A Parametric Analysis of the State-Explosion Problem in Model Checking ⋆

S. Demri, F. Laroussinie and Ph. Schnoebelen [1]

Lab. Spécification & Vérification,
CNRS & École Normale Supérieure de Cachan
61, av. Pdt. Wilson, 94235 Cachan Cedex France.

Abstract

In model checking, the state-explosion problem occurs when one checks a *non-flat system*, i.e., a system implicitly described as a synchronized product of elementary subsystems. In this paper, we investigate the complexity of a wide variety of model-checking problems for non-flat systems under the light of *parameterized complexity*, taking the number of synchronized components as a parameter. We provide precise complexity measures (in the parameterized sense) for most of the problems we investigate, and evidence that the results are robust.

1 Introduction

Model checking, i.e., the automated verification that (the formal model of) a system satisfies some formal behavioral property, has proved to be a revolutionary advance for the correctness of critical systems, see, e.g., [13,2].

Investigating the computational complexity of model checking started with [51], and today the complexity of the main model-checking problems is known [2], see [50] for a survey. This led to the understanding that, in practice, the source of intractability is the size of the model and not the size of the property to be

⋆ Extended version of [14].

Email addresses: `demri@lsv.ens-cachan.fr` (S. Demri), `fl@lsv.ens-cachan.fr` (F. Laroussinie), `phs@lsv.ens-cachan.fr` (Ph. Schnoebelen).

[1] Contact author.

[2] There are exceptions: for example, the exact complexity of model checking for the branching μ-calculus [21], or model checking over deterministic structures [15,43], are still open.

checked. LTL model checking provides a typical example: while the problem is PSPACE-complete [51], it was observed in [42] that checking whether $S \models \phi$ can be done in time $O(|S| \times 2^{|\phi|})$. In practice ϕ is small and S is huge, so that "model checking is in linear time", as is often stated.

State explosion. In practice, the main obstacle to model checking is the *state-explosion problem*, i.e., the fact that the model S is described implicitly, as a synchronized product of several components (with perhaps the addition of boolean variables, clocks, etc.), so that $|S|$ is usually exponentially larger than the size of its implicit description. For example, if S is given as a synchronized product $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ of elementary components, the input of the model-checking problem has size $n = \sum_{i=1}^{k} |\mathcal{A}_i|$ while S has size $O\Big(\prod_{i=1}^{k} |\mathcal{A}_i|\Big)$, that is $O(n^k)$, or $2^{O(n)}$ when k is not fixed.

From a theoretical viewpoint, the state-explosion problem seems inescapable in the classical worst-case complexity paradigm (see also the complexity of other problems with succinct representations in [54,23]). Indeed, studies covering all the main model-checking problems and the most common ways of combining components have repeatedly shown that model-checking problems are exponentially harder when S is given implicitly [22,31,34,39,46,47,41,52].

A parametric analysis. The state-explosion problem can be investigated more finely through *parameterized complexity*, a theoretical framework developed by Downey and Fellows for studying problems where complexity depends differently on the size n of the input and on some other parameter k that varies less (in some sense) [17,19,18].

Any of the main model-checking problems where the input S is given implicitly as a sequence $\mathcal{A}_1, \ldots, \mathcal{A}_k$ of components can be solved in polynomial-time *for every fixed value of* k, e.g., in time $O(n^k)$. That is, for every fixed k, the problem is polynomial-time. However, Downey and Fellows consider $O(n^k)$ as intractable for parameterized problems since the exponent k of n is not bounded, while problems with algorithms running in time $f(k) \times n^c$ for some function f and constant c are considered easier (see [17,19,18] for convincing arguments) and are said to be *fixed-parameter tractable* or, shortly, FPT.

Parameterized complexity adheres to the "worst-case complexity" viewpoint but it leads to finer analysis. This can be illustrated on some graph-theoretical problems: among the NP-complete problems with a natural algorithm running in $O(n^k)$, many admit another algorithm in some $f(k) \times n^c$ (e.g., the existence in a graph of a cycle of size k, or the existence of a vertex cover of size k) while many others seem not to have any such solution (e.g., the existence of a clique of size k). This difference between the two kinds of problems may have a visible impact when comparing the efficiency of the available algorithmic methods, but this cannot be explained within the classical complexity paradigm where the two kinds of problems are deemed “equivalent”.

Our contribution. In this paper, we apply the parameterized complexity viewpoint to model-checking problems where the input is a synchronized product of k components, k being the parameter. We investigate model-checking problems ranging from reachability questions to temporal model checking for several temporal logics, to equivalence checking for several behavioral equivalences.

We provide precise complexity measures (in the parameterized sense) for most of the problems we investigate, and informative lower and upper bounds for the remaining ones. We show how the results are generally robust, i.e., insensitive to slight modifications (e.g., size of the synchronization alphabet) or restrictions (e.g., to deterministic systems).

Sadly, all the considered problems are shown intractable even in the parameterized viewpoint (but they may reach different levels of intractability). See the summary of results on page 26. This shows that these problems (very probably) do not admit solutions running in time $f(k) \times n^c$ for some f and c, and strengthens the known results about the computational complexity of the state-explosion problem.

We introduce, as a useful general tool, parameterized problems for Alternating Turing machines and relate them to Downey and Fellows' hierarchy. Finally, we enrich the known catalog of parameterized problems with problems from an important application field. While mainly aimed at model checking, our study is also interesting for the field of parameterized complexity itself. For example, we are able to sharpen the characterization of the complexity of FAI-II and FAI-III (from [17, p. 470]) that are basic parameterized problems from automata theory (see section 5).

Related work. For the most part, model-checking of synchronized products of systems has been studied within the classical computational complexity paradigm [22,31,34,39,46,47,41,52]. These works consider model checking of temporal logics and checking of behavioral equivalences.

In the parameterized complexity framework, works considering model-checking problems depart from our investigation on either one of the following two main points: the model is not given as a combination of k systems, k being the parameter, or the property to be checked is not given as a temporal logic formula, or in terms of behavioral equivalences:

- [3,55,8] investigate reachability problems on systems of k synchronized automata, where k is a parameter. Additional parameters, such as alphabet size and number of states, are used in [55]. Problems complete for the classes $\mathsf{W}[1]$, $\mathsf{W}[2]$ and $\mathsf{W}[\mathsf{P}]$ are investigated in [55,8]. Compared to our study, these works mainly consider deterministic automata and are concerned with automata-theoretic (or language-theoretic) questions rather than verification and model-checking questions.
- Some works consider model-checking problems for (fragments of) first-order logic where the parameter is the size of the *property to be checked* (or is derived from it) and where the model is given explicitly: this has no relation to the state-explosion problem and trivially leads to tractability in the parameterized sense for "temporal" logics. In [45], the evaluation problem over conjunctive queries is shown $\mathsf{W}[1]$-complete when the size of the query is the parameter. The parameterized complexity of this problem over other classes of queries (positive, first-order) is characterized leading to problems $\mathsf{W}[\mathsf{P}]$-hard and $\mathsf{W}[\mathsf{SAT}]$-hard. In the work [30] also inspired by database theory, the above $\mathsf{W}[1]$-hardness result is refined by proving that the evaluation problem of conjunctive queries of bounded tree-width becomes FPT.
- In a series of papers Grohe and Flum consider model-checking problems for first-order formulae over finite structures where the parameter is the size of the formula, aiming at characterizations of parameterized complexity problems in terms of first-order model-checking problems, for instance by controlling the alternations of quantifiers, see, e.g., [24,26]. The properties we investigate, such as reachability, are out of the scope of first-order logic. It is worth noting that in [26], characterizations of classes from the W-hierarchy, the A-hierarchy and the AW-hierarchy, are shown leading to alternative definitions of original classes introduced by Downey and Fellows [17].
- Machine-based characterization of parameterized complexity classes has probably started with the $\mathsf{W}[1]$-completeness of SHORT NDTM COMPUTATION, see, e.g., [17]. This result has been refined in [8] where $\mathsf{W}[2]$ and $\mathsf{W}[\mathsf{P}]$ are characterized by parameterized problems on Turing machines either by considering multiple tapes machines or by taking as parameter the number of nondeterministic steps. In [10,11], parameterized problems on standard random access machines are introduced and shown complete for classes such as $\mathsf{W}[1]$, $\mathsf{W}[2]$, $\mathsf{W}[\mathsf{P}]$, $\mathsf{AW}[\mathsf{P}]$ or $\mathsf{AW}[*]$. Even if our characterization of the classes XP and $\mathsf{AW}[1]$ in terms of problems on alternating Turing machines is a by-product of our investigations on parameterized model-checking problems, it nevertheless belongs to this trend that consists in characterizing parameterized complexity classes in terms of machines models.

Plan of the paper. Sections 2 and 3 recall the basic definitions about parameterized complexity and synchronized products of systems. We investigate reachability problems in sections 4 and 5, temporal logic problems in section 6, and behavioral equivalence problems in section 7. As a rule, proofs omitted from the main text can be found in the appendix.

2 Parameterized complexity

We follow [17]. A *parameterized language* P is a set of pairs $\langle x, k\rangle$ where x is a word over some finite alphabet and k, the *parameter*, is an integer. The problem associated with P is to decide whether $\langle x, k\rangle \in P$ for arbitrary $\langle x, k\rangle$.

A parameterized problem P is *(strongly uniformly) fixed-parameter tractable*, shortly "FPT", $\overset{\text{def}}{\Leftrightarrow}$ there exist a recursive function $f : \mathbb{N} \mapsto \mathbb{N}$ and a constant $c \in \mathbb{N}$ such that the question $\langle x, k\rangle \in P$ can be solved in time $f(k) \times |x|^c$ (see, e.g., [17, Chapter 2]).

A parameterized problem P is *fixed-parameter m-reducible* (fp-reducible) to the parameterized problem P' (in symbols $P \leq^{\text{fp}}_{\text{m}} P'$) $\overset{\text{def}}{\Leftrightarrow}$ there exist recursive total functions $f_1 : k \mapsto k'$, $f_2 : k \mapsto k''$, $f_3 : \langle x, k\rangle \mapsto x'$ and a constant $c \in \mathbb{N}$ such that $\langle x, k\rangle \mapsto x'$ is computable in time $k''|x|^c$ and $\langle x, k\rangle \in P$ iff $\langle x', k'\rangle \in P'$. Clearly, when $P \leq^{\text{fp}}_{\text{m}} P'$ and P' is FPT, then also P is FPT because in the definition of fp-reduction k' only depends on k (not on the input x) and f_3 can be viewed as an FPT function. P and P' are *fixed-parameter equivalent* (fp-equivalent) $\overset{\text{def}}{\Leftrightarrow}$ $P \leq^{\text{fp}}_{\text{m}} P' \leq^{\text{fp}}_{\text{m}} P$.

Parameterized complexity comes with an array of elaborate techniques to devise fp-feasible algorithms, and another set of techniques to show that a problem is (very probably[3]) not FPT.

2.1 Downey and Fellows's hierarchies

Downey and Fellows introduced the following hierarchy of classes of parameterized problems [17]:

$$\mathsf{FPT} \subseteq \mathsf{W}[1] \subseteq \mathsf{W}[2] \subseteq \cdots \subseteq \mathsf{W}[\mathsf{SAT}] \subseteq \mathsf{AW}[1] \subseteq \mathsf{AW}[\mathsf{SAT}] \subseteq \mathsf{AW}[\mathsf{P}] \subseteq \mathsf{XP},$$

[3] Since proving that a PSPACE-complete problem is not FPT entails $\mathsf{P} \neq \mathsf{PSPACE}$, most techniques for proving hardness of parameterized problems just show hardness in a class of problems that are conjectured not FPT.

where it is known that the inclusion between FPT, the class of FPT problems, and XP is strict. All these classes are closed under fp-reductions. W[1] is usually considered as the parameterized analogue of NP (from classical complexity theory) and a W[1]-hard problem is seen as intractable. Recent developments [16] indicate that the newly introduced class MINI[1] (between FPT and W[1]) is the source of untractability. XP contains all problems that can be solved in time $O(n^{g(k)})$ for some function g and is considered as the parameterized analogue of EXPTIME. It should be stressed that the above analogies are only useful heuristics: there is no known formal correspondence between standard complexity classes (NP, PSPACE, EXPTIME, ...) and parameterized complexity classes (W[1], AW[P], XP, ...) [4].

We do not recall the formal definitions of these classes since they are not required for understanding our results. It is enough to admit that W[1] is intractable, and to understand the parameterized problems dealing with short or compact computations we introduce in the next subsection. Most of the parameterized model-checking problems we consider in this paper are easily seen to be in XP.

2.2 Short and compact TM computations

Not surprisingly, some fundamental parameterized problems consider Turing machines (shortly, "TMs"): SHORT COMPUTATION (resp. COMPACT COMPUTATION) is the parameterized problem where one is given a TM M and where it is asked whether M accepts in at most k steps (resp. using at most k work tape squares). These are the parameterized versions of the time and space bounds from classical complexity theory.

We consider TMs with just one initially blank work-tape (an input word can be encoded in the control states of the TM). One obtains different problems by considering deterministic (DTM), non-deterministic (NDTM), or alternating (ATM) machines. For instance, SHORT DTM COMPUTATION is defined as follows:

Instance: a single-tape deterministic Turing machine M and a positive integer k (in unary);
Parameter: k;
Question: Does the computation of M on the empty string input reach an accepting state in at most k steps?

