The Complexity of Propositional Linear Temporal Logics in Simple Cases [1]

Stéphane Demri and Philippe Schnoebelen

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

It is well-known that model checking and satisfiability for PLTL are **PSPACE**-complete. By contrast, very little is known about whether there exist some interesting fragments of PLTL with a lower worst-case complexity. Such results would help understand why PLTL model checkers are successfully used in practice.

In this paper we investigate this issue and consider model checking and satisfiability for all fragments of PLTL obtainable by restricting (1) the temporal connectives allowed, (2) the number of atomic propositions, and (3) the temporal height.

Key Words: logic in computer science, computational complexity, verification, temporal logic, model checking

1. INTRODUCTION

Background. PLTL is the standard linear-time propositional temporal logic used in the specification and automated verification of reactive systems [MP92, Eme90]. It is well-known that model checking and satisfiability for PLTL are **PSPACE**-complete [SC85, HR83, Wol83]. This did not deter some research groups from implementing PLTL model checkers or provers, and using them successfully in practice [BBC$^+$96, CGH97, Hol97]. The fundamental question this raises is "*what makes PLTL feasible in practice ?*".

To this question, the common answer starts with the observation that the **PSPACE** complexity only applies to the formula part of the problem [LP85], and it is only a *worst-case complexity*. Then, it is often argued that the PLTL formulae used in actual practical situations are not very complex, have a low temporal height

[1] This article is a completed version of [DS98].

(number of nested temporal connectives) and are mainly boolean combinations of simple eventuality, safety, responsiveness, fairness, … properties.

Certainly the question calls for a systematic theoretical study, aiming at turning the above answers into formal theorems and helping understand the issue at hand. If we consider for example SAT, the famous Boolean Satisfiability problem, there are current in-depth investigations of tractable subproblems (e.g., [Dal96, FV98]). Regarding PLTL, we know of no systematic study of this kind in the literature. This is all the more surprising when considering the wide use of PLTL model checkers for reactive systems.

Our objectives. In this article, we develop a systematic study, looking for natural subclasses of PLTL formulae for which complexity decreases. The potential results are (1) a better understanding of what makes the problem **PSPACE**-hard, (2) the formal identification of classes of temporal formulae with lower complexity, called *simple cases*, (3) the discovery of more efficient algorithms for such simple cases. Furthermore, since PLTL is the most basic temporal logic, simple cases for PLTL often have corollaries for other logics.

As a starting point, we revisit the complexity questions from [SC85] when there is a bound on the number of propositions and/or on the temporal height of formulae. More precisely, let us write $\mathsf{H}_1, \mathsf{H}_2, \ldots$ for an arbitrary set of linear-time combinators among $\{\mathsf{U}, \mathsf{F}, \mathsf{X}, \ldots\}$ and let $\mathtt{L}^k_n(\mathsf{H}_1, \ldots)$ denote the fragment of PLTL restricted to formulae (1) only using combinators $\mathsf{H}_1, \ldots$, (2) of temporal height at most k, and (3) with at most n distinct atomic propositions. In this article we measure the complexity of model checking and satisfiability for all these fragments.

The choice of this starting point is very natural, and it is relevant for our original motivations:

- For the propositional calculus and for several modal logics (K45, KD45, S5, von Wright's *logic of elsewhere*, …), satisfiability becomes linear-time when at most n propositions can be used (see [Hal95, Dem96]). By contrast, satisfiability for K remains **PSPACE**-complete even when only one proposition is allowed. What about PLTL ?

- In practical applications, the temporal height often turns out to be at most 2 (or 3 when fairness is involved) even when the specification is quite large and combines a large number of temporal constraints. This bounded height is often invoked as a reason why PLTL model checking is feasible in practice. Can this be made formal ?

Our contribution.

1. Our first contribution is an evaluation of the computational complexity of model checking and satisfiability for all $\mathtt{L}^k_n(\mathsf{H}_1, \ldots)$ fragments. A table in section 8 summarizes this.

2. We also identify new *simple cases* for which the complexity is lowered (only **NP**-complete). For these we give (non-deterministic) algorithms. We think it is worth investigating whether the ideas underlying these algorithms could help develop deterministic algorithms that perform measurably better (on the relevant *simple case*) than the usual methods. These results also have implications beyond PLTL: e.g. **NP**-completeness of PLTL without temporal nesting (Prop. 7.4) leads to a Δ_2^P model checking algorithm for CTL$^+$ and FCTL [LMS01].

3. A third contribution is the proof techniques we develop: we show how a few logspace reductions allow to compute almost all the complexity measures we needed (only a few remaining ones are solved with ad-hoc methods). These reductions lead to a few rules of thumb (summarized in section 8) that can be used as guidelines. Additionally, some of our reductions transform well-known problems (SAT or QBF) into model checking problems for formulae with a simple structure (e.g., low temporal height) and can be used in other contexts. The second author used them for very restricted fragments of CTL+Past [LS00].

We believe that these constructions are interesting in their own right and think that the scarcity of available proofs and exercices suitable for a classroom framework is unfortunate when PLTL model checking is now widely taught in computer science curriculums.

Related work. It is common to find papers considering *extensions* of earlier temporal logics. The search for *fragments* with lower complexity is less common (especially works considering model checking). [EES90] investigates (very restricted) fragments of CTL (a branching-time logic) where satisfiability is polynomial-time. [KV98] studies particular PLTL formulae for which there is a linear-sized equivalent CTL formula: one of the aims is to understand when and why PLTL model checking often behaves computationally well in practice. [BK98] tries to understand why Mona performs well in practice and isolates a fragment of WS1S where the usual non-elementary blowup does not occur. [Hal95] investigates, in a systematic way, the complexity of satisfiability (*not model checking*) for various multimodal logics when the modal height or the number of atomic propositions is restricted: in fact PLTL is quite different from the more standard multimodal logics and we found it behaves differently when syntactic restrictions are enforced. In [Hem00], the complexity of fragments of modal logics is also studied by restricting the set of logical (boolean and temporal) operators. These fragments are mainly relevant for description logics (see, e.g., [DLNN97]).

As far as PLTL is concerned, some complexity results for some particular restricted fragments of PLTL can be found in [EL87, CL93, Spa93, DFR00] but these are not systematic studies sharing our objectives. [Har85] has a simple proof, based on a general reduction from tiling problems into modal logics, that *satisfiability* for $\mathsf{L}(\mathsf{F}, \mathsf{X})$ is **PSPACE**-hard. In fact, the same proof (or the proofs from [Spa93, DFR00]) shows that **PSPACE**-hardness is already obtained with temporal height 2.

Finally, there is a special situation with $\mathsf{L}(\mathsf{F})$ and $\mathsf{L}(\mathsf{X})$. These two very limited fragments of PLTL actually coincide (semantically) with, respectively, the modal logics S4.3Dum (also called S4.3.1 or D) [Bul65, Seg71, Gor94] and KDAlt_1 [Seg71]. **NP**-completeness of S4.3Dum satisfiability has been first proved in [ON80] and generalized in [Spa93] to any modal logic extending the modal logic S4.3. The complexity of $\mathsf{L}(\mathsf{X})$ satisfiability is also studied in [SR99].

Plan of the article. Section 2 recalls various definitions we need throughout the article. Sections 3 and 4 study the complexity of PLTL fragments when the number of atomic propositions is bounded. Logspace transformations from QBF into model checking can be found in Section 5 and Section 6. Section 7 studies the complexity of PLTL fragments when the temporal height is bounded. Section 8 contains concluding remarks and provides a table summarizing the complete picture we have established about complexity for PLTL fragments.

2. BASIC DEFINITIONS AND RESULTS

Computational complexity. We assume that the reader understands what is meant by complexity classes such as **L** (deterministic logspace), **NL** (non-deterministic logspace), **P** (polynomial-time), **NP** and **PSPACE**, see e.g. [Pap94]. Given two decision problems $\mathcal{P}_1$ and $\mathcal{P}_2$, we write $\mathcal{P}_1 \leq_L \mathcal{P}_2$ when there exists a logspace transformation (many-one reduction) from $\mathcal{P}_1$ into $\mathcal{P}_2$. In the rest of the article, all the reductions are logspace, and by "$\mathcal{C}$-hardness" we mean "logspace hardness in the complexity class $\mathcal{C}$".

Temporal logic. We follow notations and definitions from [Eme90]: PLTL is a propositional linear-time temporal logic based on a countably infinite set $Prop = \{A_1, A_2, \ldots, P_1, P_2, \ldots\}$ of propositional variables, the classical boolean connectives $\top$, $\neg$, $\wedge$, and the temporal operators X (next), U (until), F (sometimes). The set $\{\varphi, \ldots\}$ of formulae is defined in the standard way, using the connectives $\perp$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ and G (always) as abbreviations with their standard meaning. We let $|\varphi|$ denote the *length* (or *size*) of the string φ, assuming a reasonably succinct encoding.

