A Parametric Analysis of the State Explosion Problem in Model Checking (Extended Abstract)

S. Demri, F. Laroussinie, and Ph. Schnoebelen

Lab. Spécification & Vérification
ENS de Cachan & CNRS UMR 8643
61, av. Pdt. Wilson, 94235 Cachan Cedex France
Email: {demri,fl,phs}@lsv.ens-cachan.fr

Abstract. In model checking, the state explosion problem occurs when one verifies a *non-flat system*, i.e. a system described implicitly 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 [CGP99]. Investigating the computational complexity of model checking started with [SC85], and today the complexity of the main model checking problems is known.

It is now understood that, in practice, the source of intractability is the size of the model and not the size of the property to be checked. This can be illustrated with LTL model checking as an example: while the problem is PSPACE-complete [SC85], it was observed in [LP85] 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 $n = \sum_i |\mathcal{A}_i|$ while S has size $O\Big(\prod_i |\mathcal{A}_i|\Big)$, that is $O(n^k)$, or $O(2^n)$ when k is not fixed.

From a theoretical viewpoint, the state explosion problem seems inescapable in the classical worst-case complexity paradigm. 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 [Esp98,HKV97,JM96,KVW00,Rab97,Rab00,LS00].

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), see e.g. [DF99].

Any of the main model checking problems where the input is 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 $O(n^k)$. 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 algorithms running in time $f(k) \times n^c$ for some function f and constant c are considered tractable (see [DF99] for convincing arguments).

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. existence in a graph of a cycle of size k) while many others seem not to have any such solution (e.g. existence of a clique of size k). Note that these problems are "equivalent" in the classical complexity paradigm.

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).

All the considered problems are shown intractable even in the parameterized viewpoint (but they reach different intractability levels). See summary of results in section 7. 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.

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 [DF99, p. 470]) as shown in [DLS01, Appendix C]. We also introduce, as a useful general tool, parameterized problems for Alternating Turing machines and relate them to Downey and Fellows' W-hierarchy. Finally, we enrich the known catalog of parameterized problems with problems from an important application field.

Related work. Parameterized complexity has been applied to model checking problems where the parameter is the size of the *property to be checked* (or derived from it) and where the model is given explicitly: this has no relation with the state explosion problem and trivially leads to tractability in the parameterized sense for temporal logics (but becomes interesting when one considers more powerful logics [Gro99] or problems with database queries [PY99], or when one tries to identify parameters (e.g. tree width) that make problems tractable [GSS01]).

Parameterized complexity has been applied to problems where the input is, like in our work, a sequence of k synchronized automata, k being the parameter [B$^+$95,War01,Ces01]. These works are concerned with automata-theoretic (or language-theoretic) questions rather than verification and model checking questions.

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 section 4, temporal logic problems in section 5, and behavioral equivalence problems in section 6. As a rule proofs omitted from the main text can be found in [DLS01].

2 Parameterized complexity

We follow [DF99]. 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. [DF99, 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 $\langle x', k'\rangle \in P'$. P and P' are *fixed-parameter equivalent* (fp-equivalent) $\overset{\text{def}}{\Leftrightarrow}$ $P \leq^{\text{fp}}_{\text{m}} P' \leq^{\text{fp}}_{\text{m}} P$. Clearly, if $P \leq^{\text{fp}}_{\text{m}} P'$ and P' is FPT, then P too is FPT.

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 not FPT (or hard for a class conjectured to be strictly larger than FPT).

Downey and Fellows introduced the following hierarchy of classes of parameterized problems [DF99]:

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

where it is known that FPT $\neq$ XP. These classes are closed under fp-equivalence. 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. XP contains all problems that can be solved in time $O(n^k)$ 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, ...) [1].

We don't 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.

2.1 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.

SHORT DTM COMPUTATION is FPT while SHORT NDTM COMPUTATION is W[1]-complete [DF99]. COMPACT COMPUTATION is more complex and reaches high levels in the W-hierarchy: COMPACT NDTM COMPUTATION is AW[P]-hard [ADF95] and COMPACT DTM COMPUTATION is AW[SAT]-hard (see e.g. [DF99]).

Remark 2.1. More precise measures are still lacking and [DF99, 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).

[DF99] does not consider parameterized problems with ATMs, but these proved very useful in our study [2]. Our first results show how they correspond to existing levels of the W-hierarchy:

Theorem 2.2. SHORT ATM COMPUTATION *is AW[1]-complete.*

[1] But see the recent work by Flum & Grohe in this volume.