The parameter is k. The other problems are defined analogously.

[4] But see the recent work [25].

Short DTM Computation is FPT while Short NDTM Computation is $\mathsf{W}[1]$-complete [17]. Compact Computation is more complex and reaches high levels in the W-hierarchy: Compact DTM Computation is already $\mathsf{AW}[\mathsf{SAT}]$-hard [10]. Some of the parameterized problems on Turing machines do not yet admit a full characterization in terms of parameterized complexity classes (see, e.g., [6,9,8,11]) although the parameterized complexity classes $\mathsf{W}[1]$, $\mathsf{W}[2]$ and $\mathsf{W}[\mathsf{P}]$ are characterized by parameterized problems on TMs in [8]. For instance, Short NDTM Computation with multiple tapes is $\mathsf{W}[2]$-complete [8].

Remark 2.1 *More precise measures are still lacking and [17, Chapter 14] recalls that it is not known whether* Compact DTM Computation *and* Compact NDTM Computation *are fp-equivalent (it is not known whether a parameterized version of Savitch's theorem holds). A related question is that it is not known whether co*Compact NDTM Computation *(the complement parameterized problem of* Compact NDTM Computation *defined in the obvious way) and* Compact NDTM Computation *are fp-equivalent. Indeed, Lemma 2 of Immerman's proof [33] does not provide an fp-reduction from* Compact NDTM Computation *to co*Compact NDTM Computation *because the alphabet size has to be taken into account in an essential way.*

[17] does not consider parameterized problems with ATMs, but such problems proved very useful in our study. Our first results show how they correspond to existing levels of the W-hierarchy:

Theorem 2.2 Short ATM Computation *is* $\mathsf{AW}[1]$*-complete.*

Theorem 2.2 is a corollary of the two reductions given in Lemmas 2.3 and 2.5 proving equivalence with Parameterized-QBFSAT$_t$, a problem shown $\mathsf{AW}[1]$-complete in [17, Chapter 14]. An instance of Parameterized-QBFSAT$_t$ is a quantified boolean formula $\Psi = \exists^{=k_1} X_1 \forall^{=k_2} X_2 \ldots \forall^{=k_{2p}} X_{2p} \Phi$ where Φ, a positive boolean combination of literals, has at most t alternations between conjunctions and disjunctions. The literals use variables in $X = X_1 \cup \cdots \cup X_{2p}$ and the quantifications "$\exists^{=k_i} X_i$" and "$\forall^{=k_i} X_i$" are relativized to valuations of X_i where exactly k_i variables are set to true. The parameter k is $k_1 + \cdots + k_{2p}$.

Lemma 2.3 *For every $t \geq 0$,*
Parameterized-QBFSAT$_t$ $\leq^{\text{fp}}_{\text{m}}$ Short ATM Computation.

PROOF. With an instance Ψ of Parameterized-QBFSAT$_t$, we associate an ATM M_Ψ that picks $k_1 + \cdots + k_{2p}$ variables in $X_1 \cup \cdots \cup X_{2p}$ and checks that Φ evaluates to true under the corresponding valuation. The structure of Φ is reflected in the transition table of M_Ψ, and we use universal states to encode both the universal quantifications "$\forall^{=k_{2i}} \ldots$" and the conjunctions in

Φ. The machine M_Ψ can be made to answer in $O(k+t)$ steps, which gives us an fp-reduction since t is a constant. $\Box$

In order to show that SHORT ATM COMPUTATION is in $\mathsf{AW}[1]$, we introduce below the parameterized problem STRICT SHORT ATM COMPUTATION shown to be fp-equivalent to SHORT ATM COMPUTATION:

Instance: a single-tape ATM $M = \langle Q^\exists \cup Q^\forall, \Sigma, \delta, q_0, q_F\rangle$ such that $q_0, q_F \in Q^\forall$ and M has clean alternation (it moves from existential states to universal states and vice versa), and a positive integer k (in unary);
Parameter: k;
Question: Does M on the empty string input has an accepting run using less than k steps?

As usual, $\{Q^\exists, Q^\forall\}$ forms a partition of the set of states and $Q^\exists$ denotes the set of existential states, and δ is the transition relation with $\delta \subseteq Q \times \Sigma \times Q \times \Sigma \times \{L, R, -\}$.

First, we show that STRICT SHORT ATM COMPUTATION is indeed equivalent to SHORT ATM COMPUTATION.

Lemma 2.4
SHORT ATM COMPUTATION $\leq_{\mathrm{m}}^{\mathrm{fp}}$ STRICT SHORT ATM COMPUTATION.

PROOF. (Idea) Let $M = \langle Q^\exists \cup Q^\forall, \Sigma, \delta, q_0, q_F\rangle$ be an ATM and k be a positive integer. One can build a strict ATM $M' = \langle Q'^\exists \cup Q'^\forall, \Sigma', \delta', q_0', q_F'\rangle$ such that M on the empty string input has an accepting run using less than k steps iff M' on the empty string input has an accepting run using less than $2 \times k + 2$ steps. The idea of the reduction is simply to add intermediate states when the alternation is not strict and to consider two counters, one to count the number of steps in the original M and another one to count the number of steps in M'. Details are omitted herein since there is no technical difficulty. $\Box$

Lemma 2.5
STRICT SHORT ATM COMPUTATION $\leq_{\mathrm{m}}^{\mathrm{fp}}$ PARAMETERIZED-QBFSAT$_t$ *for some* $t \geq 0$.

PROOF. With an ATM M and an odd $k = 2p+1$, we associate a formula Ψ_M that is true iff M accepts in k moves. The variables in Ψ are all $x[i,t,l]$ and mean "l is the ith symbol in the instantaneous description (i.d.) of M at step t". i and t range over $0, \ldots, k$, while l is any tape symbol or pair $\langle\text{symbol}, \text{control state}\rangle$ of M. Assuming M starts with a universal move, Ψ_M has the general form $\exists^{=k+1} X_0 \forall^{=k+1} X_1 \ldots \forall^{=k+1} X_k \Phi$ where

$X_t = \{x[i,t,l] \mid i,l\ldots\}$ and Φ checks that the chosen valuations correspond to a run, i.e., has the form

$$\overbrace{\Big(\bigwedge_{t=0}^{p} \Phi_{\mathrm{seq}}(X_{2t}, X_{2t+1})\Big)}^{\Phi_\forall} \Rightarrow \Big(\Phi_{\mathrm{init}}(X_0) \wedge \Phi_{\mathrm{accept}}(X_k) \wedge \overbrace{\bigwedge_{t=1}^{p} \Phi_{\mathrm{seq}}(X_{2t-1}, X_{2t})\Big)}^{\Phi_\exists}$$

where $\Phi_{\mathrm{seq}}(X, X')$ checks that (the valuations of) X and X' describe valid i.d.'s in valid succession. The different treatment between $\Phi_\forall$ and $\Phi_\exists$ reflects the fact that valid successions of existential states are only performed when valid successions of universal states are done. Moreover, this way of grouping the $\Phi_{\mathrm{seq}}(X_l, X_{l+1})$'s allows us to bound the number of and-or alternations. The formula $\Phi_{\mathrm{init}}(X)$ [resp. $\Phi_{\mathrm{accept}}(X)$] expresses that X describes an initial i.d. [resp. an accepting i.d.].

Finally, we observe that Φ is equivalent to a positive boolean combination of literals with 5 and-or alternations and therefore we obtain an instance of Parameterized-QBFSAT$_5$ with $k' = (k+1)^2$ and size n' in $O(k^2 n^3)$. $\square$

Theorem 2.6 Compact ATM Computation *is* XP-*complete.*

Theorem 2.6 is a corollary of the two reductions given in Lemmas 2.7 and 2.8.

We show fp-equivalence with Pebble Game, shown XP-complete in [17, Theorem 15.5]. An instance of Pebble Game is a set N of nodes, a starting position $S = \{s_1, \ldots, s_k\} \subseteq N$ of k pebbles on k nodes, a terminal node $T \in N$ and a set of possible moves $R \subseteq N \times N \times N$. Players I and II play in turn, moving pebbles and trying to reach T. A move $\langle x, y, z\rangle \in R$ means that any player can move a pebble from x to z if y is occupied (the pebble jumps over y) and z is free. The problem is to determine whether player I has a winning strategy. The parameter is $k = |S|$.

Lemma 2.7 Compact ATM Computation $\leq^{\mathrm{fp}}_{\mathrm{m}}$ Pebble Game.

PROOF. Immediate from [37]. Indeed, [37, Theorem 3.1] shows that Pebble Game is EXPTIME-hard by reducing space-bounded ATMs. Their reduction can be turned into an fp-reduction where an ATM of size n running in space k gives rise to a pebble game instance where k' is $k+1$, and where n' is bounded by a polynomial of n. $\square$

Lemma 2.8 Pebble Game $\leq^{\mathrm{fp}}_{\mathrm{m}}$ Compact ATM Computation.

PROOF. Given an instance $I = \langle N, S, T, R\rangle$ with $|S| = k$, one constructs an ATM M_I that emulates the game and accepts iff player I wins. The alphabet of M_I is N and k work-tape squares are sufficient to store the current configuration at any time in the game. Moves by player I are emulated with existential states, moves by player II use universal states. Information about R (the set of rules) and S is stored in the transition table of M_I. This gives an fp-reduction since $|M_I|$ is in $O(|I|)$ and $k' = k$. $\square$

3 Synchronized transition systems

3.1 Models

A *labeled transition system* (LTS) over some alphabet $\Sigma = \{a, b, \dots\}$ is a tuple $\mathcal{A} = \langle Q, \Sigma, \rightarrow\rangle$ where $Q = \{s, t, \dots\}$ is the set of states and $\rightarrow \subseteq Q \times \Sigma \times Q$ is the set of transitions. We assume the standard notation $s \xrightarrow{a} t$, $s \xrightarrow{w} t$ ($w \in \Sigma^*$), $s \xrightarrow{*} t$, $s \xrightarrow{+} t$, etc. The size of a finite LTS $\mathcal{A}$ is $|\mathcal{A}| \stackrel{\text{def}}{=} |Q| + |\Sigma| + |\rightarrow|$. *Non-flat systems* are products of (flat) component LTSs. Assuming $\mathcal{A}_i = \langle Q_i, \Sigma_i, \rightarrow_i\rangle$ for $i = 1, \dots, k$, the product $\mathcal{A}_1 \times \dots \times \mathcal{A}_k$ denotes a LTS $\langle Q, \Sigma, \rightarrow\rangle$ where $Q \stackrel{\text{def}}{=} \prod_{i=1}^{k} Q_i$, $\Sigma \stackrel{\text{def}}{=} \bigcup_{i=1}^{k} \Sigma_i$ and where $\rightarrow \subseteq Q \times \Sigma \times Q$ depends on the synchronization protocol one considers: strong or binary synchronization. A state $\bar{s} = \langle s_1, \dots, s_k\rangle$ of $\mathcal{A}_1 \times \dots \times \mathcal{A}_k$ is also called a configuration and it corresponds to the state in the composed system $\mathcal{A}_1 \times \dots \times \mathcal{A}_k$ in which for every $i \in \{1, \dots, k\}$, $\mathcal{A}_i$ is in the state s_i.

In *strong synchronization*, the components synchronize on common actions and move in lockstep fashion: $\langle s_1, \dots, s_k\rangle \xrightarrow{a}_{\text{str}} \langle t_1, \dots, t_k\rangle$ iff $s_i \xrightarrow{a}_i t_i$ for all $i = 1, \dots, k$.

In *binary synchronization*, any two components synchronize while the rest wait: $\langle s_1, \dots, s_k\rangle \xrightarrow{a}_{\text{bin}} \langle t_1, \dots, t_k\rangle$ iff there exist i and j ($i \neq j$) s.t. $s_i \xrightarrow{a}_i t_i$ and $s_j \xrightarrow{a}_j t_j$ while $s_l = t_l$ for all $l \notin \{i, j\}$.

Hence a transition $\bar{s} \xrightarrow{a} \bar{t}$ in the composed system $\mathcal{A}_1 \times \dots \times \mathcal{A}_k$ corresponds to a set of transitions from the underlying subsystems, depending on the synchronization mode. In this paper, we consider *strong synchronization* as the natural model for non-flat systems and the notation $\mathcal{A}_1 \times \dots \times \mathcal{A}_k$ assumes strong synchronization when we do not explicitly say otherwise. As shown in Appendix B, adopting binary synchronization does not modify the complexity in an essential way.

[53] surveys the main behavioral equivalences (and preorders) used in the semantics of concurrent systems. We recall below the definition of bisimilarity and trace inclusion since they are used in Sections 6 and 7. Other relations can be found in [53].

Given two LTSs $\mathcal{A} = \langle Q, \Sigma, \rightarrow\rangle$ and $\mathcal{A}' = \langle Q', \Sigma', \rightarrow'\rangle$, a bisimulation is any relation $R \subseteq Q \times Q'$ satisfying the following transfer properties:

- for all qRq' and $q \xrightarrow{a} r$, there is $q' \xrightarrow{a}{}' r'$ such that rRr';
- for all qRq' and $q' \xrightarrow{a}{}' r'$, there is $q \xrightarrow{a} r$ such that rRr'.

The largest bisimulation is called bisimilarity and is denoted by $\sim$.

