})$ that can be obtained as follows:
  (1) The point $r_{\ell+1}$ does not satisfy any of the propositions $b_0, \ldots, b_\ell$.
  (2) The states of $\mathcal{T}_{\ell+1}$ are obtained by the union of $\{r_{\ell+1}\}$ and for each $j \in [0, \mathsf{Tower}(\ell)-1]$ *exactly one* of the possible two ℓ-sotrees of value j (either 0-pointed or 1-pointed), let us denote it by $(r_\ell(j), \mathcal{T}_\ell(j))$.
  (3) Add the a_ℓ-labeled transitions $\{(r_{\ell+1}, r_\ell(j)) \mid j \in [0, \mathsf{Tower}(\ell)-1]\}$ to $\mathcal{T}_{\ell+1}$.
  (4) Add the $\Leftarrow$-labeled transitions $\{(r_\ell(j), r_\ell(j')) \mid j, j' \in [0, \mathsf{Tower}(\ell)-1], j > j'\}$ and the $\Rightarrow$-labeled transitions $\{(r_\ell(j), r_\ell(j')) \mid j, j' \in [0, \mathsf{Tower}(\ell)-1], j < j'\}$ between siblings.
  (5) Define the *value* of $(r_{\ell+1}, \mathcal{T}_{\ell+1})$ as
  $$\mathsf{val}(r_{\ell+1}, \mathcal{T}_{\ell+1}) \stackrel{\text{def}}{=} \sum \left\{2^j \mid j \in [0, \mathsf{Tower}(\ell)-1] : (r_\ell(j), \mathcal{T}_\ell(j)) \text{ is 1-pointed}\right\}.$$
  Note that $\mathsf{val}(r_{\ell+1}, \mathcal{T}_{\ell+1}) \in [0, \mathsf{Tower}(\ell+1)-1]$.
  (6) In case $r_{\ell+1}$ satisfies $b_{\ell+1}$ we say $(r_{\ell+1}, \mathcal{T}_{\ell+1})$ is 1-*pointed*, otherwise we say $(r_{\ell+1}, \mathcal{T}_{\ell+1})$ is 0-*pointed*.

  Again recall that, up to isomorphism, for each $j \in [0, \mathsf{Tower}(\ell+1)-1]$ there are exactly two $\ell+1$-sotrees of value j, one being 0-pointed and one being 1-pointed.

Figure 1 shows an example 2-sotree. Let $\mathsf{size}(\ell)$ denote the *number of nodes of each ℓ-sotree*. Note $\mathsf{size}(\ell)$ can be expressed by the recurrence

$$\mathsf{size}(0) = 2 \qquad \text{and} \qquad \mathsf{size}(\ell+1) = \mathsf{Tower}(\ell) \cdot \mathsf{size}(\ell) + 1.$$

▶ **Lemma 4.** *For each $\ell \geq 3$ we have $\mathsf{size}(\ell+1) \leq \mathsf{Tower}(\ell)^2$.*

Recall that the *value* of each ℓ-sotree lies in the interval $[0, \mathsf{Tower}(\ell)-1]$ and for every value in this interval there is an ℓ-sotree with this value. In the following, for each $j, \ell \in \mathbb{N}$ with $j \in [0, \mathsf{Tower}(\ell)-1]$ we define $\Upsilon_\ell(j) \stackrel{\text{def}}{=} (r_\ell(j), \mathcal{T}_\ell(j))$ to be the (unique) 0-pointed ℓ-sotree of value j.

Next, we aim at defining formulas that allow us to compare the values of ℓ-sotrees with small formulas.

Let $(s, \mathcal{T})$ be a pointed transition system with $\mathcal{T} = (S, \{S_p \mid p \in \mathsf{P}\}, \{\xrightarrow{a} \mid a \in \mathsf{A}\})$ and let α be some label. We define the *asynchronous α-extension of $\mathcal{T}$* to be the pointed transition system with point s over the signature $(\{\alpha\} \times \mathsf{P}, \{\alpha\} \times \mathsf{A})$ that can be obtained from $\mathcal{T}$ by setting $S_{(\alpha,p)} \stackrel{\text{def}}{=} S_p$ and by setting $\xrightarrow{(\alpha,a)} \stackrel{\text{def}}{=} \xrightarrow{a}$. Likewise, we define the *synchronous α-extension of $\mathcal{T}$* to be the pointed transition system with point s over the signature $(\{\alpha\} \times \mathsf{P}, \mathsf{A})$ that can be obtained from $\mathcal{T}$ by setting $S_{(\alpha,p)} \stackrel{\text{def}}{=} S_p$ and keeping $\xrightarrow{a}$ as it is. The *value* val of each asynchronous/synchronous α-extension of a pointed ℓ-sotree $(r, \mathcal{T})$ is inherited from $\mathsf{val}(r, \mathcal{T})$.

The following lemma states that one can construct ML formulas of size $O(\log^*(n))$ that allow us to compare the value of the asynchronous product of an asynchronous α-extension and an asynchronous β-extension of ℓ-sotrees of value $O(n)$. The formulas are inspired from [10], but, as will be discussed in Section 4, we require the additional transitions $\Leftarrow$ and $\Rightarrow$ for expressing order.

▶ **Lemma 5** (Formulas for the asynchronous product of ℓ-sotrees)**.** *For each $\ell \in \mathbb{N}$ and labels α, β there are formulas $(\mathsf{eq}_\ell^\otimes(\alpha,\beta))_{\ell\in\mathbb{N}} = (\mathsf{eq}_\ell^\otimes)_{\ell\in\mathbb{N}}$ and $(\mathsf{less}_\ell^\otimes(\alpha,\beta))_{\ell\in\mathbb{N}} = (\mathsf{less}_\ell^\otimes)_{\ell\in\mathbb{N}}$, each over the signature $(\{\alpha,\beta\} \times \mathsf{P}_\ell, \{\alpha,\beta\} \times \mathsf{A}_\ell)$ and each of size $2^{O(\ell)}$ such that the asynchronous α-extension $(r, \mathcal{T})$ of each pointed ℓ-sotree the asynchronous β-extension $(r', \mathcal{T}')$ of each pointed ℓ-sotree the following holds:*

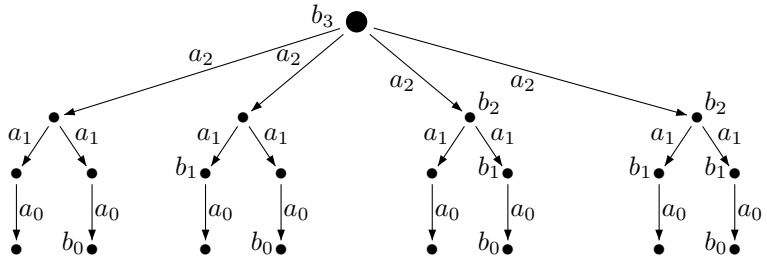

Figure 1 The 1-pointed 2-sotree of value 12, where the horizontal transitions $\Rightarrow$ and $\Leftarrow$ between siblings are omitted.