[2] Note that [Ces01] also introduced parameterized problems on TMs to characterize the parameterized complexity classes W[1], W[2], and W[P].

Proof. We show equivalence with PARAMETERIZED-QBFSAT$_t$, shown AW[1]-complete in [DF99, 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 $k_1 + \cdots + k_{2p}$.

<u>PARAMETERIZED-QBFSAT$_t$ $\leq^{\text{fp}}_{\text{m}}$ SHORT ATM COMPUTATION:</u>

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

<u>SHORT ATM COMPUTATION $\leq^{\text{fp}}_{\text{m}}$ PARAMETERIZED-QBFSAT$_t$:</u>

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 an 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

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

where $\Phi_{\text{seq}}(X, X')$ 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.

Finally, we can easily rewrite Φ as a positive boolean combination of literals with 5 alternations and therefore obtain an instance of PARAMETERIZED-QBFSAT$_5$ with $k' = (k+1)^2$ and size $n' = O(k^2 n^3)$. □

Theorem 2.3. COMPACT ATM COMPUTATION *is XP-complete.*

Proof. We show fp-equivalence with PEBBLE GAME, shown XP-complete in [DF99, 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 possible 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 $k = |S|$.

<u>COMPACT ATM COMPUTATION $\leq^{\text{fp}}_{\text{m}}$ PEBBLE GAME:</u>

[KAI79, 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.

<u>PEBBLE GAME $\leq_{\mathrm{m}}^{\mathrm{fp}}$ COMPACT ATM COMPUTATION:</u>
Given an instance $G = \langle N, S, T, R\rangle$ with $|S| = k$, one constructs an ATM M_G that emulates the game and accepts iff player I wins. The alphabet of M_G is N and k worktape 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_G. This gives an fp-reduction since $|M_G|$ is in $O(|G|)$ and $k' = k$. $\square$

3 Synchronized transition systems

A *labeled transition system* (LTS) $\mathcal{A}$ over some alphabet Σ is a tuple $\langle Q, \Sigma, \rightarrow\rangle$ where $Q = \{s, t, \ldots\}$ is the set of states and $\rightarrow \subseteq Q \times \Sigma \times Q$ are the 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 $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ of (flat) component LTSs. Assuming $\mathcal{A}_i = \langle Q_i, \Sigma_i, \rightarrow_i\rangle$ for $i = 1, \ldots, k$, the product 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.

In *strong synchronization* all components move at the same time: $\langle s_1, \ldots, s_k\rangle \xrightarrow{a}_{\text{str}} \langle t_1, \ldots, t_k\rangle$ iff $s_i \xrightarrow{a}_i t_i$ for all $i = 1, \ldots, k$.

In *binary synchronization* any two components synchronize while the rest does not move: $\langle s_1, \ldots, s_k\rangle \xrightarrow{a}_{\text{bin}} \langle t_1, \ldots, t_k\rangle$ 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 $l \notin \{i, j\}$.

In this paper, we consider *strong synchronization* as the natural model for non-flat systems and the notation $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ assumes strong synchronization when we don't explicitly say otherwise. However, our results are robust and remain unchanged when one adopts binary synchronization (see [DLS01, Appendix B]).

4 Parameterized complexity of non-flat reachability

Reachability problems are the most fundamental problems in model checking.

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 $\bar{s} \xrightarrow{*} \bar{t}$?

We are interested in the parameterized versions k-EXACT-REACH, where k is the parameter. Other variants of this problem are used in model checking, for instance by considering a finite set of target states instead of a unique state $\bar{t}$, or by asking for repeated reachability of a control state, that is we ask for $\bar{s} \xrightarrow{*} \bar{t} \xrightarrow{+} \bar{t}$. Finally, another standard variant consists in considering fair reachability. Details of the definition of these parameterized problems can be found in [DLS01]. It is a folklore result that the four non-flat reachability problems are equivalent in the classical sense (i.e. via logspace reductions) and are PSPACE-complete and we can show this equivalence can be lifted to the parameterized case [DLS01, Theorem 4.1].

Lemmas 4.2 and 4.3 allow the following characterization:

Theorem 4.1. *k-*Exact-Reach *is fp-equivalent to* Compact NDTM Computation.

Hence k-Exact-Reach and the above mentionned variants of k-Exact-Reach are AW[P]-hard. The characterization given by Theorem 4.1 is robust: it stays unchanged when we consider binary synchronization or when we restrict to a binary alphabet or to deterministic LTSs (see [DLS01, Appendix B & C]).