Given a LTS $\mathcal{A} = \langle Q, \Sigma, \rightarrow\rangle$ and $q \in Q$, a trace from q is a sequence $a_1 \ldots a_n \ldots$ (possibly infinite) such that there exists $q_0, q_1, \ldots, q_n \ldots \in Q$ with $q = q_0$ and $q_{i-1} \xrightarrow{a_i} q_i$ for every i. Given two LTSs $\mathcal{A} = \langle Q, \Sigma, \rightarrow\rangle$ and $\mathcal{A}' = \langle Q', \Sigma', \rightarrow'\rangle$, $q \in Q$ and $q' \in Q'$, we write $\mathcal{A}, q \subseteq_{\mathrm{tr}} \mathcal{A}', q'$ to denote that every trace from q is a trace from q'.

4 Reachability for non-flat systems

Reachability problems are the most fundamental problems in model checking. We define below four reachability problems.

Exact Reachability (Exact-Reach)
Instance: k LTSs $\mathcal{A}_1, \cdots, \mathcal{A}_k$, two configurations $\bar{s}$ and $\bar{t}$ of $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$.
Question: Does $\bar{s} \xrightarrow{*} \bar{t}$?

Local Reachability (Local-Reach)
Instance: k LTSs $\mathcal{A}_1, \cdots, \mathcal{A}_k$, sets $F_1, \ldots, F_k$ of states with $F_i \subseteq Q_i$, and a configuration $\bar{s}$ of $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$.
Question: Does $\bar{s} \xrightarrow{*} \bar{t}$ for some $\bar{t} \in \bar{F}$ where $\bar{F} \stackrel{\text{def}}{=} F_1 \times \cdots \times F_k$?

Repeated Reachability (Rep-Reach)
Instance: As in Local-Reach.
Question: Does $\bar{s} \xrightarrow{*} \bar{t} \xrightarrow{+} \bar{t}$ for some $\bar{t} \in \bar{F}$?

Fair Reachability (Fair-Reach)
Instance: k LTSs $\mathcal{A}_1, \cdots, \mathcal{A}_k$, sets $(F_i^j)_{i=1,\ldots,k}^{j=1,\ldots,p}$ with $F_i^j \subseteq Q_i$ for all i, j, and a configuration $\bar{s}$ of $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$. For all j we write $\bar{F}^j$ for $F_1^j \times \cdots \times F_k^j$.
Question: Does $\bar{s} \xrightarrow{*} \bar{t_1} \xrightarrow{*} \bar{t_2} \ldots \xrightarrow{*} \bar{t_p} \xrightarrow{+} \bar{t_1}$ for some $\langle \bar{t_1}, \ldots, \bar{t_p} \rangle \in \bar{F}^1 \times \cdots \times \bar{F}^p$?

The reachability criterion in the problem Rep-Reach corresponds to the acceptance condition in Büchi automata. Repeated reachability asks that the final state $\bar{t}$ should be accessible from itself with a non-zero number of transitions. The reachability criterion in the problem Fair-Reach introduces a fairness condition.

We are interested in the parameterized versions k-Exact-Reach, etc., where k is the parameter. The choice of such a parameter is quite natural since k varies less than $\Sigma_{i=1}^k |\mathcal{A}_i|$. It is well-known that the four non-flat reachability problems are equivalent in the classical sense (i.e., via logspace reductions) and are PSPACE-complete. These are folklore results for which some hints can be found in [31,22]. However, in the setting of parameterized complexity, reductions have to preserve more structure, so that Theorem 4.1 and other forthcoming results require some more care [5]. Additionally, we shall consider extra parameters and/or further restrictions on the LTSs which require some new constructions in proofs.

Theorem 4.1 *k-Exact-Reach, k-Local-Reach, k-Rep-Reach and k-Fair-Reach are fp-equivalent.*

Theorem 4.1 allows us to write k-$*$-Reach to denote any of the four problems, as we do below. For a proof, observe first that Exact-Reach is the restriction of Local-Reach where $|F| = 1$, and Rep-Reach is the restriction of Fair-Reach where $p = 1$. We prove below that k-Fair-Reach $\leq_{\mathrm{m}}^{\mathrm{fp}}$ k-Rep-Reach, refer to Appendix A for k-Rep-Reach $\leq_{\mathrm{m}}^{\mathrm{fp}}$ k-Exact-Reach, and omit the easy k-Local-Reach $\leq_{\mathrm{m}}^{\mathrm{fp}}$ k-Rep-Reach.

Lemma 4.2 k-Fair-Reach $\leq_{\mathrm{m}}^{\mathrm{fp}}$ k-Rep-Reach.

PROOF. (Sketch) Consider an instance $\mathcal{A}_1, \ldots, \mathcal{A}_k, (F_i^j)_{i=1,\ldots,k}^{j=1,\ldots,p}, \bar{s}$ of k-Fair-Reach and write $\bar{F}^j$ for $F_1^j \times \cdots \times F_k^j$. Assume w.l.o.g. that the $\mathcal{A}_i$s are over some common Σ.

[5] For simplicity, we consider that the size n of the input is simply $\sum_i |\mathcal{A}_i|$ since the extra inputs (designated states, etc.) are $O(n)$ anyway. There is one exception with Fair-Reach where n is considered to be $p \times \sum_i |\mathcal{A}_i|$, an innocuous approximation of $\sum_i |\mathcal{A}_i| + \sum_i \sum_j |F_i^j|$.

We reduce this to an instance $\mathcal{A}'_0, \mathcal{A}'_1, \cdots, \mathcal{A}'_k$ of k-Rep-Reach where $\mathcal{A}'_0$ is a special "controller" LTS and where $\mathcal{A}'_1, \ldots, \mathcal{A}'_k$ are obtained from the $\mathcal{A}_i$s. Let Σ' be $\Sigma \cup \{1, \ldots, p\}$. For any i, $\mathcal{A}'_i$ is obtained from $\mathcal{A}_i$ by adding all transitions $s \xrightarrow{j} s$ for $s \in F_i^j$ for every $j \in \{1, \ldots, p\}$. Hence, $\bar{s} \in \bar{F}^j$ iff $\bar{s} \xrightarrow{j} \bar{s}$ in $\mathcal{A}'_1 \times \cdots \times \mathcal{A}'_k$. The controller $\mathcal{A}'_0$ has a loop $0 \xrightarrow{\Sigma} 1 \xrightarrow{1} 2 \xrightarrow{2} \ldots p \xrightarrow{p} 0$ with local loops $i \xrightarrow{\Sigma} i$ for any $i = 1, \ldots, p$. ("$s \xrightarrow{\Sigma} t$" is short for "$s \xrightarrow{a} t$ for all $a \in \Sigma$"). We set $\bar{F}' = \{0\} \times \bar{F}^p$.

Clearly, there exist $\bar{t_1}, \ldots, \bar{t_p} \in \bar{F}^1, \ldots, \bar{F}^p$ with $\bar{s} \xrightarrow{*} \bar{t_1} \xrightarrow{*} \bar{t_2} \ldots \xrightarrow{*} \bar{t_p} \xrightarrow{+} \bar{t_1}$ in $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ iff there is a $\bar{t}' \in \bar{F}'$ s.t. $0, \bar{s} \xrightarrow{*} \bar{t}' \xrightarrow{+} \bar{t}'$ in $\mathcal{A}'_0 \times \mathcal{A}'_1 \times \cdots \times \mathcal{A}'_k$.

There remains to check that this classical construction is indeed an fp-reduction: for $i = 1, \ldots, k$, $|\mathcal{A}'_i|$ is in $O(p \times |\mathcal{A}_i|)$, $|\mathcal{A}'_0|$ is in $O(p \times |\Sigma|)$. Hence, with $n = \sum_i |\mathcal{A}_i| + \sum_i \sum_j |F_i^j|$ and $n' = \sum_i |\mathcal{A}'_i|$, we have $k' = k + 1$, $|\Sigma'| = p + |\Sigma|$, and n' in $O(n^2)$. $\square$

5 Parameterized complexity of non-flat reachability

5.1 Equivalence with Compact NDTM Computation

In this section we give two reductions (Lemmas 5.2 and 5.3) that allow the following characterization:

Theorem 5.1 k-$*$-Reach *is fp-equivalent to* Compact NDTM Computation.

Hence all the parameterized reachability problems are $\mathsf{AW[SAT]}$-hard.

Lemma 5.2 Compact NDTM Computation $\leq_{\mathrm{m}}^{\mathrm{fp}}$ k-Local-Reach.

PROOF. (Sketch) With an NDTM M and an integer k we associate a product $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k \times \mathcal{A}_{\mathrm{state}} \times \mathcal{A}_{\mathrm{head}}$ of $k+2$ LTSs that emulate the behavior of M on a k-bounded tape. An $\mathcal{A}_i$ stores the current content of the i-th tape square, $\mathcal{A}_{\mathrm{state}}$ stores the current control-state of M and $\mathcal{A}_{\mathrm{head}}$ stores the position of the TM head. These LTSs synchronize on labels of the form $\langle t, i \rangle$ that stand for "rule t of M is fired while head is in position i". Successful acceptance by M is directly encoded as a local reachability criterion. Altogether, we translate our instance to a k-Local-Reach instance with $k' = k+2$ and n' in $O(kn^2)$. Details are on the line of the proof of Theorem 6.6. $\square$

The proof of Lemma 5.2 is similar to either the proof of the PSPACE-hardness of the finite automaton intersection problem [28,38] (see also a generalization in [40]) or the proof of the PSPACE-hardness of REACHABLE DEADLOCK for a system of communicating processes [44, Theorem 19.10] (see also [29, Appendix AL6]). One can also prove in a similar way that a linearly bounded automaton of size n can be simulated by a 1-safe Petri net of size $O(n^2)$ [35]. Such a similarity does not prevent us from checking that we are in presence of an fp-reduction.

Lemma 5.3 *k-EXACT-REACH $\leq_m^{fp}$ COMPACT NDTM COMPUTATION.*

PROOF. (Sketch) An instance of k-EXACT-REACH of the form $\mathcal{A}_1, \ldots, \mathcal{A}_k$, $\bar{s}$, $\bar{t}$, is easily reduced to an instance of COMPACT NDTM COMPUTATION. The TM M emulates the behavior of the product $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ by writing the initial configuration $\bar{s}$ on its tape (one component per tape square, the tape alphabet contains all control states of the $\mathcal{A}_i$'s). Then M picks non-deterministically a synchronization letter a, updates all local states of the $\mathcal{A}_i$s by firing one of their a-transitions (M blocks if some local state has no a-transition), and repeats until the configuration $\bar{t}$ is reached. This yields an fp-reduction: $k' = k$ and n' is in $O(kn)$. □

Hence, k-EXACT-REACH is fp-equivalent to one of the most natural parameterized problems on Turing machines: COMPACT NDTM COMPUTATION.

5.2 Variants of non-flat reachability problems

In Theorem 4.1, we state that the four parameterized non-flat reachability problems are fp-equivalent, and in Theorem 5.1 we show that they are fp-equivalent to COMPACT NDTM COMPUTATION. This characterization is quite robust: It stays unchanged when we consider binary synchronization or when we restrict to a binary alphabet or to deterministic LTSs.

5.2.1 Binary synchronization.

We write k-$*$-REACH $_{\text{bin}}$ the variants of the k-$*$-REACH problems where the LTSs are combined using binary synchronization instead of strong synchronization. Similarly, we write k, Σ-$*$-REACH$_{\text{bin}}$ to denote the variants of the k-$*$-REACH $_{\text{bin}}$ problems with parameters k and $|\Sigma|$. By definition, there is an fp-reduction from k, Σ-$*$-REACH$_{\text{bin}}$ into k-$*$-REACH $_{\text{bin}}$ as it is the case whenever a parameter is added to a parameterized problem.

Theorem 5.4 *k-$*$-Reach$_{\text{bin}}$ and k, Σ-$*$-Reach$_{\text{bin}}$ are fp-equivalent to* Compact NDTM Computation.

The proof of Theorem 5.4 can be found in Appendix B.

5.2.2 Bounded size alphabet.

Obviously, k-$*$-Reach$_{|\Sigma|=2}$, the restriction of k-$*$-Reach to binary alphabet reduces to k, Σ-$*$-Reach, the variant of k-$*$-Reach with parameters $k, |\Sigma|$, which itself reduces to k-$*$-Reach. We show the following result:

Theorem 5.5 *k, Σ-$*$-Reach and k-$*$-Reach$_{|\Sigma|=2}$ are fp-equivalent to* Compact NDTM Computation.

In order to prove Theorem 5.5, first we reduce reachability properties over any LTS to reachability properties over a LTS using an alphabet $\Sigma' = \{0, 1\}$. This is done by the following construction. Note that a simpler construction is possible but the current one is used again in Section 7 where stronger properties are required. Let $\mathcal{A} = \langle Q, \Sigma, \to \rangle$ be a LTS over some $\Sigma = \{a_1, \ldots, a_m\}$ and $l = \lceil \log_2 m \rceil$. With each $a_i \in \Sigma$ we associate a bit-string w_{a_i} of length l in $\{0, 1\}^*$, representing the binary writing of i. Let $\widehat{\mathcal{A}} = \langle \widehat{Q}, \{0, 1\}, \to \rangle$ be the LTS defined as follows:

- $\widehat{Q} \stackrel{\text{def}}{=} \{\langle q, v \rangle : q \in Q,\ v \in \{0, 1\}^i,\ 0 \leq i \leq l\}$;
- for $i \in \{0, 1\}$, $\langle q, v \rangle \xrightarrow{i} \langle q', v' \rangle \stackrel{\text{def}}{\Leftrightarrow}$

(1) either $q = q'$ and $v' = vi$,
(2) or $i = 0$, $|v| = l$, $v' = \varepsilon$ and $q \xrightarrow{a_j} q'$ in $\mathcal{A}$ with $w_{a_j} = v$.