(1) $(r,\mathcal{T}) \otimes (r',\mathcal{T}') \models \mathsf{eq}_\ell^\otimes$ *if and only if* $\mathsf{val}(r,\mathcal{T}) = \mathsf{val}(r',\mathcal{T}')$,
(2) $(r,\mathcal{T}) \otimes (r',\mathcal{T}') \models \mathsf{less}_\ell^\otimes$ *if and only if* $\mathsf{val}(r,\mathcal{T}) < \mathsf{val}(r',\mathcal{T}')$, *and*
(3) $(r,\mathcal{T}) \otimes (r',\mathcal{T}') \models \mathsf{succ}_\ell^\otimes$ *if and only if* $\mathsf{val}(r,\mathcal{T}) + 1 = \mathsf{val}(r',\mathcal{T}')$.

Proof. We define the formulas by induction on ℓ. For the induction base $\ell = 0$ we put:
(1) $\mathsf{eq}_0^\otimes \stackrel{\text{def}}{=} \langle\alpha,a_0\rangle\langle\beta,a_0\rangle\,((\alpha,b_0) \leftrightarrow (\beta,b_0))$.
(2,3) $\mathsf{less}_0^\otimes \stackrel{\text{def}}{=} \mathsf{succ}_0^\otimes \stackrel{\text{def}}{=} \langle\alpha,a_0\rangle\langle\beta,a_0\rangle\,(\neg(\alpha,b_0) \wedge (\beta,b_0))$.
For the induction step, we define:

(1) $\mathsf{eq}_{\ell+1}^\otimes \stackrel{\text{def}}{=} [\alpha,a_{\ell+1}][\beta,a_{\ell+1}]\left(\mathsf{eq}_\ell^\otimes \longrightarrow ((\alpha,b_\ell) \leftrightarrow (\beta,b_\ell))\right)$.
(2) $\mathsf{less}_{\ell+1}^\otimes \stackrel{\text{def}}{=} \langle\alpha,a_{\ell+1}\rangle\langle\beta,a_{\ell+1}\rangle\varphi_{\ell+1}^\otimes$, where
$$\varphi_{\ell+1}^\otimes \stackrel{\text{def}}{=} \left(\mathsf{eq}_\ell^\otimes \wedge \neg(\alpha,b_\ell) \wedge (\beta,b_\ell) \wedge [\alpha,\Rightarrow][\beta,\Rightarrow]\left(\mathsf{eq}_\ell^\otimes \rightarrow ((\alpha,b_\ell) \leftrightarrow (\beta,b_\ell))\right)\right),$$
thus expressing that there is an $i \in [0,\mathsf{Tower}(\ell)-1]$ such that the i^{th} bit of $\mathsf{val}(r,\mathcal{T})$ is *not* set, the i^{th} bit of $\mathsf{val}(r',\mathcal{T}')$ is set and moreover $\mathsf{val}(r,\mathcal{T})$ and $\mathsf{val}(r',\mathcal{T}')$ agree on the j^{th} bit for all $i < j \leq \mathsf{Tower}(\ell)-1$.
(3) $\mathsf{succ}_{\ell+1}^\otimes \stackrel{\text{def}}{=} \langle\alpha,a_{\ell+1}\rangle\langle\beta,a_{\ell+1}\rangle(\varphi_{\ell+1}^\otimes \wedge [\alpha,\Leftarrow](\alpha,b_\ell) \wedge [\beta,\Leftarrow]\neg(\beta,b_\ell))$, thus expressing that there is an $i \in [0,\mathsf{Tower}(\ell)-1]$ such that the i^{th} bit of $\mathsf{val}(r,\mathcal{T})$ is *not* set, the i^{th} bit of $\mathsf{val}(r',\mathcal{T}')$ is set, $\mathsf{val}(r,\mathcal{T})$ and $\mathsf{val}(r',\mathcal{T}')$ agree on the j^{th} bit for all $i < j \leq \mathsf{Tower}(\ell)-1$ and finally the j^{th} bit of $\mathsf{val}(r,\mathcal{T})$ is set whereas the j^{th} bit of $\mathsf{val}(r',\mathcal{T}')$ is *not* set for each $0 \leq j < i$. ◀

Similar formulas as in Lemma 5 can be shown for the synchronous product.

▶ **Lemma 6** (Formulas for the synchronous product of ℓ-sotrees)**.** *For each $\ell \in \mathbb{N}$ and labels α,β there are formulas $(\mathsf{eq}_\ell^\times(\alpha,\beta))_{\ell\in\mathbb{N}} = (\mathsf{eq}_\ell^\times)_{\ell\in\mathbb{N}}$, $(\mathsf{succ}_\ell^\times(\alpha,\beta))_{\ell\in\mathbb{N}} = (\mathsf{succ}_\ell^\times)_{\ell\in\mathbb{N}}$, and $(\mathsf{less}_\ell^\times(\alpha,\beta))_{\ell\in\mathbb{N}} = (\mathsf{less}_\ell^\times)_{\ell\in\mathbb{N}}$, each over the signature $(\{\alpha,\beta\} \times \mathsf{P}_\ell, \mathsf{A}_\ell)$ and each of size $2^{O(\ell)}$ such that for the synchronous α-extension $(r,\mathcal{T})$ of each pointed ℓ-sotree and the synchronous β-extension $(r',\mathcal{T}')$ of each pointed ℓ-sotree the following holds:*
(1) $(r,\mathcal{T}) \times (r',\mathcal{T}') \models \mathsf{eq}_\ell^\times$ *if and only if* $\mathsf{val}(r,\mathcal{T}) = \mathsf{val}(r',\mathcal{T}')$,
(2) $(r,\mathcal{T}) \times (r',\mathcal{T}') \models \mathsf{less}_\ell^\times$ *if and only if* $\mathsf{val}(r,\mathcal{T}) < \mathsf{val}(r',\mathcal{T}')$, *and*
(3) $(r,\mathcal{T}) \times (r',\mathcal{T}') \models \mathsf{succ}_\ell^\times$ *if and only if* $\mathsf{val}(r,\mathcal{T}) + 1 = \mathsf{val}(r',\mathcal{T}')$.