Lemma 4.2. Compact NDTM Computation $\leq^{\mathrm{fp}}_{\mathrm{m}}$ k-Exact-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 behaviour of M on a k-bounded tape. For $i = 1, \ldots, k$, $\mathcal{A}_i$ stores the current contents 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 an exact reachability criterion (we add an extra label for the final transition). Finally we translated our instance to a k-Exact-Reach instance with $k' = k + 2$ and $n' = O(kn^2)$. □

Lemma 4.3. k-Exact-Reach $\leq^{\mathrm{fp}}_{\mathrm{m}}$ 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 behaviour 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 $O(kn)$. □

5 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, HML, the modal μ-calculus (see [Eme90,CGP99,BS01]).

For modal logics, LTSs are the natural models, while for temporal logics like CTL or LTL 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 $|\mathcal{A}| + |m|$.

We omit the standard definition of when state s in $\mathcal{M}$ satisfies formula ϕ, written $\mathcal{M}, s \models \phi$. There is one detail though: for linear-time logics (LTL and its fragments) we follow [SC85] 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.

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 w.l.o.g. 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 problems we consider have the following general form, where L is LTL, CTL, the modal μ-calculus, or some of their fragments, and where the parameterized version has the pair $k, |\phi|$ as parameter:

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.
Question: Does $\mathcal{M}_1 \times \cdots \times \mathcal{M}_k, \bar{s} \models \phi$?

5.1 Linear time

LTL model checking for non-flat systems is PSPACE-complete. In our parameterized setting we have:

Theorem 5.1. *k, ϕ-$\mathrm{MC}_{\mathrm{LTL}}$ is fp-equivalent to* Compact NDTM Computation *and to k-*Exact-Reach *(and hence is AW[P]-hard).*

Proof. k-Exact-Reach reduces to k, ϕ-$\mathrm{MC}_{\mathrm{LTL}}$ since $\bar{s} \xrightarrow{*} \langle t_1, \ldots, t_k\rangle$ in some $\mathcal{A}_1 \times \ldots \times \mathcal{A}_k$ iff $\langle \mathcal{A}_1, \{\langle t_1, \mathrm{p}_1\rangle\}\rangle \times \ldots \langle \mathcal{A}_k, \{\langle t_k, \mathrm{p}_k\rangle\}\rangle, \bar{s} \models \mathsf{F}(\mathrm{p}_1 \wedge \ldots \wedge \mathrm{p}_k)$. 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 [3]. There remains to check that

[3] 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 [DLS01, Appendix B], the parametrized reachability problems for this form of synchronized products can also be proved fp-equivalent to Compact NDTM Computation.

this classical reduction is a reduction in the parameterized sense: since $|\mathcal{B}_\phi|$ is in $O(2^{|\phi|})$, the reduction has k' in $O(k)$ and n' in $O(2^k \times n)$, which is enough for fp-reducibility. □

Model checking is already intractable for the fragment LT0 of LTL *that does not allow to state reachability questions.* LT0 formulae are built with atomic propositions, X and $\vee$ only (no negation allowed):

Theorem 5.2. [DLS01] *k, ϕ-$\mathrm{MC}_{\mathrm{LT0}}$ is W[1]-complete, even if we restrict to formulae using only one atomic proposition.*

5.2 Branching time

Model checking non-flat systems is EXPTIME-complete for the μ-calculus [Rab00] and PSPACE-complete for HML or CTL. (Observe that HML, the fragment of the μ-calculus without fixed-points, does not allow stating reachability questions). In our parameterized setting we have:

Theorem 5.3. *k, ϕ-MC_μ is XP-complete.*

Proof. Writing n for $\sum_i |\mathcal{M}_i|$, k, ϕ-MC_μ can be solved in time $O\big((|\phi|.n^k)^{|\phi|}\big)$ [KVW00, Theo. 6.4] and hence is in XP.

XP-hardness is proved by a reduction from non-flat bisimilarity. Andersen [And93, section 4.3] showed how bisimilarity checking can be stated in the branching-time mu-calculus: consider two LTSs $\mathcal{A}$ and $\mathcal{B}$ on a common Σ, and build $\mathcal{B}'$ out of $\mathcal{B}$ by renaming all its actions $a \in \Sigma$ by copies $a' \notin \Sigma$. Then

$$\mathcal{A} \text{ and } \mathcal{B} \text{ are bisimilar iff } \mathcal{A} \parallel \mathcal{B}' \models \nu X. \bigwedge_{a \in \Sigma} ([a]\langle a' \rangle X \wedge [a']\langle a \rangle X). \tag{1}$$

The interleaved product $\mathcal{A} \parallel \mathcal{B}'$ can be replaced by strong synchronization if we add Σ-loops in all states of $\mathcal{B}'$ and Σ'-loops in all states of $\mathcal{A}$. Using $(\mathcal{A}_1 \times \mathcal{A}_2)' = \mathcal{A}_1' \times \mathcal{A}_2'$, the reduction carries to non-flat systems.

