Proceedings, Fifteenth International Conference on
Principles of Knowledge Representation and Reasoning (KR 2016)

Parameterized Complexity Results for Symbolic Model Checking of Temporal Logics

Ronald de Haan and **Stefan Szeider**
Algorithms and Complexity Group
TU Wien, Vienna, Austria
[dehaan,sz]@ac.tuwien.ac.at

Abstract

Reasoning about temporal knowledge is a fundamental task in the area of artificial intelligence and knowledge representation. A key problem in this area is model checking, and indispensable for the state-of-the-art in solving this problem in large-scale settings is the technique of bounded model checking. We investigate the theoretical possibilities of this technique using parameterized complexity theory. In particular, we provide a complete parameterized complexity classification for the model checking problem for symbolically represented Kripke structures for various fragments of the temporal logics LTL, CTL and CTL*. We argue that a known result from the literature for a restricted fragment of LTL can be seen as an fpt-reduction to SAT, and show that such reductions are not possible for any of the other fragments of the temporal logics that we consider. As a by-product of our investigation, we develop a novel parameterized complexity class that can be seen as a parameterized variant of the Polynomial Hierarchy.

Introduction

Temporal reasoning is an important part of knowledge representation and reasoning, and of artificial intelligence in general, and has applications for topics such as agent-based systems and planning (Fisher, Gabbay, and Vila 2005). For example, temporal logics can be used to express desired behavior of agents in multi-agent systems. A core problem related to temporal logics is the problem of model checking. (see, e.g., Fisher 2008). The problem consists of checking whether a model, given in the form of a labelled relational structure (a *Kripke structure*), satisfies a temporal property, given as a logic formula. Underlining the importance of temporal logic model checking, the ACM 2007 Turing Award was given for foundational research on the topic (Clarke, Emerson, and Sifakis 2009). Indispensable for the state-of-the-art in solving this problem in industrial-size settings is the algorithmic technique of symbolic model checking using propositional satisfiability (SAT) solvers (called *bounded model checking*), where the SAT solvers are employed to find counterexamples (Biere 2009; Biere et al. 2003; 1999; Clarke et al. 2004). A theoretical analysis identifying the cases where this vital technique can be employed is critical for improving state-of-the-art model checking algorithms. Yet no such structured complexity-theoretic investigation has been done, leaving a significant gap in the guidance offered to practitioners developing model checking algorithms.

The approach of bounded model checking generally works well in cases where the Kripke structure is large, but the temporal logic specification is small. Since the framework of parameterized complexity is able to distinguish an additional measure of the input, that can be much smaller than the input size, a parameterized complexity approach would be especially suited for the much-needed theoretical analysis. However, previous parameterized complexity analyses have not been able to fill the gap. First of all, existing parameterized complexity analyses (Demri, Laroussinie, and Schnoebelen 2006; Flum and Grohe 2006; Göller 2013; Lück, Meier, and Schindler 2015) have only considered the problem for settings where the Kripke structure is spelled-out explicitly (or consists of a small number of explicitly spelled-out components), which is highly impractical in many cases. In fact, the so-called state explosion problem is a major obstacle for developing practically useful techniques (Clarke et al. 2001). For this reason, the Kripke structures are often described symbolically, for instance using propositional formulas, which allows for exponentially more succinct encodings of the structures. Secondly, whereas parameterized complexity analysis is traditionally focused on fixed-parameter tractability for positive results, the technique of bounded model checking revolves around encoding the problem as an instance of SAT. Therefore, the standard parameterized complexity analysis is bound to concentrate on very restrictive cases in order to obtain fixed-parameter tractability, unaware of some of the more liberal settings where bounded model checking can be applied.

In this paper, we contribute to closing the gap by means of a more advanced parameterized complexity analysis that reveals the possibilities and limits of the technique of bounded model checking. More specifically, we analyze the complexity of the model checking problem for fragments of various temporal logics, where we take the size of the temporal logic formula as parameter. In our formalization of the problem, the Kripke structures are represented symbolically (and can thus be of size exponential in the size of their description). Moreover, our complexity analysis focuses on whether a reduction of the model checking problem to SAT is fixed-parameter tractable or not, rather than whether the model checking problem itself is fixed-parameter tractable. Such fixed-parameter tractable reductions (fpt-reductions, for short) to SAT can be used in many cases where polynomial-time reductions cannot be used (De Haan and Szeider 2014a; 2014b). For instance, fpt-reductions have been used to reduce problems related to answer set programming and abductive reasoning, which are located at the second level of the Polynomial Hierarchy, to SAT (Fichte and Szeider 2013; Pfandler, Rümmele, and Szeider 2013).

Contributions We consider the model checking problem for three of the most widespread temporal logics, LTL, CTL and CTL*. These are linear-time and branching-time propositional modal logics that lie at the basis of more expressive formalisms used in many settings in knowledge representation and reasoning (see, e.g., Fisher 2008). Moreover, for each of these logics, we consider also the fragments where several temporal operators (namely, U and/or X) are disallowed. Using non-standard parameterized complexity methods, we give a complete complexity classification of the problem of checking whether a given Kripke structure, specified symbolically using a propositional formula, satisfies a given temporal logic specification, parameterized by the size of the temporal logic formula.

- We show that the problem is para-PSPACE-complete for all logics and all fragments if the recurrence diameter of the structure (the size of the largest loop-free path) is allowed to be exponentially large (Proposition 2).

Due to this result that without bounds on the recurrence diameter of the structure the problem is intractable even in the simplest cases, we direct our attention to the setting where the recurrence diameter is polynomially bounded.

- We identify a known result from the literature on bounded model checking (Kroening et al. 2011) as a para-co-NP-membership result for the logic LTL where both operators U and X are disallowed, and we extend this to a completeness result (Proposition 3).
- We show that the problem is para-PSPACE-complete for LTL (and so also for CTL*) when at least one of the operators U and X is allowed (Theorems 6 and 7).
- We show that in all remaining cases (all fragments of CTL, and the fragment of CTL* without the operators U and X) the problem is complete for PH(LEVEL) (Theorems 9 and 10).

The prime difficulty for the latter completeness results was to identify the parameterized complexity class PH(LEVEL), and to characterize it in various ways. The main technical obstacle that we had to overcome to show para-PSPACE-hardness was to encode satisfiability of quantified Boolean formulas without having explicit quantification available in the logic.

In short, we show that the only case (given these fragments of temporal logics) where the technique of bounded model checking can be applied is the fragment of LTL without the operators U and X. An overview of all the completeness results that we develop in this paper can be found in Table 1.

In addition, as mentioned, we introduce the parameterized complexity class PH(LEVEL), which is based on the satisfiability problem of quantified Boolean formulas parameterized by the number of quantifier alternations. We show that this class can also be characterized by means of an analogous parameterized version of first-order logic model checking, as well as by alternating Turing machines that alternate between existential and universal configurations only a small number of times (depending only on the parameter).

Related work Computational complexity analysis has been a central aspect in the study of the model checking problem for temporal logics (Emerson and Lei 1987; Kupferman, Vardi, and Wolper 2000; Sistla and Clarke 1985; Vardi and Wolper 1986). Naturally this problem has been analyzed from a parameterized complexity point of view. For instance, LTL model checking parameterized by the size of the logic formula features as a textbook example for fixed-parameter tractability (Flum and Grohe 2006). For the temporal logic CTL, parameterized complexity has also been used to study the problems of model checking and satisfiability (Demri, Laroussinie, and Schnoebelen 2006; Göller 2013; Lück, Meier, and Schindler 2015). As the SAT encoding techniques used for bounded LTL model checking yield an incomplete solving method in general, limits on the cases in which this particular encoding can be used as a complete solving method have been studied (Bundala, Ouaknine, and Worrell 2012; Clarke et al. 2004; Kroening et al. 2011).