4 Overview of the proofs

The reason for choosing our particular encoding via sibling-ordered trees from Section 3 is that we would like to provide possibly short proofs for our main results. Let us discuss this in more detail.

For showing, unless FPT = AW[$*$], that for given systems $\mathcal{T}_1, \ldots, \mathcal{T}_d$ and an ML formula φ there is no algorithm that decides whether φ holds in the asynchronous product of $\mathcal{T}_1, \ldots, \mathcal{T}_d$ and runs in time $(\sum_i |\mathcal{T}_i|)^{O(1)} \cdot f(|\varphi|)$ for any elementary function f, we reduce from model checking first-order logic on finite words with order from [9] (which has been shown not be decidable in time $|\mathcal{W}|^{O(1)} \cdot f(|\varphi|)$ for any elementary function unless FPT = AW[$*$] in [9]). In fact, it might be possible to prove our result directly by working with the asynchronous product of trees from [8] but this would have involved a complicated encoding of 3-CNF formulas as in [9] and hence result in a proof that is technically much more involved than our current proof. Moreover it would not shed new light into the problem.

In our reduction from model checking first-order logic over words we encode each position of the input word by one of our sibling-ordered trees (from Section 3) and present the whole word by an asynchronous product of these sibling-ordered trees such that ML formulas of size $O(\log^*(n))$ can test whether two such marked trees are related by the order. For encoding this we cannot use trees that were used in [10] for proving nonelementary lower bounds for satisfiability checking two-dimensional modal logic since it is not at all clear how to simulate the above-mentioned order relation between an asynchronous product between two of them (in [10] only the successor relation was simulated with involved technical machinery).

We are able to prove under the weaker assumption P $\neq$ NP that there is neither such an algorithm that runs in time $(\sum_i |\mathcal{T}_i|)^{O(1)} \cdot f(|\varphi|)$ for elementary function f for model checking ML on the synchronous product of systems nor one for model checking EF on the asynchronous product of systems. To obtain this, one possibility could have been to reduce from the model checking problem of monadic second-order logic over finite words (which has been shown not to be decidable in time $|\mathcal{W}|^{O(1)} \cdot f(|\varphi|)$ for any elementary function f unless P = NP in [9]) but for this result it turned out that a direct proof is simpler than to encode monadic second-order quantification. We reduce from the NP-complete $n \times n$-tiling problem.

Let us point out the crucial differences to model checking the asynchronous product and why we are able to weaken our complexity-theoretic assumption from FPT $\neq$ AW[$*$] to P $\neq$ NP when considering the synchronous product. The asynchronous product of systems $\mathcal{T}_1, \ldots, \mathcal{T}_d$ is generally also exponentially big in the input size, but the out-degree in $\otimes_i \mathcal{T}_i$ is only polynomially bounded. When encoding the NP-hard $n \times n$-tiling problem as instances of model checking modal logic on the synchronous product (and thus assuming the weaker P $\neq$ NP assumption), the idea is to use for each of the n^2 coordinates (x, y) one sibling-ordered tree $\mathcal{T}_{x,y}$ for representing this coordinate to be colored with a tile type. Guessing one coloring (of the exponentially many) to the tiling problem can be achieved in *one* step by an ML formula in the synchronous product, whereas it would take exponentially many steps in the asynchronous product (thus, the lower bound cannot be applied to the asynchronous product). Hence, as mentioned above, for proving lower bounds for the asynchronous product, we had to make a stronger complexity theoretic assumption, namely FPT $\neq$ AW[$*$].

5 Model Checking ML on the asynchronous product

In this section we prove that model checking ML for asynchronous product is not fixed-parameter tractable with an elementary function in the size of the formula unless FPT = AW[$*$]. We reduce from an orthogonal hardness result for model checking first-order logic on words from [9].

Given a finite alphabet Σ the signature of first-order (FO) formulas for words over Σ is $(\Sigma, <)$, where each $a \in \Sigma$ is a unary symbol (the letter predicate) and $<$ (the order on positions). We use the following lower bound result from model checking first-order logic on words.

▶ **Theorem 7** ([9]). *Let f be an elementary function. Then there is no algorithm that decides*
INPUT: A word $\mathcal{W}$ and some FO *sentence φ each over some alphabet Σ.*
QUESTION: $\mathcal{W} \models \varphi$?
in time $|\mathcal{W}|^{O(1)} \cdot f(|\varphi|)$ unless FPT $=$ AW$[*]$.

We can now present the main result of this section.

▶ **Theorem 8.** *Let f be an elementary function. Then there is no algorithm that decides*
INPUT: Transition systems $\mathcal{T}_1, \dots, \mathcal{T}_d$, a state $\overline{s}$ of $\bigotimes_{i=1}^{d} \mathcal{T}_i$ and an ML *formula ψ.*
QUESTION: $(\overline{s}, \bigotimes_{i\in[1,d]} \mathcal{T}_i) \models \psi$?
in time $(\sum_{i=1}^{d} |\mathcal{T}_i|)^{O(1)} \cdot f(|\psi|)$ unless FPT $=$ AW$[*]$.