Since non-flat bisimilarity is XP-hard already when $|\Sigma| = 2$ [DLS01, Theorem D.4], we can bound the size of the μ-formula in (1) and have an fp-reduction. □

Theorem 5.4. *k, ϕ-$\mathrm{MC}_{\mathrm{HML}}$ is 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 5.2 to ATMs. □

Theorem 5.5. Compact NDTM Computation $\leq_{\mathrm{m}}^{\mathrm{fp}}$ k, ϕ-$\mathrm{MC}_{\mathrm{CTL}}$. *(Hence k, ϕ-$\mathrm{MC}_{\mathrm{CTL}}$ is AW[P]-hard).*

Proof (idea). CTL allows to state reachability questions. □

Remark 5.6. For the moment, we do not have a more precise characterization for k, ϕ-$\mathrm{MC}_{\mathrm{CTL}}$. Observe that, by definition, model checking CTL is closed by complementation (unlike the LTL case) so that a conjecture about their 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[P].

6 Parameterized complexity of non-flat bisimilarity

We assume familiarity with bisimulation and the other behavioral equivalences in the branching time – linear time spectrum [Gla01]. Checking for bisimilarity among non-flat systems is EXPTIME-complete in the classical framework [JM96,LS00]. For our parametric analysis, k-Bisim asks whether two given configurations in a product of k LTSs are bisimilar.

Theorem 6.1. k-Bisim *is XP-complete.*

Proof (idea). k-Bisim is in XP since bisimilarity of flat systems is polynomial-time [KS90]. XP-hardness is seen by observing that the reduction in the proof of [LS00, Theorem 4.1] can be seen as an fp-reduction from Compact ATM Computation to k-Bisim. □

This result is robust and [DLS01, Appendix D] shows that it still holds when we consider binary synchronization or restricted alphabets (a result we used in the proof of Theorem 5.3). [DLS01] further proves XP-hardness for a wide range of behavioral equivalences and preorders, along the lines of [Rab97,LS00,SJ01].

7 Conclusion

We studied the complexity of model checking synchronized products of LTSs under the light of Downey and Fellows's theory of parameterized complexity. Here 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 Downey and Fellows's hierarchy collapses). Therefore our results show that these problems are probably not tractable *even in the parameterized sense of being FPT*, and their complexity is in general quite high in the hierarchy (see summary in Fig. 1 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 less than the size of the components themselves.

References

[ADF95] K. A. Abrahamson, R. G. Downey, and M. R. Fellows. Fixed-parameter tractability and completeness IV: On completeness for W[P] and PSPACE analogs. *Annals of Pure and Applied Logic*, 73(3):235–276, 1995.