An illustration of the construction can be found in Fig. 1. Basically, with each state in Q we associate a binary tree of depth l in $\widehat{\mathcal{A}}$. By construction, we guarantee that $q \xrightarrow{a_i} q'$ in $\mathcal{A}$ iff $\langle q, w_{a_i} \rangle \xrightarrow{0} \langle q', \varepsilon \rangle$, where $\langle q, w_{a_i} \rangle$ is the leaf in the binary tree associated with q that is reached via the path w_{a_i}. $\langle q', \varepsilon \rangle$ is the root of the binary tree associated with q'.

Lemma 5.6 states that $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ and $\widehat{\mathcal{A}_1} \times \cdots \times \widehat{\mathcal{A}_k}$ are equivalent as far as reachability is concerned.

Lemma 5.6 *Let $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ be a product of LTSs over some Σ. Then, $\langle s_1, \ldots, s_k \rangle \xrightarrow{a_1 \ldots a_n} \langle t_1, \ldots, t_k \rangle$ in $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ iff $\langle \langle s_1, \varepsilon \rangle, \ldots, \langle s_k, \varepsilon \rangle \rangle \xrightarrow{w_{a_1} 0 \ldots w_{a_n} 0} \langle \langle t_1, \varepsilon \rangle, \ldots, \langle t_k, \varepsilon \rangle \rangle$ in $\widehat{\mathcal{A}_1} \times \cdots \times \widehat{\mathcal{A}_k}$.*

Consequently,

Lemma 5.7 *k-$*$-Reach $\leq^{\text{fp}}_{\text{m}}$ k-$*$-Reach$_{|\Sigma|=2}$.*

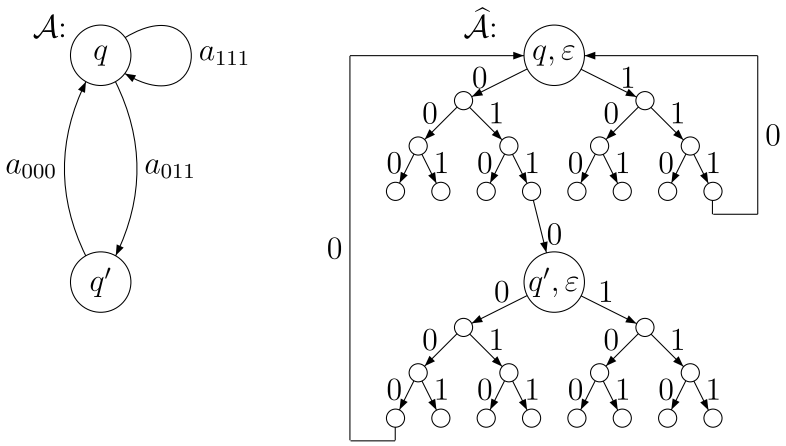

Fig. 1. $\mathcal{A}$ and $\widehat{\mathcal{A}}$: an example

PROOF. Let $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ be a product of LTSs over some Σ. Consider the product $\widehat{\mathcal{A}_1} \times \cdots \times \widehat{\mathcal{A}_k}$. By Lemma 5.6, $\langle s_1, \ldots, s_k \rangle \xrightarrow{a_1 \ldots a_n} \langle t_1, \ldots, t_k \rangle$ in $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ iff $\langle \langle s_1, \varepsilon \rangle, \ldots, \langle s_k, \varepsilon \rangle \rangle \xrightarrow{w_{a_1} 0 \ldots w_{a_n} 0} \langle \langle t_1, \varepsilon \rangle, \ldots, \langle t_k, \varepsilon \rangle \rangle$ in $\widehat{\mathcal{A}_1} \times \cdots \times \widehat{\mathcal{A}_k}$. Hence, any of the four non-flat reachability problems with parameter k is reducible to the analogous problem restricted to binary alphabets. These are fp-reductions since $k' = k$ and, for $i = 1, \ldots, k$, $|\widehat{\mathcal{A}_i}|$ is in $O(|\mathcal{A}_i| \times m|\Sigma|)$, i.e., n' is in $O(n^2)$. $\square$

Theorem 5.5 is then an immediate corollary.

5.2.3 Deterministic LTSs.

We write k-$*$-REACH $_{|\Sigma|=2,\mathrm{det}}$ [resp. k, Σ-$*$-REACH$_{\mathrm{det}}$] to denote the restriction of k-$*$-REACH $_{|\Sigma|=2}$ [resp. k, Σ-$*$-REACH] to deterministic LTSs. It is easy to show that all our reachability problems remain fp-equivalent to COMPACT NDTM COMPUTATION when we restrict them to products of Deterministic LTSs. Indeed, all the LTSs in the proof of Lemma 5.2 are deterministic even if the TM M is nondeterministic. Since the reductions in the proof of Theorem 4.1 preserve the determinism of the LTSs, we can state:

Theorem 5.8 *k-$*$-REACH$_{\mathrm{det}}$ is fp-equivalent to* COMPACT NDTM COMPUTATION.

The proof of Lemma 5.7 works for the deterministic case, therefore we obtain Theorem 5.9.

Theorem 5.9 *k, Σ-$*$-REACH$_{\mathrm{det}}$ and k-$*$-REACH$_{|\Sigma|=2,\mathrm{det}}$ are fp-equivalent to* COMPACT NDTM COMPUTATION.

As a corollary, we are able to improve the known lower bounds for Finite Automata Intersection (FAI) problems defined in [17, page 470] (W[t]-hard for all t ≥ 1, see, e.g., [17]). The general schema is the following:

Instance: k deterministic Σ-automata $\mathcal{A}_1, \dots, \mathcal{A}_k$, a configuration $\bar{s}$, an integer m (in unary) and a set $\bar{F} = F_1 \times \cdots \times F_k$ of final states, given as some $F_1, \dots, F_k$ with $F_i \subseteq Q_i$ for all i.
Question: Is there a $\bar{t} \in \bar{F}$ s.t. $\bar{s} \xrightarrow{*} \bar{t}$ in *at least* m steps?

The above problem with parameter k, Σ (resp. k) is referred to as FAI-III (resp. II) in [17, page 470].

By an easy manipulation, one can show

Theorem 5.10 FAI-II *is fp-equivalent to* k-LOCAL-REACH$_{\text{det}}$ *and* FAI-III *is fp-equivalent to* k, Σ-LOCAL-REACH$_{\text{det}}$.

PROOF. Omitted and obvious.

Corollary 5.11 FAI-II *and* FAI-III *are fp-equivalent to* COMPACT NDTM COMPUTATION *(hence* $\mathsf{AW[SAT]}$*-hard).*

By contrast, the above problem with parameter k, m is $\mathsf{W[1]}$-complete [8].

6 Parameterized complexity of non-flat temporal logic model checking

In this section, we investigate the parameterized complexity of temporal logic model-checking problems when the input is a synchronized product of LTSs (and a temporal formula!). We assume familiarity with the standard logics used in verification: LTL, CTL, Hennessy-Milner Logic (HML), the modal μ-calculus (see [20,13,4]).

In general, parameterized model checking will be harder than k-$*$-REACH, which should not come as a surprise. That is why, in this section, we focus our attention on showing that parameterized model-checking problems are higher than k-$*$-REACH in the hierarchy of parameterized complexity classes. Moreover, we shall also exhibit fragments of logics such that the corresponding parameterized model-checking problems are intractable in the parameterized sense although the logics cannot express reachability questions.

For modal logics, LTSs are the natural models, while for temporal logics like CTL [12] or LTL [27] the natural models are Kripke structures. Below we call *Kripke structure* (or shortly KS) a pair $\mathcal{M} = \langle \mathcal{A}, m \rangle$ of a finite LTS $\mathcal{A} = \langle Q, \Sigma, \rightarrow \rangle$ extended with a finite valuation $m \subseteq Q \times AP$ of its states (with AP a set of atomic propositions). The size $|\mathcal{M}|$ of $\mathcal{M} = \langle \mathcal{A}, m \rangle$ is $|\mathcal{A}| + |m|$.

The labels of the transitions of a KS do not appear in temporal formulae. They are only used for synchronization purposes: $\langle \mathcal{A}_1, m_1 \rangle \times \cdots \times \langle \mathcal{A}_k, m_k \rangle$ is the KS $\langle \mathcal{A}, m \rangle$ where $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ (implicitly assuming strong synchronization) and where m is a valuation built from $m_1, \ldots, m_k$. For the sake of simplicity, we assume in the sequel that m is the "sum" of $m_1, \ldots, m_k$, that is $\langle \langle q_1, \ldots, q_k \rangle, \mathrm{p} \rangle \in m$ as soon as $\langle q_i, \mathrm{p} \rangle \in m_i$ for some i. The synchronized structure $\langle \mathcal{A}, m \rangle$ is assumed to be defined on a unary alphabet.

The problems we consider have the following general form, where L is LTL, CTL, the modal μ-calculus, or some of their fragments.
Parameterized model checking for logic L ($\mathbf{MC}_L$)

Instance: Kripke structures $\mathcal{M}_1, \ldots, \mathcal{M}_k$, a configuration $\bar{s}$, an L-formula ϕ.
Parameter: k, $|\phi|$.
Question: Does $\mathcal{M}_1 \times \cdots \times \mathcal{M}_k, \bar{s} \models \phi$?

6.2 Linear time

We assume familiarity with Linear-Time Temporal Logic LTL, see, e.g., [20]. We recall below a few points. The LTL formulae are defined as follows

$$\phi ::= \mathrm{p} \ \mid \ \neg\phi \ \mid \ \phi_1 \vee \phi_2 \ \mid \ \phi_1 \wedge \phi_2 \ \mid \ \phi_1 \mathsf{U} \phi_2 \ \mid \ \mathsf{X}\phi,$$

where $\mathrm{p} \in AP$ (set of propositional variables), X is the next-time modality and U is the until modality ($\mathsf{F}\phi$ is an abbreviation for $\top \mathsf{U} \phi$). The LTL models are ω-sequences in $\mathcal{P}(AP)^\omega$. We recall that $\mathcal{M}, s \models \phi$ where $\mathcal{M}$ is a Kripke structure and ϕ is an LTL formula whenever there is an infinite path σ starting from s such that $\sigma \models \phi$. Hence, for linear-time logics (LTL and its fragments) we follow [51] and assume, for the sake of uniformity, that the question "$\mathcal{M}, s \models \phi$?" asks for the *existence* of a path from s that verifies ϕ, which is dual to the universal "all paths from s" formulation commonly used in applications.

LTL model checking for non-flat systems is PSPACE-complete (consequence of [51]). In our parameterized setting we have:

Theorem 6.1 *k, ϕ-MC$_{\text{LTL}}$ is fp-equivalent to* Compact NDTM Computation *and to* k-$*$-Reach *(and hence is* $\mathsf{AW[SAT]}$*-hard).*

PROOF. k-Exact-Reach reduces to k, ϕ-MC$_{\text{LTL}}$ since $\bar{s} \xrightarrow{*} \langle t_1, \ldots, t_k \rangle$ in some $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ iff $\langle \mathcal{A}_1, \{\langle t_1, \mathrm{p}_1\rangle\}\rangle \times \cdots \times \langle \mathcal{A}_k, \{\langle t_k, \mathrm{p}_k\rangle\}\rangle, \bar{s} \models \mathsf{F}(\mathrm{p}_1 \wedge \ldots \wedge \mathrm{p}_k)$. This provides an fp-reduction since $|\mathsf{F}(\mathrm{p}_1 \wedge \ldots \wedge \mathrm{p}_k)|$ is in $O(k \log k)$.

In the other direction, the question "does $\mathcal{M}_1 \times \cdots \times \mathcal{M}_k, \bar{s} \models \phi$?" reduces to a repeated reachability problem for $\mathcal{M}_1 \times \cdots \times \mathcal{M}_k \times \mathcal{B}_\phi$, where $\mathcal{B}_\phi$ is a Büchi automaton that accepts the paths satisfying ϕ [6]. There remains to check that this well known reduction is a reduction in the parameterized sense: since $|\mathcal{B}_\phi|$ is in $2^{O(|\phi|)}$, the reduction has $k' = k + 1$ and $n' \in O(2^{|\phi|} \times n)$, which is enough for fp-reducibility since k, $|\phi|$ are parameters of k, ϕ-MC$_{\text{LTL}}$. $\square$

LTL0 formulae are defined as LTL formulae only built with atomic propositions, the next-time modality "X", and disjunction "$\vee$" (no negation allowed). Observe that LTL0 cannot express reachability questions. However we have:

Theorem 6.2 *k, ϕ-MC$_{\text{LTL0}}$ is* $\mathsf{W[1]}$*-complete, even if we restrict to formulae using only one atomic proposition.*

PROOF. We prove that Short NDTM Computation, a standard $\mathsf{W[1]}$-complete problem, is fp-equivalent to our restricted model-checking problem.