Proof. The idea is to show that the existence of such an elementary function f contradicts Theorem 7. So for the sake of contradiction, let us assume that there were an elementary function f and an algorithm that decides the model checking problem for a given ML formula ψ and a state $\overline{s}$ in the asynchronous product of d finite systems $\mathcal{T}_1, \dots, \mathcal{T}_d$ in time $(\sum_{i\in[1,d]} |\mathcal{T}_i|)^{O(1)} \cdot f(|\psi|)$. Let us show that this algorithm can be used for model checking FO over words. For this, let $\mathcal{W} = a_0 \cdots a_{n-1}$, where $a_i \in \Sigma$ for each $i \in [0, n-1]$ and let $\varphi = \exists x_1 \forall x_2 \cdots \exists x_{2k-1} \forall x_{2k} \psi(x_1, \dots, x_{2k})$ be an FO sentence over Σ in prenex normal form with alternating quantifiers without loss of generality. We will compute some $d \geq 1$, transition systems $\mathcal{T}_1, \dots, \mathcal{T}_d$, some state $\overline{s}$ of $\mathcal{T} \stackrel{\text{def}}{=} \bigotimes_{i=1}^{d} \mathcal{T}_i$ and some ML formula ψ such that

(i) $\mathcal{W} \models \varphi$ if and only if $(\overline{s}, \mathcal{T}) \models \psi$,
(ii) $d \leq O(|\varphi|)$,
(iii) $|\mathcal{T}_i| \leq |\mathcal{W}|^{O(1)}$ for each $i \in [1, d]$,
(iv) $|\psi| \leq O(|\varphi|) \cdot 2^{O(\log^*(|\mathcal{W}|))}$, and
(v) each $\mathcal{T}_i$, $\overline{s}$ and ψ is computable in time $|\mathcal{W}|^{O(1)} \cdot g(|\varphi|)$ for an elementary g.

Before we show how to compute the $\mathcal{T}_i$, $\overline{s}$ and ψ, let us start with the desired contradiction. Note that an elementary computation of such $\mathcal{T}_i$, $\overline{s}$ and ψ as stated above would lead to an algorithm that decides $\mathcal{W} \models \varphi$ in time $|\mathcal{W}|^{O(1)} \cdot g(|\varphi|)$ (for the reduction, Point (v) from above) plus

$$
\begin{aligned}
& (\sum_{i\in[1,d]} |\mathcal{T}_i|)^{O(1)} \cdot f(|\psi|) \\
\overset{\text{(iii),(iv)}}{\leq}\ & (d \cdot |\mathcal{W}|^{O(1)})^{O(1)} \cdot f(O(|\varphi|) \cdot 2^{O(\log^*(|\mathcal{W}|))}) \\
\leq\ & |\mathcal{W}|^{O(1)} \cdot d^{O(1)} \cdot f(O(|\varphi|) \cdot 2^{O(\log^*(|\mathcal{W}|))}) \\
\overset{\text{(ii)}}{\leq}\ & |\mathcal{W}|^{O(1)} \cdot |\varphi|^{O(1)} \cdot f(O(|\varphi|) \cdot 2^{O(\log^*(|\mathcal{W}|))})
\end{aligned}
$$

by the binomial theorem. Furthermore, it is easy to see that there exist elementary functions f_1, f_2 such that the latter is bounded by

$$
|\mathcal{W}|^{O(1)} \cdot f_1(|\varphi|) \cdot f_2(2^{O(|\log^*(|\mathcal{W}|))}) \quad \leq \quad |\mathcal{W}|^{O(1)} \cdot f_1(|\varphi|)
$$

since $f_2(2^{O(\log^*(|\mathcal{W}|))})$ is bounded by a sublinearly growing function in $|\mathcal{W}|$ (and is thus in particular bounded by $|\mathcal{W}|^{O(1)}$). The latter contradicts Theorem 7. This concludes the analysis of the overall running time of our reduction.

Let us conclude to show that we can compute transition systems $\mathcal{T}_1, \dots, \mathcal{T}_d$, state $\overline{s}$ of $\mathcal{T} \stackrel{\text{def}}{=} \bigotimes_{i=1}^{d} \mathcal{T}_i$ and some ML formula ψ such that moreover Points (i) to (v) from above

hold. Recall that $\mathcal{W} = a_0 \cdots a_{n-1}$. Let us first compute the smallest $\ell \geq 1$ such that $\mathsf{Tower}(\ell-1) < n \leq \mathsf{Tower}(\ell)$. Note that $\ell \leq O(\log^*(n)) \leq O(\log^*(|\mathcal{W}|))$. Let us compute the 0-pointed ℓ-sotrees $\Upsilon_\ell(0), \ldots, \Upsilon_\ell(n-1)$, with their points $r_\ell(0), \ldots, r_\ell(n-1)$, respectively.

Recall that each pointed ℓ-sotree was defined over the signature $(\mathsf{P}_\ell, \mathsf{A}_\ell)$. Without loss of generality we assume that $\Sigma \cap \mathsf{P}_\ell = \emptyset$.

Let $\mathcal{U}$ be the pointed transition system over $(\mathsf{P}_\ell \cup \Sigma, \mathsf{A}_\ell \cup \{\gamma\})$ that one obtains from the disjoint union of $\Upsilon_\ell(0), \ldots, \Upsilon_\ell(n-1)$ and some fresh state s (which will be the point of $\mathcal{U}$) and by adding the atomic proposition $a_j \in \Sigma$ to $\mathcal{U}$'s state $r_\ell(j)$ and connecting s with $r_\ell(j)$ with a γ-labeled transition for each $j \in [0, n-1]$. Recall that $\varphi = \exists x_1 \forall x_2 \cdots \exists x_{2k-1} \forall x_{2k}$ $\psi(x_1, \ldots, x_{2k})$ is our input FO formula. We set $d \stackrel{\text{def}}{=} 2k$, hence $d \leq O(|\varphi|)$ and thus Point (ii) is shown. We define $\mathcal{T}_i$ to be the asynchronous i-extension of $\mathcal{U}$ for each $i \in [1, 2k]$. Recall that $\mathsf{size}(\ell)$ denotes the size of each $\Upsilon_\ell(j)$. Hence

$$|\mathcal{T}_i| \;\leq\; O(n \cdot \mathsf{size}(\ell)) \;\stackrel{\text{Lemma 4}}{\leq}\; O(n \cdot \mathsf{Tower}(\ell-1)^2) \;\leq\; O(n^3) \;\leq\; |\mathcal{W}|^{O(1)}$$

for each $i \in [1, d]$ and thus (iii) is shown. We define the pointed transition system $\mathcal{T} \stackrel{\text{def}}{=} \bigotimes_{i \in [1,2k]} \mathcal{T}_i$ with point $\overline{s} \stackrel{\text{def}}{=} (s, \ldots, s)$. The intuition is that each position $j \in [0, n-1]$ of $\mathcal{W}$ will be presented by $\Upsilon_\ell(j)$ and $\mathcal{U}$ will correspond to $\mathcal{W}$. The transition system $\mathcal{T}$ consists of the asynchronous product of $d = 2k$ copies of $\mathcal{U}$ since φ consists of $2k$ quantifiers. Thus, the pointed transition system $\mathcal{T}_i$ will handle the position where variable x_i will be bound to, for each $i \in [1, 2k]$.