Following the usual notations (see, e.g., [SC85, Eme90]), we let $\mathtt{L}(\mathsf{H}_1, \mathsf{H}_2, \ldots)$ denote the fragment of PLTL for which only the temporal operators $\mathsf{H}_1, \mathsf{H}_2, \ldots$ are allowed [3]. For instance $\mathtt{L}(\mathsf{U})$ is "PLTL without X", as used in [Lam83].

$Prop(\varphi)$ denotes the set of propositional variables occurring in φ. The *temporal height* of φ, written $th(\varphi)$, is the maximum number of nested temporal operators in φ. We write $\mathtt{L}^k_n(\mathsf{H}_1, \ldots)$ to denote the fragment of $\mathtt{L}(\mathsf{H}_1, \ldots)$ where at most $n \geq 1$ propositions are used, and at most temporal height $k \geq 0$ is allowed. We write nothing for n and/or k (or we use ω) when no bound is imposed: $\mathtt{L}(\mathsf{H}_1, \ldots) = \mathtt{L}^\omega_\omega(\mathsf{H}_1, \ldots)$.

For example, for φ given as $(A \Rightarrow \mathsf{XX}B)\mathsf{U}(\neg\mathsf{X}A)$, we have $Prop(\varphi) = \{A, B\}$ and $th(\varphi) = 3$ so that $\varphi \in \mathtt{L}^3_2(\mathsf{U}, \mathsf{X})$.

Flat Until. We say a PLTL formula φ, of the form $\psi\mathsf{U}\psi'$, uses *flat Until* when the left-hand side, ψ, does not contain any temporal combinator (i.e., ψ is a boolean combination of propositional variables) and we write $\psi\mathsf{U}^-\psi'$ when we want to stress that this occurrence of U is flat. E.g., we sometimes write $(A\mathsf{U}^-B)\mathsf{U}C$ for $(A\mathsf{U}B)\mathsf{U}C$.

To the best of our knowledge, Dams was the first to explicitely isolate and name this restricted use of Until [4] and prove that U^- is less expressive than U [Dam99]. He argued that flat Until is often sufficiently expressive in practice, and hoped model checking and satisfiability would be simpler for U^- than for U. In the following, we treat U^- as if it were one more PLTL combinator, more expressive than F but less than U.

Semantics. A *linear-time structure* (also called a *model*) is a pair (S, ε) of an ω-sequence $S = s_0, s_1, \ldots$ of *states*, with a mapping $\varepsilon : \{s_0, s_1, \ldots\} \to 2^{Prop}$ labeling each state s_i with the set of propositions that hold in s_i. We often only write S for a structure, and use the fact that a structure S can be viewed as an infinite string of subsets of $Prop$. Let S be a structure, $i \in \mathbb{N}$ a position, and φ a PLTL formula. The satisfiability relation $\models$ is inductively defined as follows (we omit the usual conditions for the propositional connectives):

- $S, i \models A \overset{\text{def}}{\Leftrightarrow} A \in \varepsilon(s_i)$ (when $A \in Prop$);
- $S, i \models \mathsf{X}\varphi \overset{\text{def}}{\Leftrightarrow} S, i+1 \models \varphi$;
- $S, i \models \mathsf{F}\varphi \overset{\text{def}}{\Leftrightarrow}$ for some $j \geq i$, $S, j \models \varphi$;

[3] Negations are allowed. For instance, $\mathtt{L}(\mathsf{F})$ and $\mathtt{L}(\mathsf{G})$ denote the same fragment.

[4] But flat fragments of temporal logics have been used in many places, e.g. [MC85, DG99, CC00].

• $S, i \models \varphi \mathsf{U} \psi \overset{\text{def}}{\Leftrightarrow}$ there is a $j \geq i$ such that $S, j \models \psi$ and for all $i \leq j' < j$, $S, j' \models \varphi$.

We write $S \models \varphi$ when $S, 0 \models \varphi$.

Satisfiability. We say that a formula φ is *satisfiable* iff $S \models \varphi$ for some S. The *satisfiability problem* for a fragment $\mathtt{L}(\ldots)$, written $SAT(\mathtt{L}(\ldots))$, is the set of all satisfiable formulae in $\mathtt{L}(\ldots)$.

Model checking. A *Kripke structure* $T = (N, R, \varepsilon)$ is a triple such that N is a non-empty set of *states*, $R \subseteq N \times N$ is a total [5] *next-state relation*, and $\varepsilon : N \rightarrow 2^{Prop}$ labels each state s with the (finite) set of propositions that hold in s. A *path* in T is an ω-sequence $S = s_0, s_1, \ldots$ of states of N such that $s_i R s_{i+1}$ for all $i \in \mathbb{N}$. (A path in T is a linear-time structure and a linear-time structure is a possibly infinite Kripke structure where R is a total function.) We follow [Eme90, SC85] and write $T, s \models \varphi$ when *there exists* in T a path S starting from s such that $S \models \varphi$ [6]. The *model checking problem* for a fragment $\mathtt{L}(\ldots)$, written $MC(\mathtt{L}(\ldots))$, is the set of all $\langle T, s, \varphi \rangle$ such that $T, s \models \varphi$ where T is finite and φ is in $\mathtt{L}(\ldots)$. For the definition of $|T|$, the size of T, we use a reasonably succinct encoding of $T = (N, R, \varepsilon)$. In practice, it is convenient to pretend $|T| = card(R) + card(N)$.

Complexity of PLTL. As far as computational complexity is concerned we make a substantial use of the already known upper bounds:

THEOREM 2.1. **[ON80, HR83, SC85]**
$SAT(\mathtt{L}(\mathsf{F}))$ and $MC(\mathtt{L}(\mathsf{F}))$ are **NP***-complete.*
$SAT(\mathtt{L}(\mathsf{F}, \mathsf{X}))$, $MC(\mathtt{L}(\mathsf{F}, \mathsf{X}))$, $SAT(\mathtt{L}(\mathsf{U}))$ and $MC(\mathtt{L}(\mathsf{U}))$ are **PSPACE***-complete.*

As a consequence, most of our proofs establish lower bounds.

Stuttering equivalence. Two models are *equivalent modulo stuttering*, written $S \approx S'$, if they display the same sequence of subsets of $Prop$ when repeated (consecutive) elements are seen as one element only (see [Lam83, BCG88] for a

[5] Only considering Kripke structures with *total* relations is a common technical simplification. Usually it has no impact on the complexity of temporal logic problems. However the "total R" assumption implies that any two states satisfy the same temporal formulae in $\mathtt{L}^{\omega}_{0}(\mathsf{U}, \mathsf{X})$, a fragment for which satisfiability is trivial. In "non-total R" frameworks there is a branching-time formula that behaves as a propositional variable. This can impact complexity: satisfiability for the fragment of K with no propositions is **PSPACE**-complete in a "non-total R" framework [Hem00], and is in **L** in a "total R" framework.

[6] This existential formulation is well suited to complexity studies because it makes model checking closer to satisfiability. It is the dual of the definition used in verification ("all paths from s satisfy φ"), so that all complexity results for model checking can be easily translated, modulo duality, between the two formulations.

formal definition). Lamport argued that one should not distinguish between stutter-equivalent models and he advocated prohibiting X in high-level specifications since

THEOREM 2.2. **[Lam83]** *$S \approx S'$ iff S and S' satisfy the same $\mathtt{L}(\mathsf{U})$ formulae.*

3. BOUNDING THE NUMBER OF ATOMIC PROPOSITIONS

In this section we evaluate the complexity of satisfiability and model checking when the number of propositions is bounded, i.e. for fragments $\mathtt{L}_n(\ldots)$.