Outline We begin with reviewing relevant notions from (parameterized) complexity theory. Then, we introduce the different temporal logics that we consider, we review known complexity results for their model checking problems, and we interpret a known result for bounded model checking for the fragment of LTL without U and X operators using notions from parameterized complexity. Next, we introduce the new parameterized complexity class PH(LEVEL). Finally, we provide the parameterized complexity results that indicate that bounded model checking cannot be applied efficiently for all other fragments of the temporal logics that we consider.

Due to space restrictions, we omit (technical details in) the proofs of several results. Results for which a proof is entirely omitted, we mark with an asterisk. Full detailed proofs can be found in a recent technical report (De Haan and Szeider 2015).

Preliminaries

Polynomial Space The class PSPACE consists of all decision problems that can be solved by an algorithm that uses a polynomial amount of space (memory). Alternatively, one can characterize the class PSPACE as all decision problems for which there exists a polynomial-time reduction to the problem QSAT, that is defined using quantified Boolean formulas as follows. A quantified Boolean formula (in prenex form) is a formula of the form $Q_1 x_1 Q_2 x_2 \dots Q_n x_n.\psi$, where all x_i are propositional variables, each Q_i is either an existential or a universal quantifier, and ψ is a (quantifier-free)

logic $\mathcal{L}$	LTL	CTL	CTL*
$\mathcal{L}$	para-PSPACE-complete	PH(LEVEL)-complete	para-PSPACE-complete
$\mathcal{L}\backslash$X	para-PSPACE-complete	PH(LEVEL)-complete	para-PSPACE-complete
$\mathcal{L}\backslash$U	para-PSPACE-complete	PH(LEVEL)-complete	para-PSPACE-complete
$\mathcal{L}\backslash$U,X	para-co-NP-complete	PH(LEVEL)-complete	PH(LEVEL)-complete

Table 1: Parameterized complexity results for the problem SYMBOLIC-MC$^\star[\mathcal{L}]$ for the different (fragments of) logics $\mathcal{L}$. In this problem, the recurrence diameter of the structure is polynomially bounded. The problem SYMBOLIC-MC$[\mathcal{L}]$, where the recurrence diameter is unbounded, is para-PSPACE-complete in all cases.

propositional formula over the variables $x_1, \dots, x_n$ (called the *matrix*). Truth for such formulas is defined in the usual way. The problem QSAT consists of deciding whether a given quantified Boolean formula is true.

Alternatively, the semantics of quantified Boolean formulas can be defined using QBF models (Samulowitz, Davies, and Bacchus 2006). Let $\varphi = Q_1 x_1 \dots Q_n x_n.\psi$ be a quantified Boolean formula. A *QBF model* for φ is a tree of (partial) truth assignments where (1.) each truth assignment assigns values to the variables $x_1, \dots, x_i$ for some $1 \leq i \leq n$, (2.) the root is the empty assignment, and for all assignments α in the tree, assigning truth values to the variables $x_1, \dots, x_i$ for some $1 \leq i \leq n$, the following conditions hold: (3.) if $i < n$, every child of α agrees with α on the variables $x_1, \dots, x_i$, and assigns a truth value to x_{i+1} (and to no other variables); (4.) if $i = n$, then α satisfies ψ, and α has no children; (5.) if $i < n$ and $Q_i = \exists$, then α has one child α' that assigns some truth value to x_{i+1}; and (6.) if $i < n$ and $Q_i = \forall$, then α has two children α_1 and α_2 that assign different truth values to x_{i+1}. It is straightforward to show that a quantified Boolean formula φ is true if and only if there exists a QBF model for φ. Note that this definition of QBF models is a special case of the original definition (Samulowitz, Davies, and Bacchus 2006).

Fixed-parameter tractable reductions to SAT We assume the reader to be familiar with basic notions from parameterized complexity theory, such as fixed-parameter tractability and fpt-reductions. For instance, a parameterized problem is a subset $L \subseteq \Sigma^* \times \mathbb{N}$ and is said to be *fixed-parameter tractable* if there is a computable function f, a constant c, and an algorithm that for each $(x, k) \in \Sigma^* \times \mathbb{N}$ decides whether $(x, k) \in L$ in time $f(k)n^c$. For more details, we refer to textbooks on the topic (Downey and Fellows 2013; Flum and Grohe 2006; Niedermeier 2006). We briefly highlight some notions that are useful for investigating fpt-reductions to SAT. The propositional satisfiability problem (SAT) consists of deciding whether a given propositional formula in CNF is satisfiable. When we consider SAT as a parameterized problem, we consider the trivial (constant) parameterization. The parameterized complexity class para-NP consists of all problems that can be fpt-reduced to SAT. More generally, for each non-parameterized complexity class K, the parameterized class para-K is defined as the class of all parameterized problems $L \subseteq \Sigma^* \times \mathbb{N}$, for which there exist a computable function $f : \mathbb{N} \to \Sigma^*$, and a problem $P \subseteq \Sigma^* \times \Sigma^*$ such that $P \in$ K and for all instances $(x, k) \in \Sigma^* \times \mathbb{N}$ of L we have that $(x, k) \in L$ if and only if $(x, f(k)) \in P$ (Flum and Grohe 2003). Intuitively, the class para-K consists of all problems that are in K after a precomputation that only involves the parameter. For all classical complexity classes K, K$'$ it holds that K $\subseteq$ K$'$ if and only if para-K $\subsetneq$ para-K$'$. Therefore, in particular, problems that are para-PSPACE-hard are not in para-NP, unless NP $=$ PSPACE.

Model Checking for Temporal Logics

In this section, we review the definition of the temporal logics that we consider in this paper, and we introduce the problem of model checking for symbolically represented Kripke structures. In addition, we argue why the polynomial bound on the recurrence diameter of the Kripke structures is necessary to obtain an fpt-reduction to SAT. Finally, we identify a para-co-NP-membership result from the literature on bounded model checking.

Temporal Logics

We begin with defining the semantical structures for all temporal logics. In the remainder of the paper, we let P be a finite set of propositions. A *Kripke structure* is a tuple $\mathcal{M} = (S, R, V, s_0)$, where S is a finite set of *states*, $R \subseteq S \times S$ is a binary relation on the set of states called the *transition relation*, $V : S \to 2^P$ is a *valuation function* that assigns each state to a set of propositions, and where $s_0 \in S$ is the *initial state*. An example of a Kripke structure is given in Figure 1. We say that a finite sequence $s_1 \dots s_\ell$ of states $s_i \in S$ is a *finite path* in $\mathcal{M}$ if $(s_i, s_{i+1}) \in R$ for each $1 \leq i < \ell$. Similarly, we say that an infinite sequence $s_1 s_2 s_3 \dots$ of states $s_i \in S$ is an *infinite path* in $\mathcal{M}$ if $(s_i, s_{i+1}) \in R$ for each $i \geq 1$.

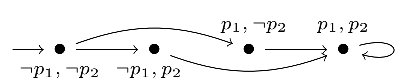

Figure 1: An example Kripke structure $\mathcal{M}_1$ for the set $P = \{p_1, p_2\}$ of propositions.