Short NDTM Computation $\leq^{\text{fp}}_{\text{m}} k, \phi$-MC$_{\text{LTL0}}$:
We emulate an NDTM M via $k+2$ KSs as in the proof of Lemma 5.2. That M accepts in at most k steps can be stated as an LTL0 formula of the form $\bigvee_{t=0}^{k} \mathsf{X}^t \mathrm{p}_{\text{acc}}$ if the KSs are labeled so that p_{acc} marks the accepting states of M.

k, ϕ-MC$_{\text{LTL0}}$ $\leq^{\text{fp}}_{\text{m}}$ Short NDTM Computation:
Consider an instance "$\mathcal{M}_1 \times \cdots \times \mathcal{M}_k, \bar{s} \models \phi$?" and write ϕ under the form $\bigvee_{i=1}^{u} \mathsf{X}^{k_i} \mathrm{p}_{l_i}$ (with the distributivity law $\mathsf{X}(\phi_1 \vee \phi_2) \equiv \mathsf{X}\phi_1 \vee \mathsf{X}\phi_2$, this induces at most a quadratic blowup). Each $k_i \in \mathbb{N}$ is a number of nested next-time modalities and each l_i is a propositional variable index.

[6] Strictly speaking, $\mathcal{B}_\phi$ synchronizes with $M_1 \times \cdots \times M_k$ using a protocol different from what we used up to now: $\bar{s} \xrightarrow{a} \bar{t}$ and $q \xrightarrow{v} q'$ synchronize iff $m(\bar{s}) = v$. However, using the same techniques as in Appendix B, the k-$*$-Reach problems for this form of synchronized products can also be proved fp-equivalent to Compact NDTM Computation.

We build a NDTM M that emulates the product $\mathcal{M}_1 \times \cdots \times \mathcal{M}_k$ as in the proof of Lemma 5.3 but here M counts how many steps have been emulated so that it may accept if, for some i, p_{l_i} holds after k_i steps. Emulating one step requires k moves, so that M answers in at most $k' = k \times \mathbf{max}_i(k_i)$ steps. □

6.3 Branching time

6.3.1 CTL.

Computation Tree Logic CTL [12] is also a well-known temporal logic for model checking. Since we mainly state a hardness result, there is no real need to recall its definition here.

Theorem 6.3 Compact NDTM Computation $\leq^{\mathrm{fp}}_{\mathrm{m}}$ k, ϕ-MC$_{\mathrm{CTL}}$. *(Hence* k, ϕ-MC$_{\mathrm{CTL}}$ *is* AW[SAT]*-hard).*

PROOF. (Idea) CTL allows to state reachability questions. □

Remark 6.4 *Even though model checking non-flat systems is* PSPACE*-complete for CTL, for the moment, we do not have a more precise characterization for* k, ϕ-MC$_{\mathrm{CTL}}$*. Observe that, by definition, model checking CTL is closed under complementation (unlike the LTL case) so that a conjecture about being fp-equivalent to* Compact NDTM Computation *would have an impact on the open problems mentioned in Remark 2.1. For upper bounds, the problem is obviously in* XP *and we failed to refine this, partly because parameterized complexity does not offer many natural classes above* AW[SAT].

Model checking non-flat systems is already W[1]-complete for BT0, a weak fragment of CTL that cannot express reachability questions. The logic BT0 only allows EX (no other modality) and $\vee$ (neither negation nor conjunction):

Theorem 6.5 k, ϕ-MC$_{\mathrm{BT0}}$ *is* W[1]*-complete, even if we restrict to formulae using only one atomic proposition.*

PROOF. This is a corollary of Theorem 6.2 since "$\mathcal{M}, s \models \bigvee_i \mathsf{X}^{k_i}\mathrm{p}_i$?" (a LTL0 question) is equivalent to "$\mathcal{M}, s \models \bigvee_i (\mathsf{EX})^{k_i}\mathrm{p}_i$?" (a BT0 question). □

6.3.2 HML.

We assume familiarity with Hennessy-Milner Logic HML [32]. We consider HML formulae of the form below

$$\phi ::= \mathrm{p} \ | \ \neg\phi \ | \ \phi_1 \vee \phi_2 \ | \ \phi_1 \wedge \phi_2 \ | \ \Box\phi \ | \ \Diamond\phi,$$

where $\mathrm{p} \in AP$ (set of propositional variables). A HML model is a Kripke structure over a unary alphabet and as usual in modal logics, $\mathcal{M}, s \models \Box\phi$ whenever $\mathcal{M}, s' \models \phi$ for every successor state s' of s. Observe that HML cannot express reachability questions since the quantifications used in the interpretation of the modalities $\Box$ and $\Diamond$ involve only direct successors.

Theorem 6.6 *k, ϕ-$\mathrm{MC}_{\mathrm{HML}}$ is $\mathsf{AW}[1]$-complete.*

PROOF. (Idea) That k, ϕ-$\mathrm{MC}_{\mathrm{HML}}$ is fp-equivalent to Short ATM Computation can be proved by adapting the techniques of Theorem 6.2 to ATMs. Actually, the equivalence is shown by restricting the class of ATMs. We exploit the natural correspondence between the modal operators $\Box$ and $\Diamond$ and the behavior of the ATMs with universal states and existential states, respectively. A detailed proof can be found in Appendix C. $\Box$

6.3.3 μ-calculus.

Model checking non-flat systems is $\mathsf{EXPTIME}$-complete for the μ-calculus [47]. We assume familiarity with the modal μ-calculus, see, e.g., [1] and we recall here only a few points. The formulae are defined as follows:

$$\phi ::= \mathrm{p} \ | \ \neg\phi \ | \ \phi_1 \vee \phi_2 \ | \ \phi_1 \wedge \phi_2 \ | \ \Box\phi \ | \ \Diamond\phi \ | \ \mu X \cdot \phi' \ | \ \nu X \cdot \phi',$$

where $\mathrm{p} \in AP$ (set of propositional variables) and the variable X occurs positively in ϕ'. μ [resp. ν] is the least [resp. greatest] fixed-point operator and the models are Kripke structures over Σ. The formulae without fixed-point operators are interpreted as HML formulae and $\mathcal{M}, s \models \mu X \cdot \phi'$ where $\mathcal{M} = \langle Q, \Sigma, \rightarrow, m \rangle$ iff s belongs to the least fixed-point of the operation $f : \mathcal{P}(Q) \rightarrow \mathcal{P}(Q)$ defined as follows ($Z \subseteq Q$):

$$f(Z) = \{s \in Q : \mathcal{M}' = \langle Q, \Sigma, \rightarrow, m \cup Z \times \{X\} \rangle, s \models \phi'\}.$$

In $\mathcal{M}'$, the state variable X is viewed as a new propositional variable.

In our parameterized setting we have:

Theorem 6.7 *k, ϕ-MC_{μ} is XP-complete.*

PROOF. (Sketch) Writing n for $\sum_i |\mathcal{M}_i|$, k,ϕ-MC$_\mu$ can be solved in time $O\Big((|\phi|.n^k)^{|\phi|}\Big)$ [39, Theo. 6.4] and hence is in XP.

XP-hardness is proved by a reduction from Compact ATM Computation. With an ATM M and an integer k we associate a product $\langle \mathcal{A}_1, m_1\rangle \times \cdots \times \langle \mathcal{A}_k, m_k\rangle \times \langle \mathcal{A}_{\text{state}}, m_{\text{state}}\rangle \times \langle \mathcal{A}_{\text{head}}, m_{\text{head}}\rangle$ of $k+2$ KSs that emulate the behavior of M on a k-bounded tape. An $\mathcal{A}_i$ stores the current content of the i-th tape square, $\mathcal{A}_{\text{state}}$ stores the current control-state of M and $\mathcal{A}_{\text{head}}$ stores the position of the TM head. $m_1 = \ldots = m_k = m_{\text{head}} = \emptyset$ and the propositional variables $\mathrm{p}_\exists$, $\mathrm{p}_\forall$ and p_{fin} identify states in $\mathcal{A}_{\text{state}}$ that are existential, universal and final, respectively. For instance $m_{\text{state}}(\mathrm{p}_\exists)$ is the set of existential states of the ATM M. These LTSs synchronize on labels of the form $\langle t, i\rangle$ that stand for "rule t of M is fired while head is in position i" (see also the proof of Theorem 6.6). One can show that M on the empty string input has an accepting run using less than k cells on the working tape iff $\langle \mathcal{A}_1, m_1\rangle \times \cdots \times \langle \mathcal{A}_k, m_k\rangle \times \langle \mathcal{A}_{\text{state}}, m_{\text{state}}\rangle \times \langle \mathcal{A}_{\text{head}}, m_{\text{head}}\rangle, \bar{s} \models \mu Y.\mathrm{p}_{\text{fin}} \vee ((\mathrm{p}_\exists \Rightarrow \Diamond Y) \wedge (\mathrm{p}_\forall \Rightarrow \Box Y))$ for some adequate $\bar{s}$ encoding the initial configuration of M on the empty string. $\Box$

7 Parameterized complexity of non-flat bisimilarity

7.1 Main parameterized problems

We assume familiarity with bisimulation and the other behavioral equivalences in the branching time–linear time spectrum [53]. Checking for bisimilarity among non-flat systems is $\mathsf{EXPTIME}$-complete in the classical framework [34,41]. For our parametric analysis, the general problem is:

Parameterized Bisimulation (Bisim)
Instance: $2k$ LTSs $\mathcal{A}_1, \cdots, \mathcal{A}_k, \mathcal{A}'_1, \cdots, \mathcal{A}'_k$, a configuration $\bar{s}$ of $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$, a configuration $\bar{s'}$ of $\mathcal{A}'_1 \times \cdots \times \mathcal{A}'_k$.
Question: Is $\langle \mathcal{A}_1 \times \cdots \times \mathcal{A}_k, \bar{s}\rangle$ (strongly) bisimilar to $\langle \mathcal{A}'_1 \times \cdots \times \mathcal{A}'_k, \bar{s'}\rangle$ (see Section 3.2)?

Theorem 7.1 *k-Bisim is XP-complete.*

PROOF. (Idea) k-Bisim is in XP since bisimilarity of flat systems is polynomial-time [36]. XP-hardness is seen by observing that the reduction in the proof of [41, Theorem 4.1] is an fp-reduction from Compact ATM Computation to k-Bisim. $\Box$

This result is robust and Section 7.2 shows that it still holds when we consider binary synchronization or restricted alphabets (a result we used in the proof of Theorem 6.7).

Regarding other behavioral equivalences in the linear time–branching time spectrum [53], we have two generic hardness results (where $\subseteq_{\mathrm{tr}}$, $\sqsubseteq$ and $\sim$ denote resp. trace inclusion, the simulation preorder, and bisimilarity):

Theorem 7.2 *For any relation R lying between trace inclusion and bisimilarity, co*COMPACT NDTM COMPUTATION *is fp-reducible to k-R-*CHECKING*, i.e., the problem of checking whether $\langle \mathcal{A}_1 \times \cdots \times \mathcal{A}_k, \bar{s}\rangle R \langle \mathcal{A}'_1 \times \cdots \times \mathcal{A}'_k, \bar{s'}\rangle$.*

PROOF. We rely on Theorem 5.1 and reduce the complement of k-EXACT-REACH to k-R-CHECKING.

Let $\mathcal{A}_1, \cdots, \mathcal{A}_k$, $\bar{s}$, $\bar{t}$ be an instance of k-EXACT-REACH where $\bar{t} = \langle t_1, \ldots, t_k\rangle$. Using a new label #, we add a loop $t_i \xrightarrow{\#} t_i$ to every $\mathcal{A}_i$, yielding a modified LTS $\mathcal{A}'_i$ that can signal reaching $\bar{t}$. Now, writing $\mathcal{S}$ for $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ and $\mathcal{S}'$ for $\mathcal{A}'_1 \times \cdots \times \mathcal{A}'_k$, we have

$$(1)\ \langle \mathcal{S}, \bar{s}\rangle \sim \langle \mathcal{S}', \bar{s}\rangle \ \text{ iff } \ (2)\ \langle \mathcal{S}', \bar{s}\rangle \subseteq_{\mathrm{tr}} \langle \mathcal{S}, \bar{s}\rangle \ \text{ iff } \ (3)\ \bar{s} \xrightarrow{*} \bar{t} \text{ does not hold in } \mathcal{S}.$$

(1) implies (2) because bisimilarity is an equivalence relation smaller than trace inclusion. (2) implies (3) because if $\bar{s} \xrightarrow{*} \bar{t}$ holds in $\mathcal{S}$, then there is a trace from $\bar{s}$ to $\bar{t}$ that contains # and such a trace cannot be obtained in $\mathcal{S}$. Moreover, (3) implies (1) because if $\bar{s} \xrightarrow{*} \bar{t}$ does not hold in $\mathcal{S}$, then in checking the transfer property for $\langle \mathcal{S}, \bar{s}\rangle \sim \langle \mathcal{S}', \bar{s}\rangle$, the label # cannot occur and hence $\langle \mathcal{S}, \bar{s}\rangle \sim \langle \mathcal{S}', \bar{s}\rangle$ holds true.