When the number of propositions is bounded, satisfiability can be reduced to model checking:

PROPOSITION 3.1. *Let $\mathsf{H}_1, \ldots$ be a non-empty set of PLTL temporal combinators. Then for any $n \in \mathbb{N}$, $SAT(\mathtt{L}_n(\mathsf{H}_1, \ldots)) \leq_L MC(\mathtt{L}_n(\mathsf{H}_1, \ldots))$.*

Proof. Take $\varphi \in \mathtt{L}_n(\mathsf{H}_1, \ldots)$ such that $Prop(\varphi) \subseteq \{A_1, \ldots, A_n\}$. Let $T = (N, R, \varepsilon)$ be the Kripke structure where $N \stackrel{\text{def}}{=} 2^{\{A_1,\ldots,A_n\}}$ is the set of all 2^n valuations, $R \stackrel{\text{def}}{=} N \times N$ relates any two states, and for all $s \in N$, s is its own valuation: $\varepsilon(s) \stackrel{\text{def}}{=} s$. One can see that φ is satisfiable iff there is a $s \in N$ s.t. $T, s \models \varphi$. For a many-one reduction, we pick any $s_0 \in N$ and use

$$(\exists s \in N,\ T, s \models \varphi) \text{ iff } T, s_0 \models \mathsf{X}\varphi \text{ iff } T, s_0 \models \mathsf{F}\varphi.$$

The reduction is logspace since n, and then $|T|$, are constants. ∎

Prop. 3.1 is used extensively in the rest of the article. Note that the reduction does not work for an *empty* set of combinators, as could be expected since $SAT(\mathtt{L}())$ is **NP**-complete while $MC(\mathtt{L}())$ amounts to evaluating a boolean expression and is in **L** [Lyn77]. Also, Prop. 3.1 holds when n is bounded and should not be confused with the reductions from model checking into satisfiability where one uses additional propositions to encode the structure of T into a temporal formula (used in, e.g., [SC85, Eme90]).

3.1. PSPACE-hardness with few propositions

The next two propositions show that, for model checking problems, n propositional variables can be encoded into only two if U is allowed, and into only one one if F and X are allowed.

PROPOSITION 3.2. *$MC(\mathtt{L}(\mathsf{H}_1, \ldots)) \leq_L MC(\mathtt{L}_2(\mathsf{U}))$ for any set $\mathsf{H}_1, \ldots$ of PLTL temporal operators.*

Proof. With a Kripke structure $T = (N, R, \varepsilon)$ and a formula $\varphi \in \mathsf{L}(\mathsf{H}_1, \ldots)$ such that $Prop(\varphi) = \{P_1, \ldots, P_n\}$, we associate a Kripke structure $D_n(T) \stackrel{\text{def}}{=} (N', R', \varepsilon')$ over $Prop' = \{A, B\}$ given by

$$N' \stackrel{\text{def}}{=} \{\langle s, i\rangle \mid s \in N, 1 \leq i \leq 2n+2\}$$

$$\langle s, i\rangle R' \langle s', i'\rangle \stackrel{\text{def}}{\Leftrightarrow} \begin{cases} s = s' \text{ and } i' = i+1, \text{ or} \\ sRs' \text{ and } i = 2n+2 \text{ and } i' = 1, \end{cases}$$

$$\varepsilon'(\langle s, 1\rangle) \stackrel{\text{def}}{=} \{A, B\}, \qquad \varepsilon'(\langle s, 2j+1\rangle) \stackrel{\text{def}}{=} \{A\},$$

$$\varepsilon'(\langle s, 2\rangle) \stackrel{\text{def}}{=} \{\}, \qquad \varepsilon'(\langle s, 2j+2\rangle) \stackrel{\text{def}}{=} \begin{cases} \{B\} \text{ if } P_j \in \varepsilon(s), \\ \{\} \text{ otherwise.} \end{cases}$$

where $j = 1, \ldots, n$. Fig. 1 displays an example. Here alternations between A and

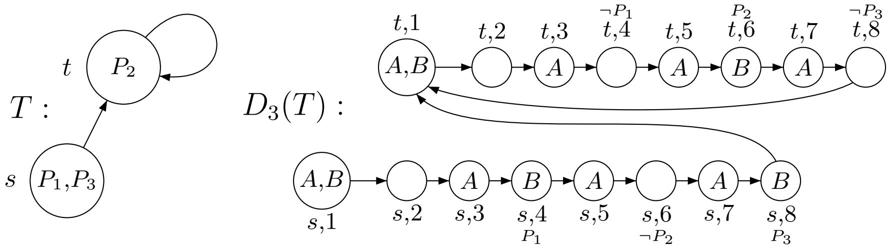


FIG. 1. T and $D_3(T)$ – An example

$\neg A$ in $D_n(T)$ define visible "slots", the $\langle s, 2j+2\rangle$'s, that are used to encode the truth value of the propositional variables: B in the i-th slot encodes that P_i holds.

Define $At_D \stackrel{\text{def}}{=} A \wedge B$, and let Alt_n^k for $k = 0, \ldots, n$ be given by

$$Alt_n^0 \stackrel{\text{def}}{=} At_D \qquad Alt_n^{k+1} \stackrel{\text{def}}{=} \neg B \wedge A \wedge \Big(A\mathsf{U}^{-}\big(\neg A \wedge (\neg A\mathsf{U}^{-} Alt_n^k)\big)\Big).$$

At_D is satisfied in $D_n(T)$ at all $\langle s, j\rangle$ with $j = 1$ and only there. Alt_n^k expresses the fact that there remain k "A–$\neg A$" alternations before the next state satisfying At_D.

We now translate formulae over T into formulae over $D_n(T)$ via the following inductive definition:

$$\begin{array}{rcl}
D_n(P_i) & \stackrel{\text{def}}{=} & A\mathsf{U}^-\Big(\neg At_D \;\wedge\; \neg At_D\mathsf{U}^-\big(Alt_n^{n+1-i} \wedge A\mathsf{U}^-B\big)\Big); \\
D_n(\varphi \wedge \varphi') & \stackrel{\text{def}}{=} & D_n(\varphi) \wedge D_n(\varphi'); \\
D_n(\neg\varphi) & \stackrel{\text{def}}{=} & \neg D_n(\varphi); \\
D_n(\mathsf{X}\varphi) & \stackrel{\text{def}}{=} & At_D\mathsf{U}^-\Big(\neg A \wedge \neg B \wedge (\neg At_D\mathsf{U}^-(At_D \wedge D_n(\varphi))\Big); \\
D_n(\mathsf{F}\varphi) & \stackrel{\text{def}}{=} & \mathsf{F}(At_D \wedge D_n(\varphi)); \\
D_n(\varphi\mathsf{U}\varphi') & \stackrel{\text{def}}{=} & (At_D \;\Rightarrow\; D_n(\varphi))\mathsf{U}(At_D \wedge D_n(\varphi')).
\end{array}$$

This gives the reduction we need since

for any $s \in N$: $T, s \models \varphi$ iff $D_n(T), \langle s, 1\rangle \models D_n(\varphi)$.

Clearly the construction of $D_n(T)$ can be done in space $\mathcal{O}(\log(|T| + |\varphi|))$ and the construction of $D_n(\varphi)$ can be done in space $\mathcal{O}(\log|\varphi|)$. ▮

Observe that $D_n(\varphi) \in \mathsf{L}_2(\mathsf{U}^-)$ when $\varphi \in \mathsf{L}(\mathsf{F}, \mathsf{X})$. Combining with Theorem 2.1 we obtain

COROLLARY 3.1. *$MC(\mathsf{L}_2(\mathsf{U}^-))$ is* **PSPACE***-complete.*

PROPOSITION 3.3. *$MC(\mathsf{L}(\mathsf{H}_1, \ldots)) \leq_L MC(\mathsf{L}_1(\mathsf{X}, \mathsf{H}_1, \ldots))$ for any set $\mathsf{H}_1, \ldots$ of PLTL temporal operators.*

Proof. With a Kripke structure $T = (N, R, \varepsilon)$ and a formula $\varphi \in \mathsf{L}(\mathsf{H}_1, \ldots)$ such that $Prop(\varphi) = \{P_1, \ldots, P_n\}$, we associate a Kripke structure $C_n(T) \stackrel{\text{def}}{=} (N', R', \varepsilon')$ over $Prop' = \{A\}$, given by

$$N' \stackrel{\text{def}}{=} \{\langle s, i\rangle : s \in N,\; 1 \leq i \leq 2n+2\},$$

$$\langle s, j\rangle R' \langle s', j'\rangle \;\stackrel{\text{def}}{\Leftrightarrow}\; s = s' \text{ and } j' = j+1, \text{ or } sRs' \text{ and } j = 2n+2 \text{ and } j' = 1,$$

$$\forall s \in N, \forall j = 1, \ldots, n: \qquad \begin{array}{rcl}
\varepsilon'(\langle s, 1\rangle) = \varepsilon'(\langle s, 2\rangle) & \stackrel{\text{def}}{=} & \{A\}, \\
\varepsilon'(\langle s, 2j+1\rangle) & \stackrel{\text{def}}{=} & \{\}, \\
\varepsilon'(\langle s, 2j+2\rangle) & \stackrel{\text{def}}{=} & \begin{cases} \{A\} \text{ if } P_j \in \varepsilon(s), \\ \{\} \text{ otherwise.} \end{cases}
\end{array}$$

Fig. 2 displays an example.