Now, we can define the syntax of the logic LTL. LTL formulas over the set P of atomic propositions are formed according to the following grammar (here p ranges over P), given by $\varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \mathrm{X}\varphi \mid \mathrm{F}\varphi \mid (\varphi \mathrm{U} \varphi)$. We consider the usual abbreviations, such as $\varphi_1 \vee \varphi_2 = \neg(\neg\varphi_1 \wedge \neg\varphi_2)$. In addition, we let the abbreviation Gφ denote $\neg$F$\neg\varphi$. Intuitively, the formula Xφ expresses that φ is true in the next (time) step, Fφ expresses that φ becomes true at some point in the future, Gφ expresses that φ is true at all times from now on, and φ_1Uφ_2 expresses that φ_2 becomes true at some point in time, and until then the formula φ_1 is true at all points. Formally, the semantics of LTL formulas are defined for Kripke structures, using the notion of (infinite) paths. Let $\mathcal{M} = (S, R, V, s_0)$ be a Kripke structure, and $\overline{s}_1 = s_1 s_2 s_3 \dots$ be a path in $\mathcal{M}$. Moreover, let $\overline{s}_i = s_i s_{i+1} s_{i+2} \dots$ for each $i \geq 2$. Truth of LTL formulas φ on paths $\overline{s}$ (denoted $\overline{s} \models \varphi$) is defined inductively as follows (for the sake of brevity, we omit the straightforward Boolean cases):

$\overline{s}_i \models \mathrm{X}\varphi$	if	$\overline{s}_{i+1} \models \varphi$
$\overline{s}_i \models \mathrm{F}\varphi$	if	for some $j \geq 0$, $\overline{s}_{i+j} \models \varphi$
$\overline{s}_i \models \varphi_1 \mathrm{U} \varphi_2$	if	there is some $j \geq 0$ such that $\overline{s}_{i+j} \models \varphi_2$ and $\overline{s}_{i+j'} \models \varphi$ for each $0 \leq j' < j$

Then, we say that an LTL formula φ is true in the Kripke structure $\mathcal{M}$ (denoted $\mathcal{M} \models \varphi$) if for all infinite paths $\overline{s}$ starting in s_0 it holds that $\overline{s} \models \varphi$. For instance, considering the example $\mathcal{M}_1$ from Figure 1, it holds that $\mathcal{M}_1 \models \mathrm{FG}p_2$.

Next, we can define the syntax of the logic CTL*, which consists of two different types of formulas: state formulas and path formulas. When we refer to CTL* formulas without specifying the type, we refer to state formulas. Given the set P of atomic propositions, the syntax of CTL* formulas is defined by the following grammar (here Φ denotes CTL* *state formulas*, φ denotes CTL* *path formulas*, and p ranges over P), given by $\Phi ::= p \mid \neg\Phi \mid (\Phi \wedge \Phi) \mid \exists\varphi$, and $\varphi ::= \Phi \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \mathrm{X}\varphi \mid \mathrm{F}\varphi \mid (\varphi \mathrm{U} \varphi)$. Again, we consider the usual abbreviations, such as $\varphi_1 \vee \varphi_2 = \neg(\neg\varphi_1 \wedge \neg\varphi_2)$, for state formulas as well as for path formulas. Moreover, we let the abbreviation Gφ denote $\neg$F$\neg\varphi$, and we let the abbreviation $\forall\varphi$ denote $\neg\exists\neg\varphi$. Path formulas have the same intended meaning as LTL formulas. State formulas, in addition, allow quantification over paths, which is not possible in LTL.

Formally, the semantics of CTL* formulas are defined inductively as follows. Let $\mathcal{M} = (S, R, V, s_0)$ be a Kripke structure, $s \in S$ be a state in $\mathcal{M}$ and $\overline{s}_1 = s_1 s_2 s_3 \dots$ be a path in $\mathcal{M}$. Again, let $\overline{s}_i = s_i s_{i+1} s_{i+2} \dots$ for each $i \geq 2$. The truth of CTL* state formulas Φ on states s (denoted $s \models \Phi$) is defined as follows (again, we omit the Boolean cases): $s \models \exists\varphi$ if and only if there is some path $\overline{s}$ in $\mathcal{M}$ starting in s such that $\overline{s} \models \varphi$. The truth of CTL* path formulas φ on paths $\overline{s}$ (denoted $\overline{s} \models \varphi$) is defined as follows:

$\overline{s}_i \models \mathrm{X}\varphi$	if	$\overline{s}_{i+1} \models \varphi$
$\overline{s}_i \models \mathrm{F}\varphi$	if	for some $j \geq 0$, $\overline{s}_{i+j} \models \varphi$
$\overline{s}_i \models \varphi_1 \mathrm{U} \varphi_2$	if	there is some $j \geq 0$ such that $\overline{s}_{i+j} \models \varphi_2$ and $\overline{s}_{i+j'} \models \varphi$ for each $0 \leq j' < j$

Then, we say that a CTL* formula Φ is true in the Kripke structure $\mathcal{M}$ (denoted $\mathcal{M} \models \Phi$) if $s_0 \models \Phi$. For example, again taking the structure $\mathcal{M}_1$, it holds that $\mathcal{M}_1 \models \exists(\mathrm{X}p_1 \wedge \forall\mathrm{GXX}p_2)$.

Next, the syntax of the logic CTL is defined similarly to the syntax of CTL*. Only the grammar for path formulas φ differs, namely $\varphi ::= \mathrm{X}\Phi \mid \mathrm{F}\Phi \mid (\Phi \mathrm{U} \Phi)$. In particular, this means that every CTL state formula, (CTL formula for short) is also a CTL* formula. The semantics for CTL formulas is defined as for their CTL* counterparts.

For each of the logics $\mathcal{L} \in \{\mathrm{LTL}, \mathrm{CTL}, \mathrm{CTL}^\star\}$, we consider the fragments $\mathcal{L}\backslash$X, $\mathcal{L}\backslash$U and $\mathcal{L}\backslash$U,X. In the fragment $\mathcal{L}\backslash$X, the X-operator is disallowed. Similarly, in the fragment $\mathcal{L}\backslash$U, the U-operator is disallowed. In the fragment $\mathcal{L}\backslash$U,X, neither the X-operator nor the U-operator is allowed. Note that the logic LTL$\backslash$X is also known as UTL, and the logic LTL$\backslash$U,X is also known as UTL$\backslash$X (see, e.g., Kroening et al. 2011).

We review some known complexity results for the model checking problem of the different temporal logics. Formally, we consider the problem MC[$\mathcal{L}$], for each of the temporal logics $\mathcal{L}$, where the input is a Kripke structure $\mathcal{M}$ and an $\mathcal{L}$ formula φ, and the question is to decide whether $\mathcal{M} \models \varphi$. Note that in this problem the Kripke structure $\mathcal{M}$ is given explicitly in the input. As parameter, we will always take the size of the logic formula. It is well known that the problems MC[LTL] and MC[CTL*] are PSPACE-complete, and that the problem MC[CTL] is polynomial-time solvable (see, e.g., Baier and Katoen 2008). It is also well known that the problems MC[LTL] and MC[CTL*] are fixed-parameter tractable when parameterized by the size of the logic formula (see, e.g., Baier and Katoen 2008; Flum and Grohe 2006).

Symbolically Represented Kripke Structures

A challenge occurring in practical verification settings is that the Kripke structures are too large to handle. Therefore, these Kripke structures are often not written down explicitly, but rather represented symbolically by encoding them succinctly using propositional formulas.

Let $P = \{p_1, \dots, p_m\}$ be a finite set of propositional variables. A *symbolically represented Kripke structure* over P is a tuple $\mathcal{M} = (\varphi_R, \alpha_0)$, where $\varphi_R(x_1, \dots, x_m, x'_1, \dots, x'_m)$ is a propositional formula over the variables $x_1, \dots, x_m$, $x'_1, \dots, x'_m$, and where $\alpha_0 \in \{0,1\}^m$ is a truth assignment to the variables in P. The Kripke structure associated with $\mathcal{M}$ is (S, R, V, α_0), where $S = \{0,1\}^m$ consists of all truth assignments to P, where $(\alpha, \alpha') \in R$ if and only if $\varphi_R[\alpha, \alpha']$ is true, and where $V(\alpha) = \{\, p_i : \alpha(p_i) = 1 \,\}$.

Example 1. Let $P = \{p_1, p_2\}$. The Kripke structure $\mathcal{M}_1$ from Figure 1 can be symbolically represented by (φ_R, α_0), where $\varphi_R(x_1, x_2, x'_1, x'_2) = [(\neg x_1 \wedge \neg x_2) \rightarrow (\neg x'_1 \leftrightarrow x'_2)] \wedge [(\neg x_1 \leftrightarrow x_2) \rightarrow (x'_1 \wedge x'_2)] \wedge [(x_1 \wedge x_2) \rightarrow (x'_1 \wedge x'_2)]$, and $\alpha_0 = (0, 0)$. ⊣

We can now consider the symbolic variant SYMBOLIC-MC[$\mathcal{L}$] of the model checking problem, for each of the temporal logics $\mathcal{L}$.