If now R is larger than bisimilarity but smaller than trace inclusion, $\bar{s} \xrightarrow{*} \bar{t}$ in $\mathcal{S}$ iff $\langle \mathcal{S}, \bar{s}\rangle R \langle \mathcal{S}', \bar{s}\rangle$ does not hold. □

Theorem 7.3 *For any relation R lying between the simulation preorder and bisimilarity, k-R-*CHECKING *is* XP*-hard.*

This is a corollary of the proof of [41, Theorem 4.1]. Theorem 7.3 above and [41, Theorem 4.5] lead to XP-completeness of k-R-CHECKING for $R = \sqsubseteq$ and many known behavioral equivalences between $\sqsubseteq$ and $\sim$.

This result can be strengthened when considering non-flat systems with *hiding*, where a subset of the alphabet is associated with the synchronized product and specifies which labels must be hidden (replaced by τ) in the resulting

LTS. In [48] it is shown that checking any relation lying between trace inclusion and bisimilarity for these systems is EXPTIME-hard. In the parameterized complexity setting, the same reduction can be done [49] and provides XP-hardness for k-R-CHECKING-H for any R between trace inclusion and bisimilarity (k-R-CHECKING-H is the generalization of k-R-CHECKING where hiding is allowed).

7.2 Variants of bisimulation checking

The aim of the following lemmas is to reduce bisimulation problems from LTSs over arbitrary alphabets to LTSs over binary alphabets. This result is useful in the proof of Theorem 6.7 about model checking of μ-calculus and it is also a result of independent interest.

The following result can be proved by using the forthcoming Lemmas 7.5 and 7.6 and the construction $\widehat{\mathcal{A}}$ defined in section 5.2.

Lemma 7.4 *k-BISIM and k-BISIM $_{|\Sigma|=2}$ are fp-equivalent.*

PROOF. Let $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ and $\mathcal{B}_1 \times \cdots \times \mathcal{B}_k$ be LTSs over Σ. The following are equivalent:

- $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k, \langle s_1, \ldots, s_k\rangle \sim \mathcal{B}_1 \times \cdots \times \mathcal{B}_k, \langle t_1, \ldots, t_k\rangle$
- $\widehat{\mathcal{A}_1 \times \cdots \times \mathcal{A}_k}, \langle\langle s_1, \ldots, s_k\rangle, \varepsilon\rangle \sim \widehat{\mathcal{B}_1 \times \cdots \times \mathcal{B}_k}, \langle\langle t_1, \ldots, t_k\rangle, \varepsilon\rangle$
  (by Lemma 7.5 below)
- $\widehat{\mathcal{A}_1} \times \ldots \times \widehat{\mathcal{A}_k}, \langle\langle s_1, \varepsilon\rangle, \ldots, \langle s_k, \varepsilon\rangle\rangle \sim \widehat{\mathcal{B}_1} \times \ldots \times \widehat{\mathcal{B}_k}, \langle\langle t_1, \varepsilon\rangle, \ldots, \langle t_k, \varepsilon\rangle\rangle$
  (by Lemma 7.6 and bisimulation is a congruence for synchronized product)

We thus have a reduction of k-BISIM to k-BISIM $_{|\Sigma|=2}$ and there only remains to check this is an fp-reduction. This is clear since $n' = \sum_{i=1}^{k}|\widehat{\mathcal{A}_i}| + \sum_{i=1}^{k}|\widehat{\mathcal{B}_i}|$ is in $O\Big(l \times m \times (\sum_{i=1}^{k}|\mathcal{A}_i| + \sum_{i=1}^{k}|\mathcal{B}_i|)\Big)$. $\square$

Lemma 7.5 *Let $\mathcal{A} = \langle Q, \Sigma, \rightarrow\rangle$, $\mathcal{A}' = \langle Q', \Sigma, \rightarrow'\rangle$ be LTSs over Σ and q, q' be states of $\mathcal{A}$ and $\mathcal{A}'$, respectively. Then, $\langle \mathcal{A}, q\rangle \sim \langle \mathcal{A}', q'\rangle$ iff $\langle \widehat{\mathcal{A}}, \langle q, \varepsilon\rangle\rangle \sim \langle \widehat{\mathcal{A}'}, \langle q', \varepsilon\rangle\rangle$.*

PROOF. (Sketch)
($\Rightarrow$) : Assume that $\langle \mathcal{A}, q\rangle \sim \langle \mathcal{A}', q'\rangle$, that is there is a bisimulation $R \subseteq Q \times Q'$ such that $\langle q, q'\rangle \in R$. Let $R' \subseteq \widehat{Q} \times \widehat{Q'}$ be defined as $\{\langle\langle q, v\rangle, \langle q', v'\rangle\rangle : qRq',\ v = v'\}$. One can check that R' has the transfer property (see Section 3.2) by an immediate use of the transfer property of R, and hence R' is a bisimulation.

($\Leftarrow$): Assume that $\langle \widehat{\mathcal{A}}, \langle q, \varepsilon\rangle\rangle \sim \langle \widehat{\mathcal{A}'}, \langle q', \varepsilon\rangle\rangle$, that is there is a bisimulation $R \subseteq \widehat{Q} \times \widehat{Q'}$ such that $\langle q, \varepsilon\rangle R \langle q', \varepsilon\rangle$. Let $R' \subseteq Q \times Q'$ be defined as $\{\langle q, q'\rangle : \langle q, \varepsilon\rangle R \langle q', \varepsilon\rangle\}$. Then R' has the transfer property. Indeed, assume that $qR'q'$ and $q \xrightarrow{a_i} r$. By definition of $\widehat{\mathcal{A}}$ we have, $\langle q, \varepsilon\rangle \xrightarrow{i_1} \langle q, i_1\rangle \xrightarrow{i_2} \langle q, i_1 i_2\rangle \ldots \xrightarrow{i_l} \langle q, i\rangle \xrightarrow{0} \langle r, \varepsilon\rangle$ where $i_1 \ldots i_l$ is the binary writing of i. Since R is a bisimulation, there are $r_1, \ldots, r_{l+1}$ such that $\langle q', \varepsilon\rangle \xrightarrow{i_1} r_1 \xrightarrow{i_2} r_2 \ldots \xrightarrow{i_l} r_l \xrightarrow{0} r_{l+1}$, $\langle r, \varepsilon\rangle \; R \; r_{l+1}$ and for $1 \le j \le l$, $\langle q, i_1 \ldots i_j\rangle \; R \; r_j$. But the form of $\widehat{\mathcal{A}'}$ implies that this sequence is such that for $1 \le j \le l$, $q_j = \langle q', i_1 \ldots i_j\rangle$ and $r_{l+1} = \langle r', \varepsilon\rangle$ for some $q' \xrightarrow{a_i} r'$. Hence, there is some $q' \xrightarrow{a_i} r'$ with $\langle r, \varepsilon\rangle R \langle r', \varepsilon\rangle$, i.e., $rR'r'$. $\square$

Lemma 7.6 *Let $\mathcal{A} = \langle Q, \Sigma, \rightarrow\rangle$, $\mathcal{A}' = \langle Q', \Sigma, \rightarrow'\rangle$ be LTSs over Σ, and q, q' be states of $\mathcal{A}$ and $\mathcal{A}'$, respectively. Then, $\langle \widehat{\mathcal{A} \times \mathcal{A}'}, \langle\langle q, q'\rangle, \varepsilon\rangle\rangle \sim \langle \widehat{\mathcal{A}} \times \widehat{\mathcal{A}'}, \langle\langle q, \varepsilon\rangle, \langle q', \varepsilon\rangle\rangle\rangle$.*

PROOF. (Sketch) One can check that $R = \{\langle\langle\langle q, q'\rangle, v\rangle, \langle\langle q, v\rangle, \langle q', v\rangle\rangle\rangle : q \in Q, q' \in Q', v \in \{0,1\}^i, 0 \le i \le l\}$ is a bisimulation. $\square$

As a conclusion,

Theorem 7.7 *k-*Bisim *$_{|\Sigma|=2}$ is* XP*-complete.*

8 Conclusion

We studied the complexity of model checking synchronized products of LTSs in the light of Downey and Fellows's parameterized complexity. In our study the parameter k is the number of components (and the size of the property). We considered a wide variety of problems, and assumed two different synchronization protocols.

It is known that for any fixed value of the parameter, the problems have polynomial-time solutions in $O(n^k)$ and we show that solutions in some $f(k) \times n^c$ (for some constant c) do not exist (unless the W-hierarchy collapses). Therefore our results show that these problems are probably not tractable *even in the parameterized sense of being FPT*, and indeed can in general be situated quite high in the hierarchy (see the summary in Fig. 2 where edges correspond to the existence of an fp-reduction).

The problems remain intractable (possibly at a weaker level) when natural restrictions are imposed. We think this must be understood as arguing against any hope of finding "tractable" algorithms for model checking synchronized products of components even when the number k of components varies much

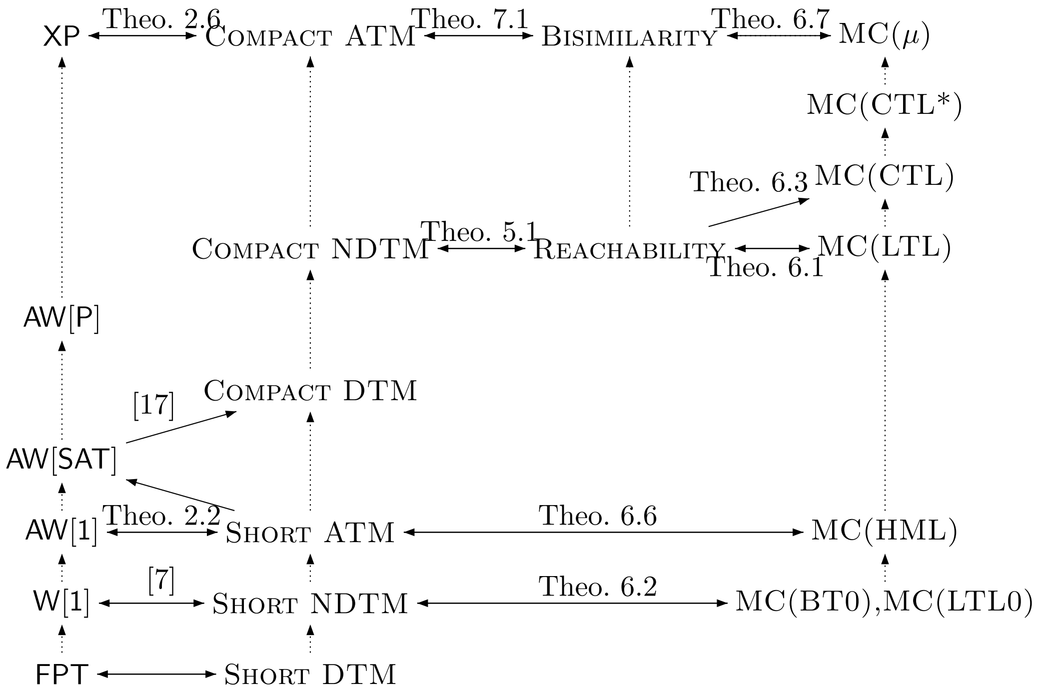

Fig. 2. A summary of existing reductions between parameterized problems

less than the size of the components themselves. We do not think the difficulty can be solved by considering other ways of treating k as a parameter (e.g., the theory of [5] where part of the input is processed off-line).

References

[1] A. Arnold and D. Niwiński. *Rudiments of μ-Calculus*, volume 146 of *Studies in Logic and the Foundations of Mathematics*. Elsevier Science, 2001.

[2] B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and Ph. Schnoebelen. *Systems and Software Verification. Model-Checking Techniques and Tools*. Springer, 2001.

[3] H. L. Bodlaender, R. G. Downey, M. R. Fellows, and H. T. Wareham. The parameterized complexity of sequence alignment and consensus. *Theoretical Computer Science*, 147(1–2):31–54, 1995.

[4] J. C. Bradfield and C. Stirling. Modal logics and mu-calculi: an introduction. In J. A. Bergstra, A. Ponse, and S. A. Smolka, editors, *Handbook of Process Algebra*, chapter 4, pages 293–330. Elsevier Science, 2001.

[5] M. Cadoli, F. M. Donini, P. Liberatore, and M. Schaerf. Preprocessing of intractable problems. *Information and Computation*, 176(2):89–120, 2002.

[6] L. Cai, J. Chen, R. G. Downey, and M. R. Fellows. Advice classes of parameterized tractability. *Annals of Pure and Applied Logic*, 84(1):119–138, 1997.

[7] L. Cai, J. Chen, R. G. Downey, and M. R. Fellows. On the parameterized complexity of short computation and factorization. *Archive for Mathematical Logic*, 36(4/5):321–337, 1997.

[8] M. Cesati. The Turing way to parameterized intractability. *Journal of Computer and System Sciences*, 67(4):654–685, 2003.

[9] M. Cesati and M. Di Ianni. Computation models for parameterized complexity. *Mathematical Logic Quarterly*, 43:179–202, 1997.

[10] Y. Chen and J. Flum. Machine characterization of the classes of the W-hierarchy. In *Proc. 17th Int. Workshop Computer Science Logic (CSL 2003) and 8th Kurt Gödel Coll. (KGL 2003), Vienna, Austria, Aug. 2003*, volume 2803 of *Lecture Notes in Computer Science*, pages 114–127. Springer, 2003.

[11] Y. Chen, J. Flum, and M. Grohe. Bounded nondeterminism and alternation in parameterized complexity theory. In *Proc. 18th Annual IEEE Conference on Computational Complexity (Complexity 2003), Aarhus, Denmark, July 2003*, pages 13–29. IEEE Comp. Soc. Press, 2003.