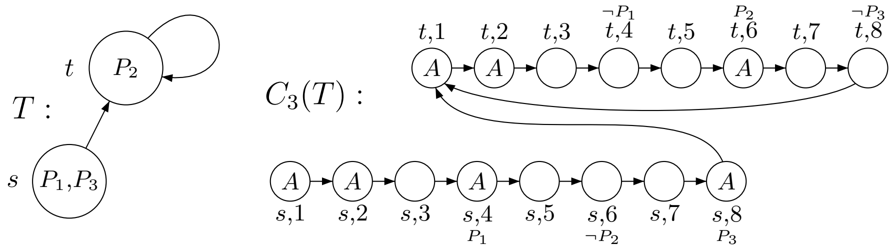


FIG. 2. T and $C_3(T)$ – An example

The idea is to use $\neg A.A$ (resp. $\neg A.\neg A$) in the i-th slot after a $A.A$ to encode that P_i holds (resp. does not hold). The $A.A$ is a marker for the beginning of some s and the $\neg A$ in a $\langle s, 2j+1\rangle$ is to distinguish slots for starting a new s and slots for a P_i. We now translate formulae over T into formulae over $C_n(T)$ via the following inductive definition:

$$\begin{aligned} C_n(P_i) &\stackrel{\text{def}}{=} \mathsf{X}^{2i+1}A, & C_n(\mathsf{X}\varphi) &\stackrel{\text{def}}{=} \mathsf{X}^{2n+2}C_n(\varphi),\\ C_n(\varphi\wedge\varphi') &\stackrel{\text{def}}{=} C_n(\varphi)\wedge C_n(\varphi'), & C_n(\mathsf{F}\varphi) &\stackrel{\text{def}}{=} \mathsf{F}(At_C\wedge C_n(\varphi)),\\ C_n(\neg\varphi) &\stackrel{\text{def}}{=} \neg C_n(\varphi), & C_n(\varphi\mathsf{U}\varphi') &\stackrel{\text{def}}{=} (At_C\Rightarrow C_n(\varphi))\mathsf{U}(At_C\wedge C_n(\varphi')). \end{aligned}$$

with $At_C \stackrel{\text{def}}{=} A\wedge\mathsf{X}A\wedge\mathsf{X}^2\neg A$. Clearly, At_C is satisfied in $C_n(T)$ at all $\langle s, j\rangle$ with $j=1$ and only there. For any $s\in N$, we have $T, s\models\varphi$ iff $C_n(T), \langle s, 1\rangle\models C_n(\varphi)$.

Finally, the construction of $C_n(T)$ can be done in space $\mathcal{O}(\log(|T|+|\varphi|))$ and the construction of $C_n(\varphi)$ can be done in space $\mathcal{O}(\log|\varphi|)$. ▌

Combining with Theorem 2.1 we obtain

COROLLARY 3.2. *$MC(\mathsf{L}_1(\mathsf{F},\mathsf{X}))$ is* ***PSPACE****-complete.*

Similar results exist for satisfiability problems:

PROPOSITION 3.4. *For $\mathsf{H}_1, \ldots$ a set of PLTL temporal operators,*
(1) $SAT(\mathsf{L}(\mathsf{H}_1,\ldots)) \leq_L SAT(\mathsf{L}_2(\mathsf{U}))$, and
(2) $SAT(\mathsf{L}(\mathsf{H}_1,\ldots)) \leq_L SAT(\mathsf{L}_1(\mathsf{F},\mathsf{X},\mathsf{H}_1,\ldots))$.

Proof. (1) Let $\varphi \in \mathsf{L}(\mathsf{H}_1,\ldots)$ be such that $Prop(\varphi)=\{P_1,\ldots,P_n\}$. Let ψ'_n be the formula

$$\begin{aligned}\psi'_n \stackrel{\text{def}}{=}\ & At_D \wedge \mathsf{G}\big(\neg A \Rightarrow (B \Rightarrow B\mathsf{U}^- A)\wedge(\neg B \Rightarrow \neg B\mathsf{U}^- A)\big)\\ & \wedge \mathsf{G}\Big[At_D \Rightarrow At_D\mathsf{U}^-\big(\neg A\wedge\neg B\wedge((\neg A\wedge\neg B)\mathsf{U}^- Alt^n_n)\big)\Big].\end{aligned}$$

ψ'_n describes the shape of models that have the form of some $D_n(S)$. More formally, one can show that for any model S, $D_n(S)\models\psi'_n$ and for any S' over $\{A,B\}$, if $S'\models\psi'_n$ then there exists a (unique) S such that $S'\approx D_n(S)$. Then an $\mathsf{L}_n(\mathsf{H}_1,\ldots)$ formula φ is satisfiable iff the $\mathsf{L}_2(\mathsf{U},\mathsf{H}_1,\ldots)$ formula $\psi'_n\wedge D_n(\varphi)$ is satisfiable. We already know that $D_n(\varphi)$ can be built in space $\mathcal{O}(\log|\varphi|)$. Moreover, ψ'_n can be also built in space $\mathcal{O}(\log|\varphi|)$ since we already know that Alt^n_n can be built in space $\mathcal{O}(\log n)$ which is *a fortiori* in space $\mathcal{O}(\log|\varphi|)$.

(2) Let $\varphi\in\mathsf{L}(\mathsf{H}_1,\ldots)$ such that $Prop(\varphi)=\{P_1,\ldots,P_n\}$. Let ψ_n be the formula

$$\psi_n \stackrel{\text{def}}{=} At_C\wedge\mathsf{G}\Big(At_C \Rightarrow (\mathsf{X}^{2n+2}At_C\wedge\bigwedge_{j=1}^{n}\mathsf{X}^{2j}\neg A)\Big).$$

ψ_n describes the shape of models of the form $C_n(S)$: for any model S, $C_n(S)\models\psi_n$ and for any S' over $\{A\}$ if $S'\models\psi_n$ then there exists a (unique) S such that S' is (isomorphic to) $C_n(S)$. Then the $\mathsf{L}_n(\mathsf{H}_1,\ldots)$ formula φ is satisfiable iff the $\mathsf{L}_1(\mathsf{F},\mathsf{X},\mathsf{H}_1,\ldots)$ formula $\psi_n\wedge C_n(\varphi)$ is satisfiable. We already know that $C_n(\varphi)$ can be built in space $\mathcal{O}(\log|\varphi|)$. Moreover, ψ_n can be also built in space $\mathcal{O}(\log|\varphi|)$ since we need to count until n which requires space in $\mathcal{O}(\log n)$. So, computing $\psi_n\wedge C_n(\varphi)$ requires space in $\mathcal{O}(\log|\varphi|)$. ▮

Since ψ'_n is a $\mathsf{L}_2(\mathsf{U}^-)$ formula, the proof of Proposition 3.4 also shows that $SAT(\mathsf{L}(\mathsf{F},\mathsf{X}))\leq_L SAT(\mathsf{L}_2(\mathsf{U}^-))$. Combining with Theorem 2.1, we get

COROLLARY 3.3. *$SAT(\mathsf{L}_2(\mathsf{U}^-))$ and $SAT(\mathsf{L}_1(\mathsf{F},\mathsf{X}))$ are **PSPACE**-complete.*

3.2. NP-hardness with few propositions

We now show that $MC(\mathsf{L}_2(\mathsf{F}))$ and $SAT(\mathsf{L}_2(\mathsf{F}))$ are **NP**-hard using Prop. 3.1 and

PROPOSITION 3.5. $SAT(\mathsf{L}^0_\omega())\leq_L SAT(\mathsf{L}_2(\mathsf{F}))$.

Proof. We consider structures on $Prop=\{A,B\}$. Say S has n A-alternations iff there exist positions $0=i_1<i'_1<i_2<i'_2<\cdots<i_{n+1}<i'_{n+1}=\omega$ such

that $S, j \models \neg A$ iff $i'_k \leq j < i_{k+1}$ for some k. Hence S contains an alternation of $2n$ consecutive non-empty segments : A holds in the first and all odd-numbered segments, A does not hold in even-numbered segments. Then there is an infinite suffix where A holds continually.

Let us define the following formulae:

- $\varphi_0 \stackrel{\text{def}}{=} \mathsf{G}(\neg A \vee \mathsf{G}A) \wedge \mathsf{F}A$;
- $\varphi_0[\varphi] \stackrel{\text{def}}{=} \top$; $\varphi_1[\varphi] \stackrel{\text{def}}{=} A \wedge \mathsf{F}(\neg A \wedge \varphi)$;
- $\varphi_{i+1}[\varphi] \stackrel{\text{def}}{=} \varphi_1[\mathsf{F}\varphi_i[\varphi]]$, for $i \geq 1$.