> SYMBOLIC-MC[$\mathcal{L}$]
> *Input:* a symbolically represented Kripke structure $\mathcal{M}$, and an $\mathcal{L}$ formula φ.
> *Question:* $\mathcal{M} \models \varphi$?

Similarly to the case of MC[$\mathcal{L}$], we will also consider SYMBOLIC-MC[$\mathcal{L}$] as a parameterized problem, where the parameter is $|\varphi|$. Interestingly, for the logics LTL and CTL*, the complexity of the model checking problem does not change when Kripke structures are represented symbolically: SYMBOLIC-MC[LTL] and SYMBOLIC-MC[CTL*] are PSPACE-complete (see Kupferman, Vardi, and Wolper 2000; Vardi and Wolper 1986). However, for the logic CTL, the complexity of the problem does show an increase. In fact, the problem is already PSPACE-hard for very simple formulas.

Proposition* 2. SYMBOLIC-MC[LTL] *is* PSPACE-*hard even when restricted to the case where* $\varphi = \mathrm{G}p$. SYMBOLIC-MC[CTL] *and* SYMBOLIC-MC[CTL*] *are* PSPACE-*hard even when restricted to the case where* $\varphi = \forall \mathrm{G}p$.

An Fpt-Reduction to SAT for LTL\U,X

The result of Proposition 2 seems to indicate that the model checking problem for the temporal logics LTL, CTL and CTL* is intractable when Kripke structures are represented symbolically, even when the logic formulas are extremely simple. However, in the literature further restrictions have been identified that allow the problem to be solved by means of an encoding into SAT, which allows the use of practically very efficient SAT solving methods. In the hardness proof of Proposition 2, the Kripke structure has only a single path, which contains exponentially many different states. Intuitively, such exponential-length paths may be the cause of PSPACE-hardness. To circumvent this source of hardness, and to go towards the mentioned setting where the problem can be solved by means of a SAT encoding, we need to restrict the recurrence diameter. The *recurrence diameter* $rd(\mathcal{M})$ of a Kripke structure $\mathcal{M}$ is the length of the longest simple (non-repeating) path in $\mathcal{M}$. We consider the following variant of SYMBOLIC-MC[$\mathcal{L}$], where the recurrence diameter of the Kripke structures is restricted.[1]

> SYMBOLIC-MC*[$\mathcal{L}$]
> *Input:* a symbolically represented Kripke structure $\mathcal{M}$, $rd(\mathcal{M})$ in unary, and an $\mathcal{L}$ formula φ.
> *Parameter:* $|\varphi|$.
> *Question:* $\mathcal{M} \models \varphi$?

This restricted setting has been studied by Kroening et al. (Kroening et al. 2011). In particular, they showed that the model checking problem for LTL\U,X allows an encoding into SAT that is linear in $rd(\mathcal{M})$, even when the Kripke structure $\mathcal{M}$ is represented symbolically, and can thus be of exponential size. Using the result of Kroening et al., we obtain para-co-NP-completeness.

Proposition 3. SYMBOLIC-MC*[LTL\U,X] *is* para-co-NP-*complete.*

Proof (sketch). Kroening et al. (2011) use the technique of bounded model checking (Biere 2009; Biere et al. 1999; Clarke et al. 2004), where SAT solvers are used to find a 'lasso-shaped' path in a Kripke structure that satisfies an LTL formula φ. They show that for LTL\U,X formulas, the largest possible length of such lasso-shaped paths that needs to be considered (also called the *completeness threshold*) is linear in $rd(\mathcal{M})$. However, the completeness threshold depends linearly on the size of a particular type of generalized Büchi automaton expressing φ, which in general is exponential in the size of φ. Therefore, this SAT encoding does not run in polynomial time, but it does run in fixed-parameter tractable time when the size of φ is the parameter. Their encoding of the problem of finding a counterexample into SAT can be seen as an encoding of the model checking problem into UNSAT.

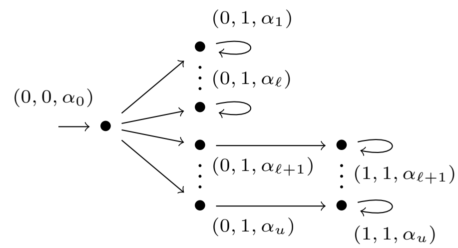

Figure 2: (The reachable part of) the structure $\mathcal{M}$ in the proof of Proposition 3.

We show para-co-NP-hardness by showing that the problem SYMBOLIC-MC*[LTL\U,X] is co-NP-hard already for formulas of constant size. We do so by a reduction from UNSAT. Let ψ be a propositional formula over the variables $x_1, \dots, x_n$. We construct an instance of SYMBOLIC-MC*[LTL\U,X] as follows. We consider the set $P = \{y_0, y_1, x_1, \dots, x_n\}$ of propositional variables. We construct the symbolically represented Kripke structure $\mathcal{M} = (\varphi_R, \alpha_0)$ as depicted in Figure 2. Here $\alpha_0 = \overline{0}$, i.e., the all-zeroes assignment to Var(ψ). Moreover, $\alpha_1, \dots, \alpha_\ell$ are the assignments to Var(ψ) that satisfy ψ, and $\alpha_{\ell+1}, \dots, \alpha_u$ are the assignments to Var(ψ) that falsify ψ. The transition relation given by φ_R allows a transition from α_0 to the state $(0, 1, \alpha)$ for any truth assignment α to the variables $x_1, \dots, x_n$. Then, if this assignment α satisfies ψ, a transition is allowed to the looping state $(1, 1, \alpha)$. Otherwise, if α does not satisfy ψ, the only transition from state $(0, 1, \alpha)$ is to itself. For a detailed description of φ_R, we refer to the technical report (De Haan and Szeider 2015). Finally, we define the LTL formula to be $\varphi = \mathrm{G}\neg y_0$.

Moreover, $rd(\mathcal{M}) = 2$, and the LTL formula φ is of constant size, and contains only the temporal operator G. It is straightforward to verify that $\mathcal{M} \models \varphi$ if and only if ψ is unsatisfiable. ☐

In the remainder of this paper, we give parameterized complexity results that give evidence that this is the only case in this setting where such an fpt-reduction to SAT is possible. In order to do so, we first make a little digression to introduce a new parameterized complexity class, that can be seen as a parameterized variant of the Polynomial Hierarchy (PH).

A Parameterized Variant of the PH

In order to completely characterize the parameterized complexity of the problems SYMBOLIC-MC*[$\mathcal{L}$], we need to introduce another parameterized complexity class, that is a parameterized variant of the PH. The PH consists of an infinite hierarchy of classes Σ_i^{p} and Π_i^{p} (see, e.g., Arora and Barak 2009, Chapter 5). For each $i \geq 0$, the complexity

[1] An equivalent way of phrasing the problem is to require that the recurrence diameter of the Kripke model $\mathcal{M}$ is polynomial in the size of its description (φ_R, α_0).

class Σ_i^{p} consists of closure of the problem QSAT_i under polynomial-time reductions, where QSAT_i is the restriction of the problem QSAT where the input formula starts with an existential quantifier and contains at most i quantifier alternations. The class Π_i^{p} is defined as co-Σ_i^{p}.