[12] E. M. Clarke and E. A. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Proc. Logics of Programs Workshop, Yorktown Heights, New York, May 1981*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer, 1981.

[13] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, 1999.

[14] S. Demri, F. Laroussinie, and Ph. Schnoebelen. A parametric analysis of the state explosion problem in model checking (extended abstract). In *Proc. 19th Ann. Symp. Theoretical Aspects of Computer Science (STACS 2002), Antibes Juan-les-Pins, France, Mar. 2002*, volume 2285 of *Lecture Notes in Computer Science*, pages 620–631. Springer, 2002.

[15] S. Demri and Ph. Schnoebelen. The complexity of propositional linear temporal logics in simple cases. *Information and Computation*, 174(1):84–103, 2002.

[16] R. G. Downey, V. Estivill-Castro, M. R. Fellows, E. Prieto, and F. A. Rosamond. Cutting up is hard to do: the parameterized complexity of k-cut and related problems. In *Proc. Computing: the Australasian Theory Symposium (CATS 2003), Adelaide, Australia, Feb. 2003*, volume 78 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science, 2003.

[17] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

[18] R. G. Downey, M. R. Fellows, and U. Stege. Computational tractability: The view from Mars. *EATCS Bull.*, 69:73–97, 1999.

[19] R. G. Downey, M. R. Fellows, and U. Stege. Parameterized complexity: A framework for systematically confronting computational intractability. In *Contemporary Trends in Discrete Mathematics: From DIMACS and DIMATIA to the Future*, volume 49 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 49–99. American Mathematical Society, 1999.

[20] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 16, pages 995–1072. Elsevier Science, 1990.

[21] E. A. Emerson, C. S. Jutla, and A. P. Sistla. On model-checking for the μ-calculus and its fragments. *Theoretical Computer Science*, 258(1–2):491–522, 2001.

[22] J. Esparza. Decidability and complexity of Petri net problems — an introduction. In *Advances in Petri Nets 1998*, volume 1491 of *Lecture Notes in Computer Science*, pages 374–428. Springer, 1998.

[23] J. Feigenbaum, S. Kannan, M. Y. Vardi, and M. Viswanathan. The complexity of problems on graphs represented as OBDDs. *Chicago J. Theoretical Computer Science*, 1999.

[24] J. Flum and M. Grohe. Fixed-parameter tractability, definability, and model checking. *SIAM Journal on Computing*, 31(1):43–73, 2001.

[25] J. Flum and M. Grohe. Describing parameterized complexity classes. In *Proc. 19th Ann. Symp. Theoretical Aspects of Computer Science (STACS'2002), Antibes Juan-les-Pins, France, Mar. 2002*, volume 2285 of *Lecture Notes in Computer Science*, pages 359–371. Springer, 2002.

[26] J. Flum and M. Grohe. Model-checking problems as a basis for parameterized intractability. In *Proc. 19th IEEE Symp. Logic in Computer Science (LICS 2004), Turku, Finland, July 2004*, pages 388–397. IEEE Computer Society Press, 2004.

[27] D. M. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Proc. 7th ACM Symp. Principles of Programming Languages (POPL '80), Las Vegas, NV, USA, Jan. 1980*, pages 163–173. ACM Press, 1980.

[28] Z. Galil. Hierarchies of complete problems. *Acta Informatica*, 6(2):77–88, 1976.

[29] M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. Freeman, 1979.

[30] M. Grohe, T. Schwentick, and L. Segoufin. When is the evaluation of conjunctive queries tractable? In *Proc. 33rd ACM Symp. Theory of Computing (STOC 2001), Heraklion, Crete, Greece, July 2001*, pages 657–666, 2001.

[31] D. Harel, O. Kupferman, and M. Y. Vardi. On the complexity of verifying concurrent transition systems. *Information and Computation*, 173(2):143–161, 2002.

[32] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32(1):137–161, 1985.

[33] N. Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing*, 17(5):935–938, 1988.

[34] L. Jategaonkar and A. R. Meyer. Deciding true concurrency equivalences on safe, finite nets. *Theoretical Computer Science*, 154(1):107–143, 1996.

[35] N. D. Jones, L. H. Landweber, and Y. E. Lien. Complexity of some problems in Petri nets. *Theoretical Computer Science*, 4(3):277–299, 1977.

[36] P. C. Kanellakis and S. A. Smolka. CCS expressions, finite state processes and three problems of equivalence. *Information and Computation*, 86(1):43–68, 1990.

[37] T. Kasai, A. Adachi, and S. Iwata. Classes of pebble games and complete problems. *SIAM J. Computing*, 8(4):574–586, 1979.

[38] D. C. Kozen. Lower bounds for natural proof systems. In *Proc. 18th IEEE Symp. Foundations of Computer Science (FOCS '77), Providence, RI, USA, Oct.-Nov. 1977*, pages 254–266. IEEE Comp. Soc. Press, 1977.

[39] O. Kupferman, M. Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, 2000.

[40] K.-L. Lange and P. Rossmanith. The emptiness problem for intersections of regular languages. In *Proc. 17th Int. Symp. Math. Found. Comp. Sci. (MFCS '92), Prague, Czechoslovakia, Aug. 1992*, volume 629 of *Lecture Notes in Computer Science*, pages 346–354. Springer, 1992.

[41] F. Laroussinie and Ph. Schnoebelen. The state explosion problem from trace to bisimulation equivalence. In *Proc. 3rd Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS 2000), Berlin, Germany, Mar.-Apr. 2000*, volume 1784 of *Lecture Notes in Computer Science*, pages 192–207. Springer, 2000.

[42] O. Lichtenstein and A. Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *Proc. 12th ACM Symp. Principles of Programming Languages (POPL '85), New Orleans, LA, USA, Jan. 1985*, pages 97–107, 1985.

[43] N. Markey and Ph. Schnoebelen. Model checking a path. In *Proc. 14th Int. Conf. Concurrency Theory (CONCUR 2003), Marseille, France, Sep. 2003*, volume 2761 of *Lecture Notes in Computer Science*, pages 251–265. Springer, 2003.

[44] C. H. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

[45] C. H. Papadimitriou and M. Yannakakis. On the complexity of database queries. *Journal of Computer and System Sciences*, 58(3):407–427, 1999.

[46] A. Rabinovich. Complexity of equivalence problems for concurrent systems of finite agents. *Information and Computation*, 139(2):111–129, 1997.

[47] A. Rabinovich. Symbolic model checking for μ-calculus requires exponential time. *Theoretical Computer Science*, 243(1–2):467–475, 2000.

[48] Z. Sawa. Equivalence checking of non-flat systems is EXPTIME-hard. In *Proc. 14th Int. Conf. Concurrency Theory (CONCUR 2003), Marseille, France, Sep. 2003*, volume 2761 of *Lecture Notes in Computer Science*, pages 233–248. Springer, 2003.

[49] Z. Sawa. Personal communication, April 2005.

[50] Ph. Schnoebelen. The complexity of temporal logic model checking. In *Advances in Modal Logic, vol. 4*, pages 437–459. King's College Publication, 2003.

[51] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, 1985.

[52] A. Valmari and A. Kervinen. Alphabet-based synchronisation is exponentially cheaper. In *Proc. 13th Int. Conf. Concurrency Theory (CONCUR 2002), Brno, Czech Republic, Aug. 2002*, volume 2421 of *Lecture Notes in Computer Science*, pages 161–176. Springer, 2002.

[53] R. van Glabbeek. The linear time – branching time spectrum I. In J. Bergstra, A. Ponse, and S. Smolka, editors, *Handbook of Process Algebra*, chapter 1, pages 3–99. Elsevier Science, 2001.

[54] K. W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23(3):325–356, 1986.

[55] H. T. Wareham. The parameterized complexity of intersection and composition operations on sets of finite-state automata. In *Proc. 5th Int. Conf. Implementation and Application of Automata (CIAA 2000), London, ON, Canada, July 2000*, volume 2088 of *Lecture Notes in Computer Science*, pages 302–310. Springer, 2001.

A Fp-equivalence of the non-flat reachability problems

We provide the fp-reduction that concludes the proof of Theorem 4.1.

<u>k-Rep-Reach $\leq^{\text{fp}}_{\text{m}}$ k-Exact-Reach:</u>
Let $\mathcal{A}_1, \ldots, \mathcal{A}_k$, $\overline{s} = \langle q_1, \ldots, q_k \rangle$, and $F_1, \ldots, F_k$ be an instance of k-Rep-Reach. W.l.o.g. we assume the LTSs are over a common Σ.

Let Σ' be $\Sigma \cup \{a' : a \in \Sigma\} \cup \{choice\}$ where every a' is a copy of the original a, and where *choice* is a new symbol. For $1 \leq i \leq k$, we build a LTS $\mathcal{A}'_i$ over Σ' as follows. Write m_i for $|F_i|$ and assume F_i is $\{f_{i,1}, \dots, f_{i,m_i}\}$. Each $\mathcal{A}'_i$ contains $m_i + 1$ disjoint copies of $\mathcal{A}_i$, say $\mathcal{A}'_{i,0}, \mathcal{A}'_{i,1}, \dots, \mathcal{A}'_{i,m_i}$, plus a new (final) state x_i. For a state $q \in Q_i$, we write q^j to denote the jth copy of q (in $\mathcal{A}'_{i,j}$).

Additional transitions of $\mathcal{A}'_i$ are given by:

- For $1 \leq j \leq m_i$, there is a transition $f^0_{i,j} \xrightarrow{choice} f^j_{i,j}$. This corresponds to choosing $f_{i,j}$ in the ith component of the repeated state from $F_1 \times \cdots \times F_k$. The extra label *choice* allows synchronization.
- For $1 \leq j \leq m_i$, for $q \in Q_i$, there is a transition $q^j \xrightarrow{a'} x_i$ if $q \xrightarrow{a} f_{i,j}$ in $\mathcal{A}_i$. This means we can only leave the $\mathcal{A}_{i,j}$ part by revisiting $f_{i,j}$.

An illustration of the construction can be found in Fig. A.1 where the nodes in bold are the respective elements of F. $\mathcal{A}'$ simulates $\mathcal{A}$ and then decides that it has reached an adequate $t \in F$. Then $\mathcal{A}'$ enters the t-copy of $\mathcal{A}$ (by a *choice* transition) and can exit it only by reaching t in that very t-copy (by a primed letter transition). The introduction of the letters a' for $a \in \Sigma$ allows us to preserves determinism whereas the transitions labeled by *choice* are used for the synchronization of the different components. Then we clearly

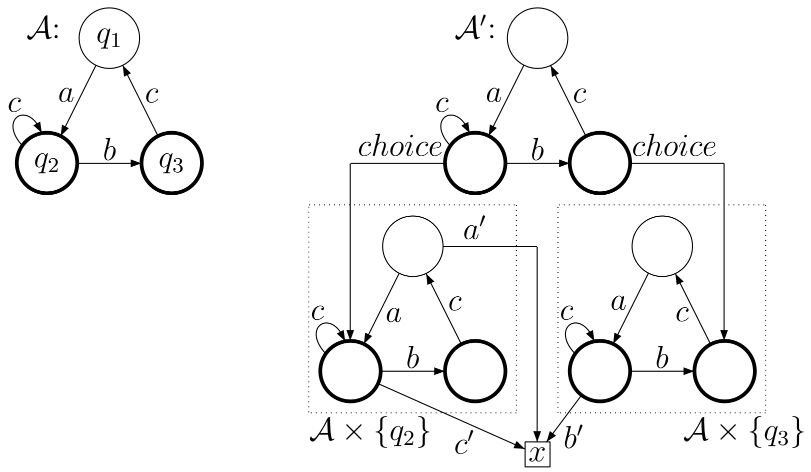

Fig. A.1. $\mathcal{A}$ and $\mathcal{A}'$: an example

have $\bar{s} \xrightarrow{*} \bar{t} \xrightarrow{+} \bar{t}$ for some $\bar{t}$ of the form $\langle f_{1,j_1}, \dots, f_{k,j_k}\rangle$ in $F_1 \times \cdots \times F_k$ iff $\langle q^0_1, \dots, q^0_k\rangle \xrightarrow{*} \xrightarrow{choice} \langle f^{j_1}_1, \dots, f^{j_k}_k\rangle \xrightarrow{+} \langle x_1, \dots, x_k\rangle$ in $\mathcal{A}'_1 \times \cdots \times \mathcal{A}'_k$, that is iff $\langle q^0_1, \dots, q^0_k\rangle \xrightarrow{*} \langle x_1, \dots, x_k\rangle$. Thus we have reduced our problem to an instance of k-Exact-Reach.

Finally, since $|\mathcal{A}'_i|$ is in $O((m_i + 1) \times |\mathcal{A}_i|)$, we have

$$k' = k, \qquad |\Sigma'| = 2 \times |\Sigma| + 1, \qquad n' \text{ is in } O(n^2),$$

so that the reduction is an fp-reduction with parameter k as well as with parameter k, Σ.

In addition, the reader can observe that, since we were careful and made copies a' of the $a \in \Sigma$, the resulting $\mathcal{A}'_i$s are deterministic when the $\mathcal{A}_i$s are. $\square$

B Proof of Theorem 5.4

We show that (I) k-$*$-Reach $_{\text{bin}}$ is fp-reducible to k-$*$-Reach, and that reciprocally (II) k-$*$-Reach is fp-reducible to k-$*$-Reach $_{\text{bin}}$. Then Theorem 5.1 concludes.