One can check that $\varphi_n[\varphi_0]$, $n \geq 1$, expresses that a structure has n' A-alternations for some $n' \geq n$. Thus

$$\psi_n \stackrel{\text{def}}{=} \varphi_n[\varphi_0] \wedge \neg\varphi_{n+1}[\varphi_0]$$

is a formula with size in $\mathcal{O}(n)$, stating that the model S has exactly n A-alternations. An A-alternation is a segment composed of an A-segment followed by an $\neg A$-segment. For $l \in \{A, \neg A\}$, an l-segment is a (non-empty) finite sequence of states where l holds true. Generally, $\varphi_n[\psi]$ expresses that there is $n' \geq n$ such that ψ holds at some state belonging to the n'th A-alternation in which $\neg A$ also holds.

When S has exactly n A-alternations, we can view it as the encoding of a valuation v_S of $\{P_1, \ldots, P_n\}$ by saying that P_k holds iff both B and $\neg B$ can be found in the k-th $\neg A$-segment in S. Formally, $v_S(P_k) \stackrel{\text{def}}{=} \top$ iff there exist $i'_k \leq j, j' < i_{k+1}$ with $S, j \models B$ and $S, j' \models \neg B$.

We now encode a propositional formula θ over $\{P_1, \ldots, P_n\}$ into $f_n(\theta)$, an $\mathtt{L}(\mathsf{F})$-formula with

$$f_n(P_i) \stackrel{\text{def}}{=} \varphi_i[B \wedge \mathsf{F}\varphi_{n-i}[\varphi_0]] \wedge \varphi_i[\neg B \wedge \mathsf{F}\varphi_{n-i}[\varphi_0]]$$

and the obvious homomorphic rules for $\wedge$ and $\neg$. One can see that, for S with n A-alternations, $v_S \models \theta$ iff $S \models f_n(\theta)$, so that θ is satisfiable iff $f_n(\theta) \wedge \psi_n$ is satisfiable.

The proof is completed by checking that $f_n(\theta) \wedge \psi_n$ is an $\mathtt{L}^{\omega}_2(\mathsf{F})$-formula that can be computed from θ in space $\mathcal{O}(\log |\theta|)$. ∎

The transformation from 3SAT into $MC(\mathtt{L}(\mathsf{F}))$ in [SC85] only uses formulae of temporal height 1. Here we provide a logspace transformation from 3SAT into $MC(\mathtt{L}(\mathsf{F}))$ using only formulae with two different propositional variables.

PROPOSITION 3.6. *3SAT* $\leq_L MC(\mathtt{L}^{\omega}_2(\mathsf{F}))$.

Proof. Consider an instance $\mathcal{I}$ of 3SAT. $\mathcal{I}$ is a conjunction $\bigwedge_{i=1}^{m} C_i$ of clauses, where each C_i is some disjunction $\bigvee_{j=1}^{3} l_{i,j}$ of literals, where each $l_{i,j}$ is a propositional variable $x_{r(i,j)}$ or the negation $\neg x_{r(i,j)}$ of a propositional variable from $X = \{x_1, \ldots, x_n\}$. W.l.o.g. we assume that $n \leq 3 \times m$ and that, for any i, the $r(i,j)$ for $1 \leq j \leq 3$ are all distinct.

We consider the structure T_n labeled with propositions A and B as in Figure 3. Observe that T_n only depends on n, the number of different boolean variables occuring in $\mathcal{I}$.

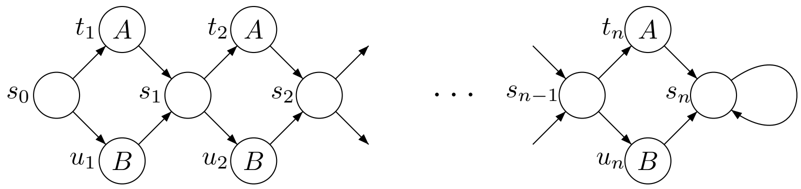


FIG. 3. The structure T_n

With a path S from s_0, we associate a valuation $v_S \in \{\top, \bot\}^X$: if S visits t_r (resp. u_r), we let $v_S(x_r) \stackrel{\text{def}}{=} \top$ (resp. $v_S(x_r) \stackrel{\text{def}}{=} \bot$). Symmetrically, any valuation v is v_S for a unique path S in T_n.

For $i = 1, \ldots, m$ we define φ_i^0, an $\mathtt{L}(\mathsf{F})$ formula stating that v_S does not satisfy clause C_i. This is done in several steps: define

$$\varphi_i^{n+1} \stackrel{\text{def}}{=} \neg \mathsf{F}(A \vee B)$$

and, for $r = 1, \ldots, n$, define inductively

$$\varphi_i^r \stackrel{\text{def}}{=} \begin{cases} \neg(A \vee B) \wedge \mathsf{F}(B \wedge \mathsf{F}\varphi_i^{r+1}) & \text{if } l_{i,j} = x_r \text{ for some } 1 \leq j \leq 3, \\ \neg(A \vee B) \wedge \mathsf{F}(A \wedge \mathsf{F}\varphi_i^{r+1}) & \text{if } l_{i,j} = \neg x_r \text{ for some } 1 \leq j \leq 3, \\ \neg(A \vee B) \wedge \mathsf{F}((A \vee B) \wedge \mathsf{F}\varphi_i^{r+1}) & \text{if no } l_{i,j} \text{ is } x_r \text{ or its negation.} \end{cases}$$

Because it involves alternations between $\neg(A \vee B)$ and $A \vee B$, φ_i^r cannot be satisfied starting from $s_{n-r'}$ for $r' > r$. Thus, if $S \models \varphi_i^0$, the rth positive occurence of A or B or $A \vee B$ is necessarily satisfied in t_r or u_r. Hence

$$S \models \varphi_i^0 \text{ iff } v_S \not\models C_i.$$

Now define $\varphi_{\mathcal{I}} \stackrel{\text{def}}{=} \bigwedge_{i=1}^{m} \neg\varphi_i^0$. Then $T_n, s_0 \models \varphi_{\mathcal{I}}$ iff $\mathcal{I}$ is satisfiable. Finally, both T_n and $\varphi_{\mathcal{I}}$ can be computed in space $\mathcal{O}(\log |\mathcal{I}|)$. ▌

COROLLARY 3.4. *$MC(\mathtt{L}_2(\mathsf{F}))$ and $SAT(\mathtt{L}_2(\mathsf{F}))$ are* **NP***-complete.*

4. FRAGMENTS WITH ONLY ONE PROPOSITION

In this section, we give a polynomial-time algorithm for $\mathtt{L}_1^\omega(\mathsf{U})$ that relies on linear-sized Büchi automata.

Recall that the standard approach for PLTL satisfiability and model checking computes, for a given PLTL formula φ, a Büchi automaton [7] $\mathcal{A}_\varphi$ recognizing exactly the models of φ (the alphabet of the Büchi automaton is the set of possible valuations for the propositional variables from φ).

Satisfiability of φ is non-emptiness of $\mathcal{A}_\varphi$. Checking whether a path in some T satisfies φ is done by computing a synchronous product of T and $\mathcal{A}_\varphi$ and checking for non-emptiness of the resulting system (a larger Büchi automaton) This method was first presented in [WVS83], where a first algorithm for computing $\mathcal{A}_\varphi$ was given.

The complexity of this approach comes from the fact that $\mathcal{A}_\varphi$ can have exponential size. Indeed, once we have $\mathcal{A}_\varphi$ the rest is easy:

LEMMA 4.1. **[Var94]** *It is possible, given a Büchi automaton $\mathcal{A}$ recognizing the models of formula φ, and a Kripke structure T, to say in non deterministic space $\mathcal{O}(\log|T| + \log|\mathcal{A}|)$ whether there is a computation in T accepted by $\mathcal{A}$.*

From these remarks, it easily follows that fragments of PLTL will have low complexity if the corresponding $\mathcal{A}_\varphi$ are small.

4.1. The fragment $\mathtt{L}_1^\omega(\mathsf{U})$

Here we consider a single proposition: $Prop = \{A\}$. Any linear model is equivalent, modulo stuttering, to one of the following: for $n \in \mathbb{N}$

$$
\begin{array}{lll}
S_1^n \stackrel{\text{def}}{=} (A.\neg A)^n.A^\omega, & S_2^n \stackrel{\text{def}}{=} \neg A.(A.\neg A)^n A^\omega, & S_3^n \stackrel{\text{def}}{=} (A.\neg A)^\omega, \\
S_4^n \stackrel{\text{def}}{=} (\neg A.A)^n.\neg A^\omega, & S_5^n \stackrel{\text{def}}{=} A.(\neg A.A)^n \neg A^\omega, & S_6^n \stackrel{\text{def}}{=} (\neg A.A)^\omega,
\end{array}
$$

where S_3^n and S_6^n do not depend on n.