In other words, for each level of the PH, the number of quantifier alternations is bounded by a constant. If we allow an unbounded number of quantifier alternations, we get the complexity class PSPACE (see, e.g., Arora and Barak 2009, Theorem 5.10). Parameterized complexity theory allows a middle way: neither letting the number of quantifier alternations be bounded by a constant, nor removing all bounds on the number of quantifier alternations, but bounding the number of quantifier alternations by a function of the parameter. We consider the parameterized problem QSAT(LEVEL), where the input is a quantified Boolean formula $\varphi = \exists X_1 \forall X_2 \exists X_3 \dots Q_k X_k \psi$, where each X_i is a sequence of variables. The parameter is k, and the question is whether φ is true. We define the parameterized complexity class PH(LEVEL) to be the class of all parameterized problems that can be fpt-reduced to QSAT(LEVEL).

Alternative Characterizations

Alternatively, we can characterize the class PH(LEVEL) using Alternating Turing machines (ATMs), which generalize regular (non-deterministic) Turing machines (see, e.g., Chandra, Kozen, and Stockmeyer 1981). We will use this characterization below to show membership in PH(LEVEL).

The states of an ATM are partitioned into *existential* and *universal states*. Intuitively, if the ATM $\mathbb{M}$ is in an existential state, it accepts if there is some successor state that accepts, and if $\mathbb{M}$ is in a universal state, it accepts if all successor states accept. We say that $\mathbb{M}$ is ℓ-alternating for a problem Q, for $\ell \geq 0$, if for each input x of Q, for each run of $\mathbb{M}$ on x, and for each computation path in this run, there are at most ℓ transitions from an existential state to a universal state, or vice versa. The class PH(LEVEL) consists of all problems that can be solved in fixed-parameter tractable time by an ATM whose number of alternations is bounded by a function of the parameter.

Proposition* 4. *Let Q be a parameterized problem. Then $Q \in$ PH(LEVEL) if and only if there exist a computable function $f : \mathbb{N} \to \mathbb{N}$ and an ATM $\mathbb{M}$ such that: (1) $\mathbb{M}$ solves Q in fixed-parameter tractable time, and (2) for each slice Q_k of Q, $\mathbb{M}$ is $f(k)$-alternating.*

As a direct consequence of this definition, we get that the class PH(LEVEL) is closed under fpt-reductions. Next, to further illustrate the robustness of the class PH(LEVEL), we characterize this class using first-order logic model checking (which has also been used to characterize the classes of the well-known W-hierarchy and the A-hierarchy, see, e.g. Flum and Grohe 2006). Consider the problem MC[FO], where the input consists of a relational structure $\mathcal{A}$, and a first-order formula $\varphi = \exists X_1 \forall X_2 \exists X_3 \dots Q_k X_k \psi$ in prenex form, where $Q_k = \forall$ if k is even and $Q_k = \exists$ if k is odd. The question is whether $\mathcal{A} \models \varphi$. The problem MC[FO] is PH(LEVEL)-complete when parameterized by k.

Proposition* 5. MC[FO] *parameterized by the number k of quantifier alternations in the first-order formula is* PH(LEVEL)*-complete.*

Relation to other parameterized variants of the PH

In the parameterized complexity literature, more variants of the Polynomial Hierarchy have been studied. We briefly consider how the class PH(LEVEL) relates to these. Firstly, for each $i \geq 1$, the parameterized complexity classes para-Σ_i^{p} and para-Π_i^{p} (which are parameterized variants of the classes Σ_i^{p} and Π_i^{p}) are contained in the class PH(LEVEL). Moreover, PH(LEVEL) is contained in para-PSPACE. These inclusions are all strict, unless the PH collapses.

Another parameterized variant of the PH that has been studied is the A-hierarchy (Flum and Grohe 2006, Chapter 8), containing the parameterized complexity classes A[t] for each $t \geq 1$. Each class A[t] is defined as the class of all problems that can be fpt-reduced to MC[FO], restricted to first-order formulas φ (in prenex form) with a quantifier prefix that starts with an existential quantifier and that contains t quantifier alternations, parameterized by the size of φ. From this definition, it directly follows that A[t] is contained in PH(LEVEL), for each $t \geq 1$. The A-hierarchy also contains the parameterized classes AW[$*$] $\subseteq$ AW[SAT] $\subseteq$ AW[P], each of which contains all classes A[t]. These classes are also contained in PH(LEVEL). Moreover, the inclusion of all these classes in PH(LEVEL) is strict, unless P = NP.

Completeness for PH(LEVEL) and para-PSPACE

In this section, we provide a complete parameterized complexity classification for the problem SYMBOLIC-MC*[$\mathcal{L}$]. We already considered the case for $\mathcal{L}$ = LTL\U,X in Section , which was shown to be para-co-NP-complete. We give (negative) parameterized complexity results for the other cases. An overview of the results can be found in Table 1. Firstly, we show that for the case of LTL, allowing at least one of the temporal operators U or X leads to para-PSPACE-completeness.

Theorem 6. SYMBOLIC-MC*[LTL\U] *is* para-PSPACE-*complete.*

Proof. Membership follows from the PSPACE-membership of SYMBOLIC-MC[LTL]. We show hardness by showing that the problem is already PSPACE-hard for a constant parameter value. We do so by giving a reduction from QSAT. Let $\varphi_0 = \forall x_1.\exists x_2 \dots Q_n x_n.\psi$ be a quantified Boolean formula. We may assume without loss of generality that $(n \bmod 4) = 1$, and thus that $Q_n = \forall$. We construct a Kripke structure $\mathcal{M}$ symbolically represented by (φ_R, α_0), whose reachability diameter is polynomial in the size of φ_0, and an LTL formula φ that does not contain the U operator, in such a way that φ_0 is true if and only if $\mathcal{M} \not\models \neg\varphi$. (So technically, we are reducing to the co-problem of SYMBOLIC-MC*[LTL\U]. Since PSPACE is closed under complement, this suffices to show PSPACE-hardness.)

The idea is to construct a full binary tree (of exponential size), with bidirectional transitions between each parent and

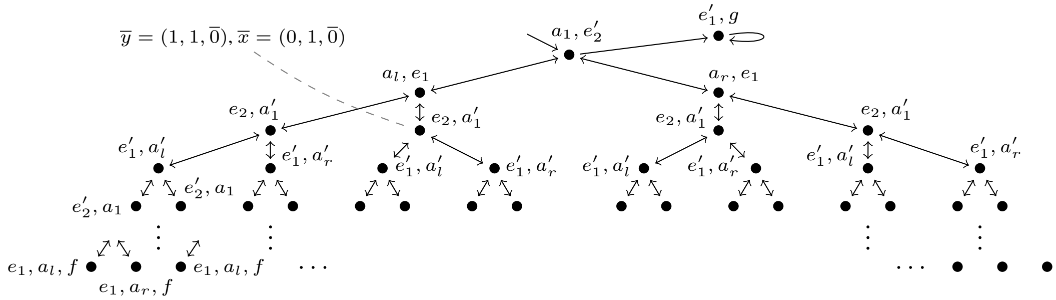

Figure 3: (The reachable part of) the structure $\mathcal{M}$ in the proof of Theorem 6.

child, and to label the nodes of this tree in such a way that a constant-size LTL formula can be used to force paths to be a traversal of this tree corresponding to a QBF model of the formula φ_0. The idea of using LTL formulas to force paths to be traversals of exponential-size binary trees was already mentioned by Kroening et al. (2011). We construct the Kripke structure $\mathcal{M}$ as depicted in Figure 3. For a detailed treatment of how to construct φ_R and α_0 to get this structure $\mathcal{M}$, we refer to the technical report (De Haan and Szeider 2015). It is straightforward to check that the recurrence diameter *rd*($\mathcal{M}$) of $\mathcal{M}$ is bounded by $2n$ as the longest simple path in $\mathcal{M}$ is from some leaf in the tree to another leaf.