Finally, we define the ML-formula ψ as follows

$$\psi \;\stackrel{\text{def}}{=}\; \langle 1, \gamma \rangle [2, \gamma] \cdots \langle 2k-1, \gamma \rangle [2k, \gamma]\; \widehat{\psi},$$

where the ML formula $\widehat{\psi}$ is obtained from φ by replacing each occurrence of $a(x_i)$ by (i, a) and by replacing each occurrence of $x_i < x_{i'}$ by $\mathsf{less}^{\otimes}_\ell(i, i')$, where each formula $\mathsf{less}^{\otimes}_\ell(i, i')$ is taken from Lemma 5 for $\alpha = i$ and $\beta = i'$. It is straightforward to verify that we have $\mathcal{W} \models \varphi$ if and only if $\left(\overline{s}, \bigotimes_{i=1}^{d} \mathcal{T}_i\right) \models \psi$, which delivers Point (i). Recall that $\ell \leq O(\log^*(|\mathcal{W}|))$ and the size of each formula $\mathsf{less}^{\otimes}_\ell(i, i')$ is bounded by $2^{O(\ell)}$ by Lemma 5, thus Point (iv) holds. Point (v) is easy to see by the fact that each $\mathcal{T}_i$ is computable in time $|\mathcal{T}_i|^{O(1)}$ and the formula ψ is computable in time $|\psi|^{O(1)}$ and by using Points (ii), (iii) and (iv) and analogous arguments as for the above running time analysis. ◀

An adaption of the latter proof can be carried out for model checking ML on the synchronous product. Remarkably, for model checking the synchronous product the complexity-theoretic assumption can be indeed be weakened to $\mathsf{P} \neq \mathsf{NP}$ as will be shown in the next section.

6 Model checking ML on the synchronous product and model checking EF on the asynchronous product

In this section we prove that the fixed-parameter tractability of model checking ML for synchronous product and EF for asynchronous product is cannot be witnessed by an elementary running time in the size of the formula unless $\mathsf{P} = \mathsf{NP}$.

We recall tiling systems for this. A *tiling system* is a tuple $\mathcal{S} = (\Theta, \mathbb{H}, \mathbb{V})$, where Θ is a finite set of *tile types*, $\mathbb{H} \subseteq \Theta \times \Theta$ is a *horizontal matching relation*, and $\mathbb{V} \subseteq \Theta \times \Theta$ is a *vertical matching relation*. A mapping $\sigma : [0, n-1]^2 \to \Theta$ (where $n \geq 0$) is an $n \times n$*-solution for* $\mathcal{S}$ if for all $x, y \in [0, n-1]$ the following holds: (i) if $x < n-1$, $\sigma(x, y) = \theta$, and $\sigma(x+1, y) = \theta'$,

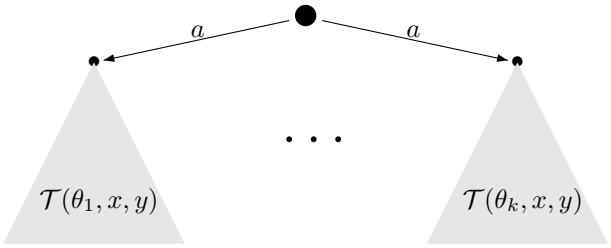

Figure 2 The pointed transition system $\mathcal{T}_{x,y}$ for each $x \in [0, n-1]$ and each $y \in [1, n-1]$.

then $(\theta, \theta') \in \mathbb{H}$, and (ii) if $y < n-1$, $\sigma(x,y) = \theta$, and $\sigma(x, y+1) = \theta'$, then $(\theta, \theta') \in \mathbb{V}$. Let $\mathcal{W} = \theta_0 \cdots \theta_{n-1} \in \Theta^n$ be a word. By $\mathsf{Sol}_n(\mathcal{S}, \mathcal{W})$ we denote the set of all $n \times n$-solutions σ for $\mathcal{S}$ such that $\sigma(x,0) = \theta_x$ for all $x \in [0, n-1]$. For a fixed tiling system $\mathcal{S}$, its $n \times n$-*tiling problem* asks for a given word $\mathcal{W} \in \Theta^n$, whether $\mathsf{Sol}_n(\mathcal{S}, \mathcal{W}) \neq \emptyset$ holds. It is folklore that there exists a *fixed* tiling system $\mathcal{S}_0$ whose $n \times n$-tiling problem is NP-hard; see also [3]. Let us fix such a tiling system $\mathcal{S}_0 = (\Theta_0, \mathbb{H}_0, \mathbb{V}_0)$ for the rest of this section.

▶ **Theorem 9.** *Let f be an elementary function. Then there is no algorithm that decides*
INPUT: Transition systems $\mathcal{T}_1, \ldots, \mathcal{T}_d$, a state $\overline{s}$ of $\prod_{i=1}^d \mathcal{T}_i$ and an ML *formula ψ.*
QUESTION: $(\overline{s}, \prod_{i=1}^d \mathcal{T}_i) \models \psi$?
in time $(\sum_{i=1}^d |\mathcal{T}_i|)^{O(1)} \cdot f(|\psi|)$ unless P = NP.