Now a satisfiable $\mathtt{L}_1^\omega(\mathsf{U},\mathsf{X})$ formula is satisfiable in some S_i^n with small n:

LEMMA 4.2. *For any $i = 1,\ldots,6$, $\varphi \in \mathtt{L}_1^\omega(\mathsf{U},\mathsf{X})$ and $n \geq th(\varphi)$, $S_i^{n+1} \models \varphi$ iff $S_i^n \models \varphi$.*

[7] or a Muller automaton, or an alternating Büchi automaton, or ...

Proof. By structural induction on φ and using the fact that the first suffix of a S_i^n is a $S_j^{n'}$ with $n-1 \leq n' \leq n$, e.g. the first suffix of S_1^n is S_2^{n-1} (for $n > 0$) and the first suffix of S_2^n is S_1^n. ▮

Recognizing the S_i^n's is easy:

LEMMA 4.3. *For any $1 \leq i \leq 6$ and $n \in \mathbb{N}$, there exists a Büchi automaton $\mathcal{A}_i^{=n}$ and a Büchi automaton $\mathcal{A}_i^{\geq n}$ s.t. $\mathcal{A}_i^{=n}$ (resp. $\mathcal{A}_i^{\geq n}$) accepts a model S iff $S \approx S_i^n$ (resp. $S \approx S_i^m$ for some $m \geq n$).*
Furthermore, the $\mathcal{A}_i^{=n}$'s and $\mathcal{A}_i^{\geq n}$'s have $\mathcal{O}(n)$ states and can be generated uniformly using $\log n$ space.

Proof. We only show $\mathcal{A}_1^2$, $\mathcal{A}_1^{\geq 2}$ and $\mathcal{A}_3^{=n}$ as examples (see Fig. 4.1).

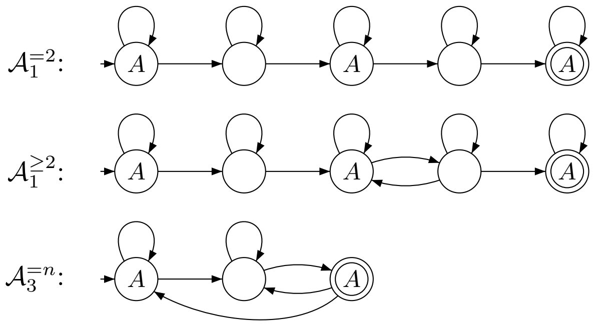


FIG. 4. Büchi automata for Lemma 4.3

▮

Combining lemmas 4.1 and 4.3, we see that the problem of deciding, given T with s_0 a state, given $n \in \mathbb{N}$ and $1 \leq i \leq 6$, whether there is a path S in T that starts from s_0 and s.t. $S \approx S_i^n$, can be solved in non deterministic space $\mathcal{O}(\log(n \times |T|))$ or in deterministic time $\mathcal{O}(n \times |T|)$. Similarly, the problem of deciding whether there is a path S and a $m \geq n$ s.t. $S \approx S_i^m$ can be solved with same complexity.

THEOREM 4.1. *Model checking for $\mathtt{L}_1^{\omega}(\mathsf{U})$ is in* **P**.

Proof. Consider a Kripke structure $T = (N, R, \varepsilon)$ and some state $s_0 \in N$. If there is a path S from s_0 satisfying $\varphi \in \mathtt{L}_1^{\omega}(\mathsf{U})$ then $S \approx S_i^n$ for some $n \in \mathbb{N}$ and some $i = 1, \ldots, 6$ and $S_i^n \models \varphi$. Conversely, if $S_i^n \models \varphi$ and there is a path $S \approx S_i^n$ starting from s_0, then $T, s_0 \models \varphi$.

It is possible to check whether T contains such a path in polynomial-time: We consider all S_i^k for $k < th(\varphi)$. When $S_i^k \models \varphi$, seen in time $\mathcal{O}(k.|\varphi|)$, we check in time $\mathcal{O}(k.|T|)$, whether, from s_0, T admits a path $S \approx S_i^k$. We also consider all S_i^k for $k = th(\varphi)$. When $S_i^k \models \varphi$ we know that $S_i^{k+m} \models \varphi$ for all m (Lemma 4.2) so that it is correct to check whether there is a m such that T admits a path $S \approx S_i^{k+m}$. Because $k \leq |\varphi|$, the complete algorithm only needs $\mathcal{O}(|T| \times |\varphi|^2)$-time. ▮

Remark 4. 1. We do not know whether $MC(\mathtt{L}_1^\omega(\mathsf{U}))$ is **P**-hard. We only know it is **NL**-hard [8]. The same open question applies to $SAT(\mathtt{L}_1^\omega(\mathsf{U}))$.

Looking at the algorithm used in the proof of Theo. 4.1, it appears [9] that this open question is linked to an important open problem that remained unnoticed for many years:

OPEN PROBLEM 4.1. *What is the complexity of model checking a path?*

Here a "path" is a finitely presented linear-time structure. It can be given by a deterministic Kripke structure (i.e., where any state has exactly one successor) or by an ω-regular expression $u.v^\omega$ where u and v are finite sequences of valuations. Model checking a path is clearly in **P** but it is not known whether it is **P**-hard or in **NL** or somewhere in between.

4.2. The fragment $\mathtt{L}_1^\omega(\mathsf{X})$

PROPOSITION 4.2. $SAT(\mathtt{L}_1^\omega(\mathsf{X}))$ *and* $MC(\mathtt{L}_1^\omega(\mathsf{X}))$ *are* **NP**-*complete.*

Proof. Satisfiability for $\mathtt{L}(\mathsf{X})$ is in **NP** because, for $\varphi \in \mathtt{L}(\mathsf{X})$ with temporal height k, it is enough to guess the first k states of a witness S. Model checking also is in **NP** for the same reason.

NP-hardness of $SAT(\mathtt{L}_1^\omega(\mathsf{X}))$ can be shown by a reduction from 3SAT: consider a boolean formula θ with propositional variables $P_1, \ldots, P_n$ and replace the P_i's by $\mathsf{X}^i A$'s: the resulting $\mathtt{L}_1^\omega(\mathsf{X})$ formula is satisfiable iff θ is. Then, by Prop. 3.1, $MC(\mathtt{L}_1^\omega(\mathsf{X}))$ is **NP**-hard too. ▮

PROPOSITION 4.3. *For any* $k, n < \omega$, $SAT(\mathtt{L}_n^k(\mathsf{U}, \mathsf{X}))$ *is in* **L**.

Proof. Here the key observation is that *there are only a finite number of essentially distinct formulae in a given fragment* $\mathtt{L}_n^k(\mathsf{U}, \mathsf{X})$. Given n and k, one can compute once and for all a finite subset $J_n^k = \{\psi_1, \ldots, \psi_N\}$ of $\mathtt{L}_n^k(\mathsf{U}, \mathsf{X})$ such that

[8] One easily shows that already $MC(\mathtt{L}_1^1(\mathsf{F}))$ is **NL**-hard by a reduction from GAP, the graph accessibility problem of [Jon75].

[9] M. Y. Vardi pointed out the connection to us.

1. any $\varphi \in \mathtt{L}^k_n(\mathsf{U},\mathsf{X})$ is equivalent to a $\psi_i \in J^k_n$ (we say ψ_i is the *canonical representative* for φ);

2. for $i \neq j$, ψ_i and ψ_j are not equivalent. Then a given φ is satisfiable iff its canonical representative is not the canonical representative of $\perp$.

Any J^k_n is finite and, more precisely, $|J^0_n| = 2^{2^n}$ and $|J^{k+1}_n|$ is in $2^{2^{\mathcal{O}(|J^k_n|^2)}}$.

We assume n and k are fixed and we consider the problem, given φ, of computing its canonical representative (or equivalently its index $1 \leq i \leq N$). This can be done in a compositional way: if $\varphi \equiv \psi_i$ and $\varphi' \equiv \psi_j$ then the representative ψ_k of $\varphi \mathsf{U} \varphi'$ (say) is the representative of $\psi_i \mathsf{U} \psi_j$, so that we just need to compute once and for all a finite table $t_{\mathsf{U}} : (i,j) \mapsto k$, and similar tables t_{X}, $t_\wedge$, $t_\neg$, ..., for all operators, temporal or boolean.

Once we have these tables, computing the canonical representative of any $\varphi \in \mathtt{L}^k_n(\mathsf{U},\mathsf{X})$ amounts to evaluating an expression over a fixed finite domain, which can be done in logspace (see [Lyn77]). ▮

PROPOSITION 4.4. *For any $k, n < \omega$, $MC(\mathtt{L}^k_n(\mathsf{U},\mathsf{X}))$ is in* **NL**.