More concretely, the intuition behind the construction of $\mathcal{M}$ is as follows. Every transition from the i-th level to the $(i + 1)$-th level (where the root is at the 0-th level) corresponds to assigning a truth value to the variable x_{i+1}. We use variables $\overline{x} = (x_1, \dots, x_n)$ to keep track of the truth assignment in the current position of the tree, and variables $\overline{y} = (y_1, \dots, y_n)$ to keep track of what level in the tree the current position is (at level i, exactly the variables $y_1, \dots, y_i$ are set to true). At the even levels i, we use the variables a_1, a_l, a_r (and a_1', a_l', a_r') to ensure that (in a single path) both possible truth assignments to the (universally quantified) variable x_{i+1} are used. At the odd levels i, we use the variables e_1, e_2 (and e_1', e_2') to ensure that one of both possible truth assignments to the (existentially quantified) variable x_{i+1} is used. We need the copies $a_1', e_1', \dots$ to be able to enforce the intended (downward and upward) traversal of the tree. Then, the variable f is used to signal that a leaf has been reached, and the variable g is used to signal that the path is in the sink state. For a detailed specification of how to construct the LTL formula φ in such a way that it enforces such a traversal of the structure $\mathcal{M}$, we refer to the technical report (De Haan and Szeider 2015).

We can then show that φ_0 is true if and only if $\mathcal{M} \not\models \neg\varphi$. By construction of $\mathcal{M}$ and φ, all paths starting in the initial state of $\mathcal{M}$ that satisfy φ naturally correspond to a QBF model of φ_0, and all QBF models of φ_0 correspond to such a path. Assume that φ_0 is true. Then there exists a QBF model of φ_0. Then there exists a path satisfying φ, and thus $\mathcal{M} \not\models \neg\varphi$. Conversely, assume that $\mathcal{M} \not\models \neg\varphi$. Then there exists a path that satisfies φ. Therefore, there exists a QBF model of φ_0, and thus φ_0 is true. $\square$

Theorem 7. SYMBOLIC-MC*[LTL\X] *is* para-PSPACE-*complete.*

Proof. Membership follows from the PSPACE-membership of SYMBOLIC-MC[LTL]. We show para-PSPACE-hardness by modifying the reduction in the proof of Theorem 6. The idea is to simulate the X operator using the U operator. Given an instance of QSAT, we construct the Kripke structure $\mathcal{M}$ and the LTL formula φ as in the proof of Theorem 6. Then, we modify $\mathcal{M}$ and φ as follows. Firstly, we add a fresh variable x_0 to the set of propositions P, we ensure that x_0 is false in the initial state α_0, and we modify φ_R so that in each transition, the variable x_0 swaps truth values. Then, it is straightforward to see that any LTL formula of the form $\mathrm{X}\varphi'$ is equivalent to the LTL formula $(x_0 \rightarrow x_0\mathrm{U}(\neg x_0 \wedge \varphi')) \wedge (\neg x_0 \rightarrow \neg x_0\mathrm{U}(x_0 \wedge \varphi'))$, on structures where x_0 shows this alternating behavior. Using this equivalence, we can recursively replace all occurrences of the X operator in the LTL formula φ. This leads to an exponential blow-up in the size of φ, but since φ is of constant size, this blow-up is permissible. $\square$

Next, we show that for the case of CTL, the problem is complete for PH(LEVEL), even when both temporal operators U and X are disallowed. In order to establish this result, we need the following technical lemma.

Lemma 8. *Given a symbolically represented Kripke structure $\mathcal{M}$ given by (φ_R, α_0) over the set P of propositional variables, and a propositional formula ψ over P, we can construct in polynomial time a Kripke structure $\mathcal{M}'$ given by (φ_R', α_0') over the set $P \cup \{z\}$ of variables (where $z \notin P$) such that:*

- *there exists an isomorphism ρ between the states in the reachable part of $\mathcal{M}$ and the states in the reachable part of $\mathcal{M}'$ that respects the initial states and the transition relations,*
- *each state s in the reachable part of $\mathcal{M}$ agrees with $\rho(s)$ on the variables in P, and*
- *for each state s in the reachable part of $\mathcal{M}$ it holds that $\rho(s) \models z$ if and only if $s \models \psi$.*

Proof. Intuitively, the required Kripke structure $\mathcal{M}'$ can be constructed by adding the variable z to the set P of propositions, and modifying the formula φ_R specifying the transition relation and the initial state α_0 appropriately. In the new initial state α_0', the variable z gets the truth value 1 if and only if $\alpha_0 \models \psi$. Moreover, the transition relation specified by φ_R' ensures that in any reached state α, the variable z gets the truth value 1 if and only if $\alpha \models \psi$.

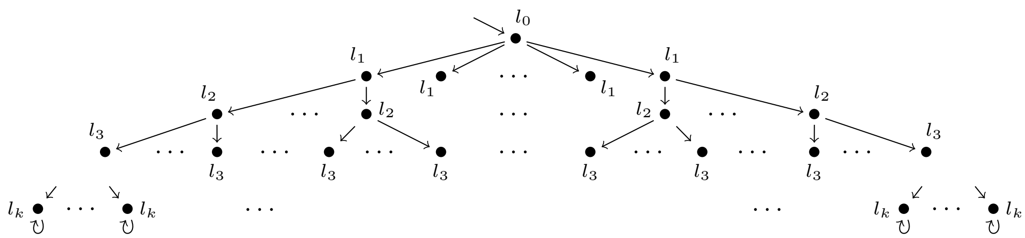

Figure 4: (The reachable part of) the structure $\mathcal{M}$ in the proof of Theorem 9.

Concretely, we define $\mathcal{M}' = (\varphi'_R, \alpha'_0)$ over the set $P \cup \{z\}$ as follows. We let $\alpha'_0(p) = \alpha_0(p)$ for all $p \in P$, and we let $\alpha'_0(z) = 1$ if and only if $\alpha_0 \models \psi$. Then, we define φ'_R by letting:

$$\varphi'_R(\overline{x}, z, \overline{x}', z') = \varphi_R(\overline{x}, \overline{x}') \wedge (z' \leftrightarrow \psi(\overline{x})).$$

The isomorphism ρ can then be constructed as follows. For each state α in $\mathcal{M}$, $\rho(\alpha) = \alpha \cup \{z \mapsto 1\}$ if $\alpha \models \psi$, and $\rho(\alpha) = \alpha \cup \{z \mapsto 0\}$ if $\alpha \not\models \psi$. □

Theorem 9. SYMBOLIC-MC*[CTL] *is* PH(LEVEL)*-complete. Moreover, hardness already holds for* SYMBOLIC-MC*[CTL$\backslash$U,X].

Proof. In order to show hardness, we give an fpt-reduction from QSAT(LEVEL). Let $\varphi = \exists X_1 \forall X_2 \dots Q_k X_k \psi$ be an instance of QSAT(LEVEL). We construct a Kripke structure $\mathcal{M}$ over a set P of propositional variables represented symbolically by (φ_R, α_0), with polynomial recurrence diameter, and a CTL formula Φ such that φ is true if and only if $\mathcal{M} \models \Phi$.

The idea is to let $\mathcal{M}$ consist of a (directed) tree of exponential size, as depicted in Figure 4. The tree consists of k levels (where the root is at the 0-th level). All nodes on the i-th level are labelled with proposition l_i. Moreover, each node is associated with a truth assignment over the variables in $X = \bigcup_{1 \leq i \leq k} X_i$. For each node n at the i-th level (for $0 \leq i < k$) with corresponding truth assignment α_n, and for each truth assignment α to the variables in X_{i+1}, there is a child node of n (at the $(i+1)$-th level) whose corresponding assignment agrees with α on the variables in X_{i+1}. Also, the truth assignment corresponding to each child of n agrees with α_n on the variables in $X_1, \dots, X_i$. Moreover, by Lemma 8, we may assume without loss of generality that there is a propositional variable z_ψ in P that in each state is set to 1 if and only if this state sets the propositional formula ψ (over X) to true.