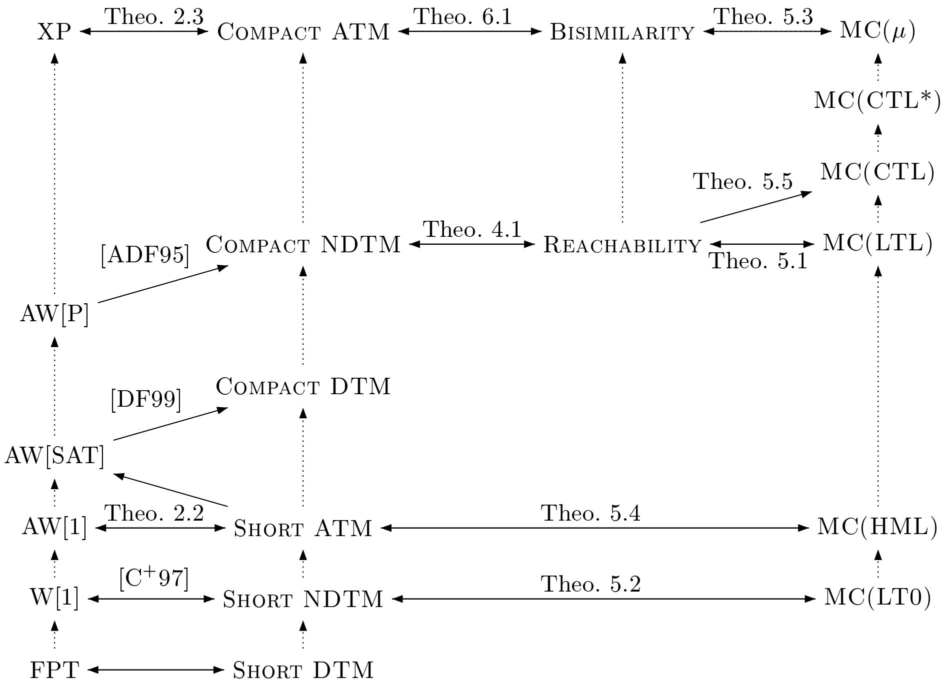

Fig. 1. A summary of existing reductions between parameterized problems

[And93] H. R. Andersen. *Verification of Temporal Properties of Concurrent Systems.* PhD thesis, Aarhus University, Denmark, June 1993.

[B+95] 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.

[BS01] J. Bradfield and C. Stirling. Modal logics and mu-calculi: an introduction. In *Handbook of Process Algebra*, ch 4, pp 293–330. Elsevier Science, 2001.

[C+97] Liming Cai, Jianer 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.

[Ces01] M. Cesati. The Turing way to the parameterized intractability, September 2001. Submitted.

[CGP99] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking.* MIT Press, 1999.

[DF99] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer, 1999.

[DLS01] S. Demri, F. Laroussinie, and Ph. Schnoebelen. A parametric analysis of the state explosion problem in model checking. Research Report LSV-01-4, Lab. Specification and Verification, ENS de Cachan, France, Apr. 2001.

[Eme90] E. A. Emerson. Temporal and modal logic. In *Handbook of Theoretical Computer Science, vol. B*, ch 16, pp 995–1072. Elsevier Science, 1990.

[Esp98] J. Esparza. Decidability and complexity of Petri net problems — an introduction. In *Advances in Petri Nets 1998*, *LNCS* 1491, pp 374–428. Springer, 1998.

[Gla01] R. J. van Glabbeek. The linear time – branching time spectrum I. In *Handbook of Process Algebra*, ch 1, pp 3–99. Elsevier Science, 2001.

[Gro99] M. Grohe. Descriptive and parameterized complexity. In *Computer Science Logic (CSL'99)*, *LNCS* 1683, pp 14–31. Springer, 1999.

[GSS01] M. Grohe, T. Schwentick, and L. Segoufin. When is the evaluation of conjunctive queries tractable? In *33rd ACM Symp. Theory of Computing (STOC'01)*, pp 657–666, 2001.

[HKV97] D. Harel, O. Kupferman, and M. Y. Vardi. On the complexity of verifying concurrent transition systems. In *Concurrency Theory (CONCUR'97)*, *LNCS* 1243, pp 258–272. Springer, 1997.

[JM96] L. Jategaonkar and A. R. Meyer. Deciding true concurrency equivalences on safe, finite nets. *Theoretical Computer Science*, 154(1):107–143, 1996.

[KAI79] T. Kasai, A. Adachi, and S. Iwata. Classes of pebble games and complete problems. *SIAM J. Comput.*, 8(4):574–586, 1979.

[KS90] 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.

[KVW00] O. Kupferman, M. Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. *J. ACM*, 47(2):312–360, 2000.

[LP85] O. Lichtenstein and A. Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *12th ACM Symp. Principles of Programming Languages (POPL'85)*, pp 97–107, 1985.

[LS00] F. Laroussinie and Ph. Schnoebelen. The state explosion problem from trace to bisimulation equivalence. In *Found. Software Science & Computation Structures (FOSSACS'2000)*, *LNCS* 1784, pp 192–207. Springer, 2000.

[PY99] C. H. Papadimitriou and M. Yannakakis. On the complexity of database queries. *J. Computer and System Sciences*, 58(3):407–427, 1999.

[Rab97] A. Rabinovich. Complexity of equivalence problems for concurrent systems of finite agents. *Information and Computation*, 139(2):111–129, 1997.

[Rab00] A. Rabinovich. Symbolic model checking for μ-calculus requires exponential time. *Theoretical Computer Science*, 243(1–2):467–475, 2000.

[SC85] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749, 1985.

[SJ01] Z. Sawa and P. Jančar. P-hardness of equivalence testing on finite-state processes. In *Current Trends in Theory and Practice of Informatics (SOFSEM'01)*, *LNCS* 2234, pp 326–335. Springer, 2001.

[War01] H. T. Wareham. The parameterized complexity of intersection and composition operations on sets of finite-state automata. In *Implementation & Application of Automata (CIAA'2000)*, *LNCS* 2088, pp 302–310. Springer, 2001.