Proof. As in the proof of Prop. 4.3, for $\varphi \in \mathtt{L}^k_n(\mathsf{U},\mathsf{X})$ we compute in logspace a canonical representative $\psi_i \in J^k_n$. By Lemma 4.1, checking whether $T, s \models \psi_i$ can be done in non deterministic space $\mathcal{O}(\log|T| + \log|\mathcal{A}_{\psi_i}|)$. Since n and k are fixed, $\max\{|\mathcal{A}_{\psi_i}| : i \in \{1,\ldots,N\}\}$ is a constant, so that $MC(\mathtt{L}^k_n(\mathsf{U},\mathsf{X}))$ is in **NL**. ▮

Since $MC(\mathtt{L}^1_1(\mathsf{F}))$ is **NL**-hard (Remark 4.1), we get

COROLLARY 4.1. *For any $1 \leq k < \omega$ and $1 \leq n < \omega$, for any set $\mathsf{H}_1, \ldots$ of PLTL temporal operators, $MC(\mathtt{L}^k_n(\mathsf{F},\mathsf{H}_1,\ldots))$ is* **NL**-*complete.*

By contrast, by [Lyn77], $MC(\mathtt{L}^0_\omega(\mathsf{U},\mathsf{X}))$ is in **L**.

This concludes the study of all fragments with a bounded number of propositions. In the remaining of the article, this bound is removed.

5. FROM QBF TO $MC(\mathtt{L}(\mathsf{U}))$

In this section, we offer a logspace transformation from validity of Quantified Boolean Formulae (QBF) into model checking for $\mathtt{L}(\mathsf{U})$ that involves rather simple constructions of models and formulae. This reduction can be adapted to various fragments and, apart from the fact that it offers a simple means to get **PSPACE**-hardness, we obtain a new master reduction from a well-known logical problem.

As a side-effect, we establish that $MC(\mathtt{L}^2_\omega(\mathsf{U}^-))$ is **PSPACE**-hard, which is not subsumed by any reduction from the literature.

Consider an instance $\mathcal{I}$ of QBF. It has the form

$$\mathcal{I} \equiv Q_1 x_1 \ \dots \ Q_n x_n \overbrace{\wedge_{i=1}^{m} \vee_{j=1}^{k_i} l_{i,j}}^{\mathcal{I}_0}$$

where every Q_r ($1 \leq r \leq n$) is a universal, $\forall$, or existential, $\exists$, quantifier. $\mathcal{I}_0$ is a propositional formula without any quantifier. Here we consider w.l.o.g. that $\mathcal{I}_0$ is a conjunction of clauses, i.e. every $l_{i,j}$ is a propositional variable $x_{r(i,j)}$ or the negation $\neg x_{r(i,j)}$ of a propositional variable from $X = \{x_1, \dots, x_n\}$. The question is to decide whether $\mathcal{I}$ is valid or not. Recall that

LEMMA 5.1. *$\mathcal{I}$ is valid iff there exists a non-empty set $\mathcal{V} \subseteq \{\top, \bot\}^X$ of valuations such that*
correctness: *$\forall v \in \mathcal{V}, v \models \mathcal{I}_0$, and*
closure: *for all $v \in \mathcal{V}$, for all r such that $Q_r = \forall$, there is a $v' \in \mathcal{V}$ such that $v'[x_r] \neq v[x_r]$ and for all $r' < r$, $v'[x_{r'}] = v[x_{r'}]$.*

With $\mathcal{I}$ we associate the Kripke structure $T_{\mathcal{I}}$ as given in Figure 5, using labels from $Prop = \{A_0, A_1, \dots, x_1^T, \dots, L_1^1, \dots\}$. Assume S is an infinite path starting

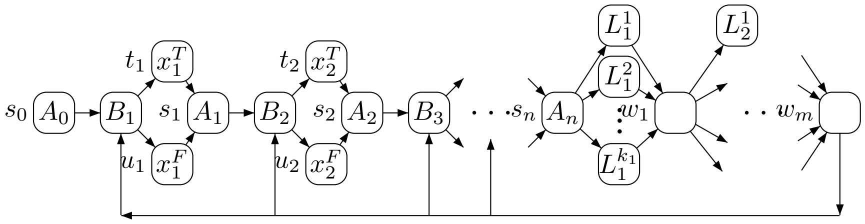


FIG. 5. The structure $T_{\mathcal{I}}$ associated with $\mathcal{I} \equiv Q_1 x_1 \ \dots \ Q_n x_n \wedge_{i=1}^{m} \vee_{j=1}^{k_i} l_{i,j}$

from s_0. Between s_0 and s_n, it picks a boolean valuation for all variables in X, then reaches w_m and goes back to some B_r-labeled state ($1 \leq r \leq n$) where (possibly distinct) valuations for $x_r, x_{r+1}, \dots, x_n$ are picked.

In S, at any position lying between a s_n and the next w_m, we have a notion of *current valuation* which associates $\top$ or $\bot$ with any x_r depending on *the latest u_r or t_r node* we visited. With S we associate the set $\mathcal{V}(S)$ of all valuations that are current at positions where S visits s_n (there are infinitely many such positions).

Now consider some r with $Q_r = \forall$ and assume that whenever S visits s_{r-1} then it visits both t_r and u_r before any further visit to s_{r-1}. In $\mathtt{L}(\mathsf{U})$, this can be written $S \models \psi_r$ with ψ_r given by

$$\psi_r \stackrel{\text{def}}{=} \mathsf{G}\Big(A_{r-1} \;\Rightarrow\; (\neg B_{r-1}\mathsf{U}x_r^T) \wedge (\neg B_{r-1}\mathsf{U}x_r^F)\Big).$$

Let $\psi_{\text{clo}} \stackrel{\text{def}}{=} \bigwedge\{\psi_r \mid Q_r = \forall\}$: if S satisfies ψ_{clo}, then $\mathcal{V}(S)$ is *closed* in the sense of Lemma 5.1.

Now, whenever S visits a L_i^j-state, we say it agrees with the current valuation v if $v \models l_{i,j}$. This too can be written in $\mathsf{L}(\mathsf{U})$, using the fact that the current valuation for x_r cannot be changed without first visiting the B_r-state. For $i = 1, \ldots, m$, for $j = 1, \ldots, k_i$, let

$$\psi_{i,j} \stackrel{\text{def}}{=} \begin{cases} \mathsf{G}[x_r^F \;\Rightarrow\; \mathsf{G}\neg L_i^j \vee \neg L_i^j \mathsf{U} B_r] & \text{if } l_{i,j} = x_r, \\[2ex] \mathsf{G}[x_r^T \;\Rightarrow\; \mathsf{G}\neg L_i^j \vee \neg L_i^j \mathsf{U} B_r] & \text{if } l_{i,j} = \neg x_r. \end{cases}$$

Let $\psi_{\text{corr}} \stackrel{\text{def}}{=} \bigwedge_{i=1}^m \bigwedge_{j=1}^{k_i} \psi_{i,j}$: if S satisfies ψ_{corr}, then $\mathcal{V}(S)$ is *correct* in the sense of Lemma 5.1.

LEMMA 5.2. *Let $\varphi_{\mathcal{I}} \stackrel{\text{def}}{=} \psi_{\text{clo}} \wedge \psi_{\text{corr}}$. Then $T_{\mathcal{I}}, s_0 \models \varphi_{\mathcal{I}}$ iff $\mathcal{I}$ is valid.*

Proof. If $S \models \varphi_{\mathcal{I}}$, then $\mathcal{V}(S)$ is non-empty, closed and correct for $\mathcal{I}$ so that $\mathcal{I}$ is valid. Conversely, if $\mathcal{I}$ is valid, there exists a validating $\mathcal{V}$ (Lemma 5.1). From $\mathcal{V}$ one can build an infinite path S starting from s_0 such that $\mathcal{V}(S) = \mathcal{V}$ and $S \models \varphi_{\mathcal{I}}$: from a lexicographical enumeration of $\mathcal{V}$, S is easily constructed so that $S \models \psi_{\text{clo}}$. Then, to ensure $S \models \psi_{\text{corr}}$, between any visit to s_n and to the next w_m, S only visits L_i^j-states validated by the current valuation v, which is possible because $v \models \mathcal{I}_0$. ∎

It is worth observing that $\varphi_{\mathcal{I}}$ belongs to $\mathsf{L}_\omega^2(\mathsf{U}^-)$. Now, because both $T_{\mathcal{I}}$ and $\varphi_{\mathcal{I}}$ can be computed from $\mathcal{I}$ in logspace, and because $th(\varphi_{\mathcal{I}}) \leq 2$ (and using Prop. 3.2), we get

COROLLARY 5.1. QBF $\leq_L MC(\mathsf{L}_\omega^2(\mathsf{U}^-)) \leq_L MC(\mathsf{L}_2^\omega(\mathsf{U}^-))$.