We show how to construct the Kripke structure $\mathcal{M} = (\varphi_R, \alpha_0)$. Remember that $P = X_1 \cup \dots \cup X_k \cup \{l_0, l_1, \dots, l_k\}$ (for the sake of simplicity, we leave treatment of the propositional variable z_ψ to the technique discussed in the proof of Lemma 8). We let α_0 be the truth assignment that sets only the propositional variable l_0 to true, and all other propositional variables to false. Then, we define $\varphi_R(\overline{p}, \overline{p}')$ as the conjunction of several subformulas. The first conjuncts ensure that in each level of the tree, the propositional variables l_i get the right truth value:

$$\bigwedge_{0 \leq i < k} l_i \rightarrow l'_{i+1}, \quad (l_k \rightarrow l'_k) \quad \text{and} \quad \bigwedge_{0 \leq i < i' \leq k} \neg(l'_i \wedge l'_{i'}).$$

The following conjunct ensures that the partial truth assignment of a node at the i-th level of the tree agrees with its parent on all variables in $X_1, \dots, X_{i-1}$.

$$\bigwedge_{1 \leq i \leq n} (l'_i \rightarrow \bigwedge_{x \in X_1 \cup \dots \cup X_{i-1}} (x \leftrightarrow x')).$$

This concludes our construction of the Kripke structure $\mathcal{M}$ as depicted in Figure 4. Clearly, the longest simple path in $\mathcal{M}$ is a root-to-leaf path, which has length k.

Then, using this structure $\mathcal{M}$, we can express the quantified Boolean formula φ in CTL as follows. We define $\Phi = \exists\mathrm{F}(l_1 \wedge \forall\mathrm{F}(l_2 \wedge \exists\mathrm{F}(l_3 \wedge \cdots Q_k\mathrm{F}(l_k \wedge z_\psi) \cdots)))$. By construction of Φ, we get that those subtrees of $\mathcal{M}$ that naturally correspond to witnesses for the truth of this CTL formula Φ exactly correspond to the QBF models for φ. From this, we directly get that φ is true if and only if $\mathcal{M} \models \Phi$.

In order to prove membership in PH(LEVEL), we show that SYMBOLIC-MC*[CTL] can be decided in fpt-time by an ATM $\mathbb{M}$ that is $f(k)$-alternating. The algorithm implemented by $\mathbb{M}$ takes a different approach than the well-known dynamic programming algorithm for CTL model checking for explicitly encoded Kripke structures (see, e.g., Baier and Katoen 2008, Section 6.4.1). Since symbolically represented Kripke structures can be of size exponential in the input, this bottom-up algorithm would require exponential time. Instead, we employ a top-down approach, using (existential and universal) non-determinism to quantify over the possibly exponential number of states.

We consider the function `CTL-MC`, given in pseudo-code in Algorithm 1, which takes as input the Kripke structure $\mathcal{M}$ in form of its representation (φ_R, α_0), a state α in $\mathcal{M}$, a CTL formula Φ and the recurrence diameter *rd*$(\mathcal{M})$ of $\mathcal{M}$ (in unary), and outputs 1 if and only if α makes Φ true. The algorithm only needs to check for paths of length at most $m = \mathit{rd}(\mathcal{M})$ in the case of the U operator, because any path longer than m must cycle. Note that in this algorithm, we omit the case for the operator F, as any CTL formula $\exists\mathrm{F}\Phi$ is equivalent to $\exists\top\mathrm{U}\Phi$. It is readily verified that this algorithm correctly computes whether $\mathcal{M}, \alpha \models \Phi$. Therefore, $\mathcal{M} \models \Phi$ if and only if `CTL-MC(`$\mathcal{M}$, α_0, Φ, m`)` returns 1, where m is the unary encoding of *rd*$(\mathcal{M})$.

It remains to verify that the algorithm `CTL-MC` can be implemented in fpt-time by an $f(k)$-alternating ATM $\mathbb{M}$. We can construct $\mathbb{M}$ in such a way that the existential guesses are done using the existential non-deterministic states of $\mathbb{M}$, and the universal guesses by the universal non-deterministic states. Note that the recursive call in the case for $\neg\Phi_1$ is

Algorithm 1: Recursive CTL model checking using bounded alternation.

```
function CTL-MC(𝓜, α, Φ, m):
  switch Φ do
    case p ∈ P: return α(p) ;
    case ¬Φ₁: return not CTL-MC(𝓜, α, Φ₁, m) ;
    case Φ₁ ∧ Φ₂: return CTL-MC(𝓜, α, Φ₁, m)
    and CTL-MC(𝓜, α, Φ₂, m) ;
    case ∃XΦ₁:
      (existentially) pick a state α' in 𝓜 ;
      /* guess successor state */
      if φ_R(α, α') is false then return 0 ;
      /* check transition */
      return CTL-MC(𝓜, α', Φ₁, m)
    case ∃Φ₁UΦ₂:
      (existentially) pick some m' ≤ m ;
      /* guess length of path */
      (existentially) pick states α₁, ..., α_{m'} in 𝓜 ;
      /* guess path */
      (universally) pick some 1 ≤ j < m' ;
      /* check all states in path */
      if φ_R(α_j, α_{j+1}) is false then return 0 ;
      /* check transition */
      if CTL-MC(𝓜, α_j, Φ₁, m) = 0
        then return 0 ;
      return CTL-MC(𝓜, α_{m'}, Φ₂, m)
```

preceded by a negation, so the existential and universal nondeterminism swaps within this recursive call. The recursion depth of the algorithm is bounded by $|\Phi| = k$, since each recursive call strictly decreases the size of the CTL formula used. Moreover, in each call of the function CTL-MC, at most two recursive calls are made (not counting recursive calls at deeper levels of recursion). Therefore, the running time of $\mathbb{M}$ is bounded by $2^k\text{poly}(n)$, where n is the input size. Also, since in each call of the function at most two alternations between existential and universal non-determinism are used (again, not counting at deeper levels of recursion), we know that $\mathbb{M}$ is 2^k-alternating. (This bound on the number of alternations needed can be improved with a more careful analysis and some optimizations to the algorithm.) ☐

Finally, we complete the parameterized complexity classification of the problem SYMBOLIC-MC* by showing membership in PH(LEVEL) for the case of CTL$^\star\backslash$U,X.

Theorem 10. SYMBOLIC-MC*[CTL$^\star\backslash$U,X] *is* PH(LEVEL)-*complete.*

Proof. Hardness for PH(LEVEL) follows from Theorem 9. We show membership in PH(LEVEL), by describing an algorithm A to solve the problem that can be implemented by an ATM $\mathbb{M}$ that runs in fpt-time and that is $f(k)$-alternating. The algorithm works similarly to Algorithm 1, described in the proof of Theorem 9, and recursively decides the truth of a CTL* formula in a state. The difference with Algorithm 1 is that it does not look only at the outermost temporal operators of the CTL* formula in a recursive step, but considers possibly larger subformulas in each recursive step. Let $\exists\varphi$ be a CTL* formula, and let s be a state in $\mathcal{M}$. The algorithm A then considers all maximal subformulas $\psi_1, \dots, \psi_\ell$ of φ that are CTL* state formulas as atomic propositions $p_1, \dots, p_\ell$, turning the formula φ into an LTL formula. Since φ does not contain the operators U and X, we know that in order to check the existence of an infinite path satisfying φ, it suffices to look for lasso-shaped paths of bounded length (linear in *rd*($\mathcal{M}$) and exponential in the size of φ), i.e., a finite path followed by a finite cycle (Kroening et al. 2011). The algorithm A then uses (existential) nondeterminism to guess such a lasso-shaped path π, and to guess for each state which of the propositions $p_1, \dots, p_\ell$ are true, and verifies that π witnesses truth of $\exists\varphi$. Then, in order to ensure that it correctly determines whether $\exists\varphi$ is true, the algorithm needs to verify that it guessed the right truth values for $p_1, \dots, p_\ell$ in π. It does so by recursively determining, for each state s' in the lasso-shaped path π, and each p_i, whether ψ_i is true in s' if and only if it guessed p_i to be true in s'. (In order to ensure that in each level of recursion there are only a constant number of recursive calls, like Algorithm 1, the algorithm A uses universal nondeterminism iterate over each p_i and each s'.) The algorithm then reports that $\exists\varphi$ is true in s if and only if (1) the guesses for π and the truth values of $p_1, \dots, p_\ell$ together form a correct witness for truth of $\exists\varphi$, and (2) for each p_i and each s' it holds that p_i was guessed to be true in s' if and only if ψ_i is in fact true in s'. The recursive cases for CTL* formulas where the outermost operator is not temporal are analogous to Algorithm 1. Like Algorithm 1, the algorithm runs in fpt-time and is 2^k-alternating. ☐