Proof. The idea is to show that the existence of such an elementary function f implies that the $n \times n$-tiling problem for $\mathcal{S}_0$ is in P and thus P = NP. So for the sake of contradiction, let us assume that there were an elementary function f and an algorithm that decides the model checking problem for a given ML formula ψ and a state $\overline{s}$ in the synchronous product of d finite systems $\mathcal{T}_1, \ldots, \mathcal{T}_d$ in time $(\sum_{i=1}^d |\mathcal{T}_i|)^{O(1)} \cdot f(|\psi|)$. Let us show that this algorithm can be used for deciding the $n \times n$-tiling problem for $\mathcal{S}_0$ in polynomial time. Recall $\mathcal{S}_0 = (\Theta_0, \mathbb{H}_0, \mathbb{V}_0)$. Let us assume $\Theta_0 = \{\theta_1, \ldots, \theta_k\}$. Let $\mathcal{W} = \mathcal{W}_0 \cdots \mathcal{W}_{n-1} \in \Theta_0^n$ be an input word to the $n \times n$-tiling problem for $\mathcal{S}_0$. We will compute transition systems $\{\mathcal{T}_{x,y} \mid x, y \in [0, n-1]\}$, some state $\overline{s}$ of $\prod_{x,y \in [0,n-1]} \mathcal{T}_{x,y}$ and some ML formula ψ such that

(i) $\mathsf{Sol}_n(\mathcal{S}_0, \mathcal{W}) \neq \emptyset$ if and only if $(\overline{s}, \prod_{x,y \in [0,n-1]} \mathcal{T}_{x,y}) \models \psi$,
(ii) $|\mathcal{T}_{x,y}| \leq |\mathcal{W}|^{O(1)}$ for each $x, y \in [0, n-1]$,
(iii) $|\psi| \leq 2^{O(\log^*(|\mathcal{W}|))}$, and
(iv) each $\mathcal{T}_{x,y}$, $\overline{s}$ and ψ is computable in time $|\mathcal{W}|^{O(1)}$.

The proof that Points (i) to (iv) lead to an overall algorithm that decides the non-emptiness of $\mathsf{Sol}_n(\mathcal{S}_0, \mathcal{W})$ in time $|\mathcal{W}|^{O(1)}$ works analogously as the proof of Theorem 8 and is therefore omitted. Recall that $\mathcal{W} = \mathcal{W}_0 \cdots \mathcal{W}_{n-1} \in \Theta^n$ and let us assume without loss of generality that $n \geq 3$.

Let us first compute the smallest $\ell \geq 1$ such that $\mathsf{Tower}(\ell - 1) < n \leq \mathsf{Tower}(\ell)$. Note that $\ell \leq O(\log^*(n)) \leq O(\log^*(|\mathcal{W}|))$. Recall that $\Upsilon_\ell(j)$ denotes the 0-pointed ℓ-sotree of value j for each $j \in [0, \mathsf{Tower}(\ell) - 1]$. Let $\Upsilon_\ell^\alpha(i)$ (resp. $\Upsilon_\ell^\beta(i)$) denote the synchronous α-extension (resp. synchronous β-extension) of $\Upsilon_\ell(j)$ for each $j \in [0, \mathsf{Tower}(\ell) - 1]$.

We will call each pointed transition system $\mathcal{T}_{x,y}$ a *component* of the product transition system $\mathcal{T} \stackrel{\text{def}}{=} \prod_{x,y \in [0,n-1]} \mathcal{T}_{x,y}$. Let us mention the purpose of the pointed transition systems $\mathcal{T}_{x,y}$. With points denoted by ●, Figure 2 shows the pointed transition system $\mathcal{T}_{x,y}$ whenever $1 \leq y \leq n-1$ and Figure 3 shows them whenever $y = 0$, where the pointed transition systems

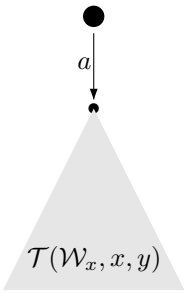

Figure 3 The pointed transition system $\mathcal{T}_{x,y}$ for each $x \in [0, n-1]$ and $y = 0$.

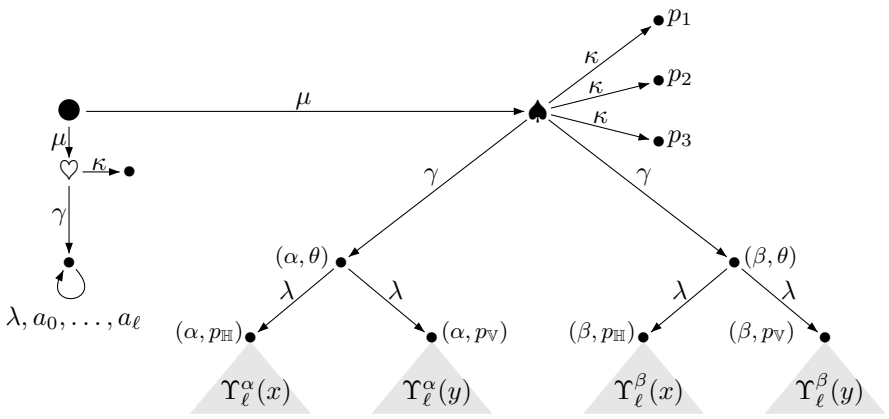

Figure 4 The pointed transition system $\mathcal{T}(\theta, x, y)$ for each $\theta \in \Theta_0$ and each $x, y \in [0, n-1]$.

$\mathcal{T}(\theta, x, y)$ for each $\theta \in \Theta_0$ and each $x, y \in [0, n-1]$ will be described in more detail below. Note that we have hereby specified the point $\overline{s}$ of $\mathcal{T} = \prod_{x,y \in [0,n-1]} \mathcal{T}_{x,y}$. Each a-successor of $\overline{s}$ in $\mathcal{T}$ hence corresponds to a choice function from $[0, n-1] \times [0, n-1]$ to Θ_0, thus selecting for each $(x, y) \in [0, n-1] \times [0, n-1]$, some tile type $\theta_{x,y}$ from Θ_0 (by taking in the component $\mathcal{T}_{x,y}$ the a-successor to the point of $\mathcal{T}(\theta_{x,y}, x, y)$); however for each (x, y) with $y = 0$ we only allow to choose $\theta_{x,y} = \mathcal{W}_x$, since we would like to restrict ourselves to choice functions ρ with $\rho(x, 0) = \mathcal{W}_x$ for each $x \in [0, n-1]$.

Having selected a choice function that respects our input word $\mathcal{W}$, let us now comment on the pointed transition systems $\mathcal{T}(\theta, x, y)$ with point $\bullet$ as depicted in Figure 4. Let assume, for the moment, that for each $\mathcal{T}(\theta, x, y)$ we are either in state $\heartsuit$ or in state $\spadesuit$. Then in the corresponding pointed synchronous product transition system the formula $\varphi_{two} \stackrel{\text{def}}{=} \langle\kappa\rangle(p_1 \wedge p_2) \wedge \neg\langle\kappa\rangle(p_1 \wedge p_2 \wedge p_3)$ holds if and only if exactly two components are in state $\spadesuit$. Note that since $k = |\Theta_0|$ is a fixed constant we have $|\mathcal{T}_{x,y}| \leq O(\mathsf{size}(\ell)) \stackrel{\text{Lemma 4}}{\leq} O(\mathsf{Tower}(\ell - 1)^2) \leq O(n^2) \leq |\mathcal{W}|^{O(1)}$ for each $x, y \in [0, n-1]$ and thus (ii) is shown.