(I) Let $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ be a product of LTSs assuming binary synchronization. Let Σ' be $\Sigma \times \{\{i,j\} : 1 \leq i < j \leq k\}$ and $\mathcal{A}'_1, \ldots, \mathcal{A}'_k$ be LTSs over Σ' defined with the idea that $a, \{i,j\}$ means that $\mathcal{A}_i$ and $\mathcal{A}_j$ synchronize on a while the other components do not move.

Formally each $\mathcal{A}'_i$ is defined from $\mathcal{A}_i$ by replacing each transition $s \xrightarrow{a} t$ by

$$\{s \xrightarrow{a,\{i,j\}} t : j \neq i\} \cup \{s \xrightarrow{a,\{j,j'\}} s : i \notin \{j,j'\}, j \neq j'\}.$$

Clearly, $\bar{s} \xrightarrow{a_1 \ldots a_n}_{\text{bin}} \bar{t}$ in $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ iff $\bar{s} \xrightarrow{a_1,\{i_1,j_1\} \ldots a_n,\{i_n,j_n\}}_{\text{str}} \bar{t}$ in $\mathcal{A}'_1 \times \cdots \times \mathcal{A}'_k$ for some $\{i_1,j_1\}, \ldots, \{i_n,j_n\} \in \{\{i,j\} : 1 \leq i < j \leq k\}$. Hence, any of the four non-flat reachability problems using binary synchronization is reducible to the analogous problem with strong synchronization. Now, since $|\mathcal{A}'_i|$ is in $O(k^2 \times |\mathcal{A}_i|)$, so that

$$\begin{aligned} k' &= k, \\ |\Sigma'| &= |\Sigma| \times k \times (k-1), \\ n' &\text{ is in } O(k^2 \times n), \end{aligned}$$

we see the reductions are fp-reductions with either parameter k or parameter k, Σ.

(II) Let $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ be a product of LTSs assuming strong synchronization. Let Σ' be $\Sigma \times \{1, \ldots, k+1\}$ and $\mathcal{A}'_0, \cdots, \mathcal{A}'_k$ be LTSs over Σ' defined as follows. Each $\mathcal{A}'_i$ is defined from $\mathcal{A}_i$ by replacing each transition $s \xrightarrow{a} t$ by the two transitions $s \xrightarrow{a,i} \langle s,a,t \rangle \xrightarrow{a,i+1} t$ where $\langle s,a,t \rangle$ is a new state. The LTS $\mathcal{A}'_0$ is a new controller with states $\{0\} \cup \{s_a : a \in \Sigma\}$ such that for $a \in \Sigma$, there are loops $0 \xrightarrow{a,1} s_a \xrightarrow{a,k+1} 0$. One can show that for $\bar{s}, \bar{t}$, $\bar{s} \xrightarrow{a}_{\text{str}} \bar{t}$ in $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ iff $0, \bar{s} \xrightarrow{a,1 \cdot \ldots \cdot a,k+1}_{\text{bin}} 0, \bar{t}$ in $\mathcal{A}'_0 \times \mathcal{A}'_1 \times \cdots \times \mathcal{A}'_k$.

Hence, any of the four non-flat reachability problems using strong synchronization is reducible to the analogous problem with binary synchronization (for LOCAL-REACH, REP-REACH and FAIR-REACH, a set $\bar{F} \subseteq Q_1 \times \cdots \times Q_k$ in the instance of the problem is replaced by $\{0\} \times \bar{F}$ in the reduction).

Since we further have

$$\begin{aligned} k' =& k+1, \\ |\Sigma'| =& |\Sigma| \times (k+1), \\ \sum_{i=0}^{k} |\mathcal{A}'_i| \text{ is in } & O\Big(\sum_{i=1}^{k} |\mathcal{A}_i| + |\Sigma|\Big), \end{aligned}$$

the reductions are fp-reductions with either parameter k or parameter k, Σ.

So k-$*$-REACH $_{\text{bin}}$ and k-$*$-REACH are fp-equivalent and, k, Σ-$*$-REACH$_{\text{bin}}$ and k, Σ-$*$-REACH are fp-equivalent. By Theorem 5.5, k, Σ-$*$-REACH$_{\text{bin}}$ and k-$*$-REACH are fp-equivalent. $\square$

C Proof of Theorem 6.6

We shall show that k, ϕ-MC$_{\text{HML}}$ is fp-equivalent to STRICT SHORT ATM COMPUTATION (defined earlier). We first establish an fp-reduction from STRICT SHORT ATM COMPUTATION to k, ϕ-MC$_{\text{HML}}$ which entails $\mathsf{AW}[1]$-hardness by Lemma 2.4 and Theorem 2.2. To do so, we use the techniques of the reduction shown in Lemma 5.2. In order to be self-contained, we provide full details. Let $M = \langle Q^{\exists} \cup Q^{\forall}, \Sigma, \delta, q_0, q_F \rangle$ be a strict ATM with blank symbol $B \in \Sigma$ and k be a positive integer. We build the Kripke structures $\mathcal{M}_1 = \langle \mathcal{A}_1, m_1 \rangle$, $\ldots$, $\mathcal{M}_k = \langle \mathcal{A}_k, m_k \rangle$, $\mathcal{M}_{\text{state}} = \langle \mathcal{A}_{\text{state}}, m_{\text{state}} \rangle$, $\mathcal{M}_{\text{head}} = \langle \mathcal{A}_{\text{head}}, m_{\text{head}} \rangle$ with $m_1 = \ldots = m_k = m_{\text{head}} = \emptyset$ and $m_{\text{state}} = \{\langle q_F, \mathrm{p} \rangle\}$ such that M on the empty string input has an accepting run using less than k steps iff $\mathcal{M}_1 \times \cdots \times \mathcal{M}_{\text{head}}, \bar{s} \models \bigvee_{0 \leq i \leq \lfloor \frac{k}{2} \rfloor} (\Box \Diamond)^i \mathrm{p}$ for some adequate $\bar{s}$ (to be defined later on).

An $\mathcal{A}_i$ stores the current content of the i-th tape square. $\mathcal{A}_{\text{head}}$ stores the position of the TM head. Similarly, $\mathcal{A}_{\text{state}}$ stores the control state of the current configuration. These LTSs synchronize on labels of the form $\langle t, i \rangle$ that stand for "rule t of M is fired while head is in position i". Successful acceptance by M is directly encoded by $\bigvee_{0 \leq i \leq \lfloor \frac{k}{2} \rfloor} (\Box \Diamond)^i \mathrm{p}$. We provide below the formal definition of the LTSs given that $\Sigma' = \{\langle t, j \rangle : t \in \delta,\ 1 \leq j \leq k\}$.

- $\mathcal{A}_i = \langle Q_i, \Sigma', \rightarrow_i \rangle$ with $Q_i = \Sigma$, $a \xrightarrow{\langle t,i \rangle}_i a'$ if t is of the form $\langle q, a, q', a', m \rangle$, and $a \xrightarrow{\langle t,j \rangle}_i a$ if $j \neq i$.
- $\mathcal{A}_{\text{state}} = \langle Q_{\text{state}}, \Sigma', \rightarrow_{\text{state}} \rangle$ with $Q_{\text{state}} = Q^{\exists} \cup Q^{\forall}$, and $q \xrightarrow{\langle t,i \rangle} q'$ if t is of the

form $\langle q, a, q', a', m \rangle$.

- $\mathcal{A}_{\text{head}} = \langle Q_{\text{head}}, \Sigma', \rightarrow_{\text{head}} \rangle$ with $Q_{\text{head}} = \{0, \ldots, k+1\}$ and $\rightarrow_{\text{head}}$ is the union of the following sets:
  - $i \xrightarrow{t,i}_{\text{head}} i+1$ for all $i \in \{1, \ldots, k-1\}$ and $t = \langle q, a, q', a, m \rangle \in \delta$ with $m = R$ (the head moves to the right);
  - $i \xrightarrow{t,i}_{\text{head}} i-1$ for all $i \in \{1, \ldots, k\}$ and $t = \langle q, a, q', a, m \rangle \in \delta$ with $m = L$ (the head moves to the left);
  - $i \xrightarrow{t,i}_{\text{head}} i$ for all $i \in \{1, \ldots, k\}$ and $t = \langle q, a, q', a, m \rangle \in \delta$ with $m = -$ (the head does not move).

$\mathcal{M}_1 \times \cdots \times \mathcal{M}_k \times \mathcal{M}_{\text{state}} \times \mathcal{M}_{\text{head}}, \langle B, \ldots, B, q_0, 1 \rangle \models \bigvee_{0 \leq i \leq \lfloor \frac{k}{2} \rfloor} (\Box\Diamond)^i \mathrm{p}$ iff M on the empty string input has an accepting run using less than k steps.
Observe that $|\mathcal{M}_1| + \cdots + |\mathcal{M}_k| + |\mathcal{M}_{\text{state}}| + |\mathcal{M}_{\text{head}}| + |\bigvee_{0 \leq i \leq \lfloor \frac{k}{2} \rfloor} (\Box\Diamond)^i \mathrm{p}|$ is polynomial in $|M| + k$, $k+2$ Kripke structures are involved in the target instance and $|\bigvee_{0 \leq i \leq \lfloor \frac{k}{2} \rfloor} (\Box\Diamond)^i \mathrm{p}|$ depends only on k. Hence, we really have defined an fp-reduction.

Now, let us show how to reduce k, ϕ-MC$_{\text{HML}}$ to Short ATM Computation. Let $\mathcal{M}_1 = \langle \mathcal{A}_1, m_1 \rangle$, $\ldots$, $\mathcal{M}_k = \langle \mathcal{A}_k, m_k \rangle$, be k Kripke structures, $\bar{s_0} = \langle s_1^0, \ldots, s_k^0 \rangle$ be a configuration of $\mathcal{M}_1 \times \cdots \times \mathcal{M}_k$ and ϕ be an HML formula. Without any loss of generality, we can assume that ϕ is in negative normal form (NNF), i.e., negation "$\neg$" occurs only in front of propositional variables.

We shall build an ATM $M = \langle Q^\exists \cup Q^\forall, \Sigma, \delta, q_0, q_F \rangle$ such that $\mathcal{M}_1 \times \cdots \times \mathcal{M}_k, \bar{s_0} \models \phi$ iff M on the empty string input has an accepting run using less than $f(k)$ steps for some adequate function $f(\cdot)$.

For each subformula ψ of ϕ we consider a state q_ψ (in $Q = Q^\exists \cup Q^\forall$). We write Q_ϕ to denote the subset of Q composed of states of the form q_ψ. We provide below the main ideas behind the definition of M (some details are hence omitted).

- Q is the union of Q_ϕ and extra states including the final state q_F.
- $q_0 = q_\phi \in Q_\phi$.
- The states of the form $q_{\Box\psi}$ and $q_{\psi_1 \wedge \phi_2}$ are universal states (in $Q^\forall$) whereas the states of the form $q_{\Diamond\psi}$ and $q_{\psi_1 \vee \phi_2}$ are existential ones.
- $\Sigma = \bigcup_i Q_i$. Only k cells on the tape are used along any computation and a current state of the tape is encoded by an element of $Q_1 \times \cdots \times Q_k$. The first task done by M in the state q_0 consists in writing $\bar{s_0}$ on the tape and the head goes back to the first cell (head position by default). This first job requires $O(k)$ steps. More generally, when in a configuration, the current state of the machine is q_ψ and the content of the tape is $\langle s_1, \ldots, s_k \rangle$, the machine M checks whether $\mathcal{M}_1 \times \cdots \times \mathcal{M}_k, \langle s_1, \ldots, s_k \rangle \models \psi$.

- Some transitions do not move the head, which corresponds in the logical setting to evaluate a subformula at the current state (for Boolean operations). For instance, the following transitions belong to δ:
  · $\langle q_{\psi_1 \wedge \psi_2}, a, q_{\psi_i}, a, - \rangle$ $(i = 1, 2)$;
  · $\langle q_{\psi_1 \vee \psi_2}, a, q_{\psi_i}, a, - \rangle$ $(i = 1, 2)$.
- When a state $q_{\Box\psi}$ [resp. $q_{\Diamond\psi}$] is reached, the machine replaces $\langle s_1, \ldots, s_k \rangle$ on the tape by $\langle s'_1, \ldots, s'_k \rangle$ assuming that $\langle s_1, \ldots, s_k \rangle \to \langle s'_1, \ldots, s'_k \rangle$ in $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ and goes back to the first cell of the tape. Again, this requires $O(k)$ steps.
- When a state of the form $q_{\neg \mathrm{p}}$ is reached, the machine goes to the final state q_F iff none of the states on the tape satisfies p on their respective Kripke structure. This requires also $O(k)$ steps. Similarly, when a state of the form q_{p} is reached, the machine goes to the final state q_F iff some of the states on the tape satisfies p on its Kripke structure.

Hence, it is not difficult to show that $\mathcal{M}_1 \times \cdots \times \mathcal{M}_k, \bar{s_0} \models \phi$ iff M on the empty string input has an accepting run using less than $O(2k \times |\phi|)$ steps. Since both k and $|\phi|$ are parameters of the parameterized problem k, ϕ-$\mathrm{MC}_{\mathrm{HML}}$, we really have defined an fp-reduction. $\Box$