Conclusion

An essential technique for solving the fundamental problem of temporal logic model checking is the SAT-based approach of bounded model checking. Even though its good performance in settings with large Kripke structures and small temporal logic specifications provides a good handle for a parameterized complexity analysis, the theoretical possibilities of the bounded model checking method have not been structurally investigated. We contributed to closing this gap by providing a complete parameterized complexity classification of the model checking problem for fragments of the temporal logics LTL, CTL and CTL*, where the Kripke structures are represented symbolically, and have a restricted recurrence diameter. In particular, we showed that the known case of LTL$\backslash$U,X is the only case that allows an fpt-reduction to SAT, by showing completeness for the classes PH(LEVEL) and para-PSPACE for all other cases. We hope that providing a clearer theoretical picture of the settings where the powerful technique of bounded model checking can be applied helps guide engineering efforts for developing algorithms for the problem of temporal logic model checking.

Future research includes extending the parameterized complexity investigation to further restrictions on the fragments of the temporal logics that we consider, with the aim of identifying more settings where bounded model checking can be applied efficiently. This can be done, for instance, by taking into account more input parameters. Additionally, the complexity investigation can be extended to other temporal logics that are used in knowledge representation and reasoning, such as the alternating-time temporal logics ATL and ATL* (Fisher 2008; Alur, Henzinger, and Kupferman 2002). For this setting, it would be important to identify further restrictions, because ATL and ATL* generalize CTL and CTL*, respectively. Moreover, it would be valuable to investigate to what extent the complexity results hold in settings where structures are described symbolically using other formalisms (see, e.g., Van der Hoek, Lomuscio, and Wooldridge 2006).

Acknowledgments

This work is supported by the Austrian Science Fund (FWF), project P26200 (Parameterized Compilation).

References

Alur, R.; Henzinger, T. A.; and Kupferman, O. 2002. Alternating-time temporal logic. *J. of the ACM* 49(5):672–713.

Arora, S., and Barak, B. 2009. *Computational Complexity – A Modern Approach*. Cambridge University Press.

Baier, C., and Katoen, J.-P. 2008. *Principles of Model Checking*. MIT Press.

Biere, A.; Cimatti, A.; Clarke, E. M.; and Zhu, Y. 1999. Symbolic model checking without BDDs. In Cleaveland, R., ed., *Proceedings of TACAS 1999*, volume 1579 of *Lecture Notes in Computer Science*, 193–207. Springer Verlag.

Biere, A.; Cimatti, A.; Clarke, E. M.; Strichman, O.; and Zhu, Y. 2003. Bounded model checking. *Advances in computers* 58:117–148.

Biere, A. 2009. Bounded model checking. In Biere, A.; Heule, M.; van Maaren, H.; and Walsh, T., eds., *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press. 457–481.

Bundala, D.; Ouaknine, J.; and Worrell, J. 2012. On the magnitude of completeness thresholds in bounded model checking. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science (LICS)*, 155–164. IEEE Computer Society.

Chandra, A. K.; Kozen, D. C.; and Stockmeyer, L. J. 1981. Alternation. *J. of the ACM* 28(1):114–133.

Clarke, E.; Grumberg, O.; Jha, S.; Lu, Y.; and Veith, H. 2001. Progress on the state explosion problem in model checking. In *Informatics*, volume 2000 of *Lecture Notes in Computer Science*, 176–194. Springer Verlag.

Clarke, E. M.; Kroening, D.; Ouaknine, J.; and Strichman, O. 2004. Completeness and complexity of bounded model checking. In *Proceedings of VMCAI 2004*, volume 2937 of *Lecture Notes in Computer Science*, 85–96. Springer Verlag.

Clarke, E. M.; Emerson, E. A.; and Sifakis, J. 2009. Model checking: algorithmic verification and debugging. *Communications of the ACM* 52(11):74–84.

Demri, S.; Laroussinie, F.; and Schnoebelen, P. 2006. A parametric analysis of the state-explosion problem in model checking. *J. of Computer and System Sciences* 72(4):547–575.

Downey, R. G., and Fellows, M. R. 2013. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer Verlag.

Emerson, E. A., and Lei, C.-L. 1987. Modalities for model checking: branching time logic strikes back. *Science of computer programming* 8:275–306.

Fichte, J. K., and Szeider, S. 2013. Backdoors to normality for disjunctive logic programs. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI)*, 320–327. AAAI Press.

Fisher, M.; Gabbay, D.; and Vila, L. 2005. *Handbook of Temporal Reasoning in Artificial Intelligence*. Elsevier Science Publishers, North-Holland.

Fisher, M. 2008. Temporal representation and reasoning. In *Handbook of Knowledge Representation*, volume 3 of *Foundations of Artificial Intelligence*. Elsevier. 513–550.

Flum, J., and Grohe, M. 2003. Describing parameterized complexity classes. *Information and Computation* 187(2):291–319.

Flum, J., and Grohe, M. 2006. *Parameterized Complexity Theory*, volume XIV of *Texts in Theoretical Computer Science. An EATCS Series*. Berlin: Springer Verlag.

Göller, S. 2013. The fixed-parameter tractability of model checking concurrent systems. In *Proceedings of CSL 2013*, volume 23 of *LIPIcs*, 332–347.

de Haan, R., and Szeider, S. 2014a. Compendium of parameterized problems at higher levels of the Polynomial Hierarchy. Technical Report TR14–143, *Electronic Colloquium on Computational Complexity* (ECCC).

de Haan, R., and Szeider, S. 2014b. The parameterized complexity of reasoning problems beyond NP. In Baral, C.; De Giacomo, G.; and Eiter, T., eds., *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR)*. AAAI Press.

de Haan, R., and Szeider, S. 2015. Parameterized complexity results for symbolic model checking of temporal logics. Technical Report AC-TR-15-002, Algorithms and Complexity Group, TU Wien.

van der Hoek, W.; Lomuscio, A.; and Wooldridge, M. 2006. On the complexity of practical ATL model checking. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 201–208. ACM.

Kroening, D.; Ouaknine, J.; Strichman, O.; Wahl, T.; and Worrell, J. 2011. Linear completeness thresholds for bounded model checking. In *Proceedings of CAV 2011*, volume 6806 of *LNCS*, 557–572. Springer Verlag.

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

Lück, M.; Meier, A.; and Schindler, I. 2015. Parameterized complexity of CTL. In *Proceedings of LATA 2015*, volume 8977 of *Lecture Notes in Computer Science*. Springer Verlag. 549–560.

Niedermeier, R. 2006. *Invitation to Fixed-Parameter Algorithms*. Oxford Lecture Series in Mathematics and its Applications. Oxford: Oxford University Press.

Pfandler, A.; Rümmele, S.; and Szeider, S. 2013. Backdoors to abduction. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*.

Samulowitz, H.; Davies, J.; and Bacchus, F. 2006. Preprocessing QBF. In *Proceedings of the 12th International Conference on Principles and Practice of Constraint Programming (CP)*, volume 4204 of *Lecture Notes in Computer Science*, 514–29. Springer Verlag.

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

Vardi, M. Y., and Wolper, P. 1986. Automata-theoretic techniques for modal logics of programs. *J. of Computer and System Sciences* 32(2):183–221.