Let us define the auxiliary formulas $\varphi_\alpha \stackrel{\text{def}}{=} \bigvee_{\theta \in \Theta_0} (\alpha, \theta)$ and $\varphi_\beta \stackrel{\text{def}}{=} \bigvee_{\theta \in \Theta_0} (\beta, \theta)$. Let us list our final formula ψ before we comment on it below.

$$\langle a\rangle[\mu]\Big[\varphi_{two} \to [\gamma]\Big(\varphi_\alpha \wedge \varphi_\beta \wedge \langle\lambda\rangle((\alpha,p_{\mathbb{H}}) \wedge (\beta,p_{\mathbb{H}}) \wedge \mathsf{eq}_\ell^\times(\alpha,\beta)) \wedge \langle\lambda\rangle((\alpha,p_{\mathbb{V}}) \wedge (\beta,p_{\mathbb{V}}) \wedge \mathsf{succ}_\ell^\times(\alpha,\beta))\Big) \to \varphi_{\mathbb{V}}\Big]$$

$$\wedge \Big[\varphi_{two} \to [\gamma]\Big(\varphi_\alpha \wedge \varphi_\beta \wedge \langle\lambda\rangle((\alpha,p_{\mathbb{H}}) \wedge (\beta,p_{\mathbb{H}}) \wedge \mathsf{succ}_\ell^\times(\alpha,\beta)) \wedge \langle\lambda\rangle((\alpha,p_{\mathbb{V}}) \wedge (\beta,p_{\mathbb{V}}) \wedge \mathsf{eq}_\ell^\times(\alpha,\beta))\Big) \to \varphi_{\mathbb{H}}\Big]$$

where

$$\varphi_{\mathbb{V}} \stackrel{\text{def}}{=} \bigvee_{(\theta,\theta')\in\mathbb{V}_0} (\alpha,\theta) \wedge (\beta,\theta') \quad \text{and} \quad \varphi_{\mathbb{H}} \stackrel{\text{def}}{=} \bigvee_{(\theta,\theta')\in\mathbb{H}_0} (\alpha,\theta) \wedge (\beta,\theta').$$

When evaluating it from $(\overline{s}, \prod_{x,y\in[0,n-1]} \mathcal{T}_{x,y})$, the formula ψ can be read as follows: There is a choice function $\rho \in \Theta_0^{[0,n-1]\times[0,n-1]}$ that respects our input word $\mathcal{W}$ (this corresponds to the part $\langle a\rangle$) such that whenever we choose (this corresponds to $[\mu]$) precisely two (this is realized by going to $\heartsuit$ and $\spadesuit$ and is is controlled by the formula φ_{two}) different elements (x,y) and (x',y') from $[0,n-1]\times[0,n-1]$ and exactly one of these two is "colored" with α and the other with β (by going along the transition γ and checking the formula $\varphi_\alpha \wedge \varphi_\beta$) we have that, firstly, $x = x'$ and $y' = y+1$ implies that $(\rho(x,y),\rho(x',y')) \in \mathbb{V}_0$ and, secondly, $x' = x+1$ and $y = y'$ implies that $(\rho(x,y),\rho(x',y')) \in \mathbb{H}_0$. Thus $\mathsf{Sol}(\mathcal{S}_0,\mathcal{W}) \neq \emptyset$ if and only if $(\overline{s}, \prod_{x,y\in[0,n-1]} \mathcal{T}_{x,y}) \models \psi$, thus Point (i) holds. The definition of ψ shows that $|\psi| \leq 2^{O(\ell)} \leq 2^{O(\log^*(|\mathcal{W}|))}$ by Lemma 6 and thus Point (iii) follows. The reader easily verifies that the transition systems $\mathcal{T}_{x,y}$, the state $\overline{s}$ of $\mathcal{T}$ and the formula ψ can in total be computed in time $|\mathcal{W}|^{O(1)}$, which delivers Point (iv) and completes the proof of the theorem. ◀

The following theorem states an analogous result for model checking EF on the asynchronous product. One can recycle the proof of Theorem 9, replace the synchronous α/β-extension by asynchronous α/β-extension, respectively, replacing the $\mathsf{succ}_\ell^\times(\alpha,\beta)$ by $\mathsf{succ}_\ell^\otimes(\alpha,\beta)$, replacing $\langle e\rangle$ by EF and $[e]$ by AG in a relativized fashion by introducing fresh atomic propositions that allow us to correctly mimick one transition in the synchronous product by a sequence of transitions in the asynchronous product. Recall that each transition system $\mathcal{T}_{x,y}$ is essentially a tree (almost) except for the substructures of the form $\Upsilon_\ell^{\alpha/\beta}(x/y)$. Simulating each modality $\langle z\rangle$, where $z \in \{a,\mu,\kappa,\gamma,\delta\}$, in the synchronous product can be simulated by the modality EF in the asynchronous product in addition to some formula that expresses (1) each asynchronous component can no longer execute z and (2) we have not moved too far down in the tree than just along one z-labeled transition.

▶ **Theorem 10.** *Let f be an elementary function. Then there is no algorithm that decides*
INPUT: Transition systems $\mathcal{T}_1,\ldots,\mathcal{T}_d$, a state $\overline{s}$ of $\bigotimes_{i=1}^d \mathcal{T}_i$ and an EF formula φ.
QUESTION: $(\overline{s}, \bigotimes_{i=1}^d \mathcal{T}_i) \models \varphi$?
in time $(\sum_{i=1}^d |\mathcal{T}_i|)^{O(1)} \cdot f(|\varphi|)$ unless $\mathsf{P} = \mathsf{NP}$.

7 Conclusion

In this paper we considered the fixed-parameter tractability of model checking modal logic and EF logic on concurrent systems that are modeled as the asynchronous or synchronous product of finite systems when the size of the input formula is the parameter. We showed that although these model checking problems are often fixed-parameter tractable one cannot hope for any FPT algorithm that runs elementary in the size of formula. It turned out that for model checking modal logic the mode of synchronization plays a role: for the asynchronous product we had to assume $\mathsf{FPT} \neq \mathsf{AWT}[*]$, whereas we were able to weaken our assumption for the synchronous product to $\mathsf{P} \neq \mathsf{NP}$. Let us conclude with some questions that we would like to answer in the full version of this paper. In analogy to [9] it would be interesting to study the question if even weaker complexity theoretic assumptions such as $\mathsf{P} \neq \mathsf{PSPACE}$ can be assumed. Moreover, we remark that some of our lower bound proofs also hold even when $d + |\varphi|$ (instead of $|\varphi|$) is the parameter; it seems worth investigating when this strengthening is possible. Too, the question arises what (elementary) bounds one can prove for transition systems of bounded degree.

References

1 Antonis Achilleos, Michael Lampis, and Valia Mitsou. Parameterized Modal Satisfiability. In *ICALP (2)*, volume 6199 of *Lecture Notes in Computer Science*, pages 369–380. Springer, 2010.

2 Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

3 Egon Börger, Erich Grädel, and Yuri Gurevich. *The classical decision problem*. Universitext. Springer-Verlag, Berlin, 2001.

4 Yijia Chen, Jörg Flum, and Martin Grohe. Bounded nondeterminism and alternation in parameterized complexity theory. In *IEEE Conference on Computational Complexity*, pages 13–29. IEEE Computer Society, 2003.

5 E. M. Clarke and E. A. Emerson. *Model Checking*. MIT Press, 1999.

6 Anuj Dawar, Martin Grohe, Stephan Kreutzer, and Nicole Schweikardt. Model Theory Makes Formulas Large. In *Proc. of ICALP*, volume 4596 of *Lecture Notes in Computer Science*. Springer, 2007.

7 Stéphane Demri, François Laroussinie, and Ph. Schnoebelen. A parametric analysis of the state-explosion problem in model checking. *J. Comput. Syst. Sci.*, 72(4):547–575, 2006.

8 Jörg Flum and Martin Grohe. *Parametrized Complexity Theory*. Springer, 2006.

9 Markus Frick and Martin Grohe. The complexity of first-order and monadic second-order logic revisited. *Ann. Pure Appl. Logic*, 130(1-3):3–31, 2004.

10 Stefan Göller, Jean Christoph Jung, and Markus Lohrey. The Complexity of Decomposing Modal and First-Order Theories. In *LICS*, pages 325–334. IEEE, 2012.

11 Stefan Göller and Anthony Widjaja Lin. Concurrency Makes Simple Theories Hard. In *STACS*, volume 14 of *LIPIcs*, pages 148–159. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

12 Orna Lichtenstein and Amir Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *POPL*, pages 97–107. ACM Press, 1985.

13 M. Praveen. *Parametrized Complexity of some Problems in Concurrency and Verification*. PhD thesis, Homi Bhabha National Institute, 2011.

14 Alexander Rabinovich. On compositionality and its limitations. *ACM Trans. Comput. Log.*, 8(1), 2007.

15 A. Prasad Sistla and Edmund M. Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3), 1985.

A Proof of Lemma 4

Proof. One easily proves via induction on n that $n^2 + 1 \leq 2^n$ for each $n \geq 5$. We prove the lemma by induction on $\ell \geq 3$. For the induction base $\ell = 3$ we have

$$\mathsf{size}(4) = \mathsf{Tower}(3) \cdot \mathsf{size}(3) + 1 = \mathsf{Tower}(3) \cdot (\mathsf{Tower}(2) \cdot \mathsf{size}(2) + 1) + 1$$

$$\overset{\text{Figure 1}}{=} 2^{16} \cdot (16 \cdot 21 + 1) + 1 = 2^{16} \cdot 337 < (2^{16})^2 = \mathsf{Tower}(3)^2.$$

For the induction step, we have for each $\ell \geq 3$

$$\begin{aligned}
\mathsf{size}(\ell+1) &= \mathsf{Tower}(\ell) \cdot \mathsf{size}(\ell) + 1 \\
&\overset{\text{IH}}{\leq} \mathsf{Tower}(\ell) \cdot \mathsf{Tower}(\ell-1)^2 + 1 \\
&\leq \mathsf{Tower}(\ell) \cdot (\mathsf{Tower}(\ell-1)^2 + 1) \\
&\overset{\mathsf{Tower}(\ell-1) \geq 5}{\leq} \mathsf{Tower}(\ell) \cdot 2^{\mathsf{Tower}(\ell-1)} \\
&= \mathsf{Tower}(\ell)^2.
\end{aligned}$$

◀

B Proof of Lemma 6

Proof. We define the formulas by induction on ℓ. For the induction base $\ell = 0$ we put:

(1) $\mathsf{eq}_0^\times \overset{\text{def}}{=} \langle a_0 \rangle ((\alpha, b_0) \leftrightarrow (\beta, b_0))$.

(2) $\mathsf{less}_0^\times \overset{\text{def}}{=} \langle a_0 \rangle (\neg(\alpha, b_0) \wedge (\beta, b_0))$.

(3) $\mathsf{succ}_0^\times \overset{\text{def}}{=} \mathsf{less}_0^\times$.

For the induction step, we define:

(1) $\mathsf{eq}_{\ell+1}^\times \overset{\text{def}}{=} [a_{\ell+1}] \left(\mathsf{eq}_\ell^\times \longrightarrow ((\alpha, b_\ell) \leftrightarrow (\beta, b_\ell))\right)$.

(2) $\mathsf{less}_{\ell+1}^\times \overset{\text{def}}{=} \langle a_{\ell+1} \rangle \varphi_{\ell+1}^\times$, where

$$\varphi_{\ell+1}^\times \overset{\text{def}}{=} (\mathsf{eq}_\ell^\times \wedge \neg(\alpha, b_\ell) \wedge (\beta, b_\ell) \wedge [\Rightarrow](\mathsf{eq}_\ell^\times \rightarrow ((\alpha, b_\ell) \leftrightarrow (\beta, b_\ell))).$$

(3) $\mathsf{succ}_{\ell+1}^\times \overset{\text{def}}{=} \langle a_{\ell+1} \rangle (\varphi_{\ell+1}^\times \wedge [\Leftarrow](\alpha, b_\ell) \wedge [\Leftarrow] \neg(\beta, b_\ell))$.

◀