COROLLARY 5.2. *$MC(\mathsf{L}_\omega^2(\mathsf{U}^-))$ and $MC(\mathsf{L}_2^\omega(\mathsf{U}^-))$ are* **PSPACE**-*hard.*

6. FROM QBF TO $MC(\mathsf{L}(\mathsf{F}, \mathsf{X}))$

As in section 5, we consider an instance $\mathcal{I} \equiv Q_1 x_1 \ldots Q_n x_n \wedge_{i=1}^m \vee_{j=1}^{k_i} l_{i,j}$ of QBF. With $\mathcal{I}$ we associate the Kripke structure $T'_{\mathcal{I}}$ given in Figure 6. Here, any path S starting from s_0 can be seen as an infinite succession of segments of length $K \stackrel{\text{def}}{=} 2n + 2m + 1$. Each segment directly yields a valuation for X: they form an infinite sequence $v_1, v_2, \ldots$ (necessarily with repetitions) and we let $\mathcal{V}(S)$ denote the associated set.

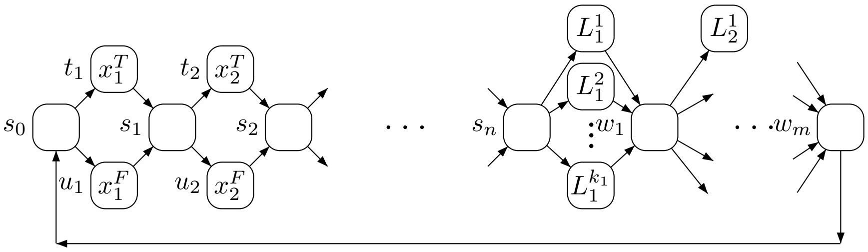


FIG. 6. The structure $T'_{\mathcal{I}}$ associated with $\mathcal{I} \equiv Q_1 x_1 \ \ldots \ Q_n x_n \wedge_{i=1}^{m} \vee_{j=1}^{k_i} l_{i,j}$

Using F and X, it is easy to state that any segment in S visits the L_i^j-states in a way that agrees with the corresponding valuation. For $i = 1, \ldots, m$, for $j = 1, \ldots, k_i$, let

$$\psi_{i,j} \stackrel{\text{def}}{=} \begin{cases} \mathsf{G}[x_r^F \ \Rightarrow \ \neg \mathsf{X}^{2(n-r+i)} L_i^j] & \text{if } l_{i,j} = x_r, \\ \mathsf{G}[x_r^T \ \Rightarrow \ \neg \mathsf{X}^{2(n-r+i)} L_i^j] & \text{if } l_{i,j} = \neg x_r. \end{cases}$$

Now $S \models \bigwedge_{i=1}^{m} \bigwedge_{j=1}^{k_i} \psi_{i,j}$ implies that $\mathcal{V}(S)$ is correct in the sense of Lemma 5.1.

There remains to enforce closure of $\mathcal{V}(S)$. For this, we require that the valuations $v_1, v_2, \ldots$ are visited according to the lexicographical ordering, and then cycling. This means that the successive choices of truth values for universally quantified propositional variables behave as the successive binary digits of counting modulo $2^{n'}$ (assuming there are n' universal quantifiers in $Q_1, \ldots, Q_n$). As usual, the existentially quantified variables are free to vary when an earlier variable varied.

Assume $Q_r = \forall$. When moving from a valuation v_t to its successor v_{t+1}, we require that $v_t(x_r)$ remains unchanged iff for some $r' > r$ with $Q_{r'} = \forall$ we have $v_t(x_{r'}) = \perp$. This is written

$$\psi_r \stackrel{\text{def}}{=} \mathsf{G}\Big((x_r^T \vee x_r^F) \ \Rightarrow \ \Big(\bigvee_{\substack{r' > r \\ Q_{r'} = \forall}} \mathsf{X}^{2r' - 2r} x_{r'}^F \ \Leftrightarrow \ \overbrace{(x_r^T \ \Leftrightarrow \ \mathsf{X}^K x_r^T)}^{\text{“}v(x_r)\textit{ does not change}\text{”}} \Big) \Big)$$

If $S \models \bigwedge \{\psi_r \mid Q_r = \forall\}$ then, restricted to the universally quantified variables, $v_1, v_2, \ldots$ behaves like counting modulo $2^{n'}$.

Assume now that $Q_{r'} = \exists$. When moving from v_t to its successor, $v_t(x_{r'})$ may not change unless $v_t(x_r)$ changes for some $r < r'$ with $Q_r = \forall$, or equivalently unless $v_t(x_r)$ changes for the latest $r < r'$ with $Q_r = \forall$ (thanks to our assumption about counting). Equivalently, this means that if for a universally quantified x_r, $v_t(x_r)$ does not change, then for any following existentially quantified $x_{r'}$, $v_t(x_{r'})$ does not change either. By "following" we mean that there is no other $\forall$ between Q_r and $Q_{r'}$, i.e. that $r' \in sc(r)$ with

$$sc(r) \stackrel{\text{def}}{=} \{r' > r \mid Q_h = \exists \text{ for all } r < h \leq r'\}.$$

This behaviour can be written:

$$\psi'_r \stackrel{\text{def}}{=} \mathsf{G}\Big((x_r^T \vee x_r^F) \;\Rightarrow\; \overbrace{(x_r^T \;\Leftrightarrow\; \mathsf{X}^K x_r^T)}^{\text{“}\textit{if } v(x_r) \textit{ does not change}\text{”}} \;\Rightarrow \bigwedge_{r' \in sc(r)} \mathsf{X}^{2r'-2r} \overbrace{(x_{r'}^T \;\Leftrightarrow\; \mathsf{X}^K x_{r'}^T)}^{\text{“}\textit{then } v(x'_r) \textit{ does not change}\text{”}}\Big).$$

Now we define

$$\varphi_{\mathcal{I}} \stackrel{\text{def}}{=} \Big(\bigwedge_{i=1}^{m} \bigwedge_{j=1}^{k_i} \psi_{i,j}\Big) \;\wedge\; \Big(\bigwedge_{\substack{r=1\\ Q_r=\forall}}^{n} \psi_r \wedge \psi'_r\Big).$$

LEMMA 6.1. *$T'_{\mathcal{I}}, s_0 \models \varphi_{\mathcal{I}}$ iff $\mathcal{I}$ is valid.*

Proof. If $S \models \varphi_{\mathcal{I}}$ then $\mathcal{V}(S)$ validates $\mathcal{I}$ as we explained. Conversely, if some $\mathcal{V}$ validates $\mathcal{I}$, then, enumerating $\mathcal{V}$ in lexicographical order, it is easy to build a S such that $S \models \varphi_{\mathcal{I}}$. ▌

Now, because $T'_{\mathcal{I}}$ and $\varphi'_{\mathcal{I}}$ can be computed from $\mathcal{I}$ in logspace (and using Prop. 3.3) we get

COROLLARY 6.1. QBF $\leq_L MC(\mathsf{L}(\mathsf{F},\mathsf{X})) \leq_L MC(\mathsf{L}_1^\omega(\mathsf{F},\mathsf{X}))$.

COROLLARY 6.2. *$MC(\mathsf{L}_1^\omega(\mathsf{F},\mathsf{X}))$ is* **PSPACE**-*hard.*

7. BOUNDING THE TEMPORAL HEIGHT

In this section we investigate the complexity of satisfiability and model checking when the temporal height is bounded. From Section 5, we already know that

$MC(\mathsf{L}^2_\omega(\mathsf{U}^-))$ is **PSPACE**-hard.

We first consider ways of reducing the temporal height (sections 7.1 and 7.2). Then we show how to improve the upper bounds when temporal height is below 2 (sections 7.3 and 7.4).

7.1. Elimination of X for model checking

Assume T is a Kripke structure and $k \in \mathbb{N}$. It is possible to partially unfold T into a Kripke structure T^k where a state $\overline{s}$ (in T^k) codes for a state s_0 in T with the k next states $s_1, \ldots, s_k$ already chosen. In T^k, $\overline{s}$ is labeled with new propositions encoding the fact that some s_i's satisfy some A_j's.

Formally, let $k \in \mathbb{N}$ and $Prop = \{A_1, \ldots, A_n\}$. First let $Prop^k \stackrel{\text{def}}{=} \{A^i_j : 1 \leq j \leq n,\ 0 \leq i \leq k\}$. Assume $T = (N, R, \varepsilon)$. Then T^k is defined as the Kripke structure $(N^k, R^k, \varepsilon^k)$ with

- $N^k \stackrel{\text{def}}{=} \{\langle s_0, \ldots, s_k\rangle : \forall i \in \{0, \ldots, k-1\}\ \langle s_i, s_{i+1}\rangle \in R\}$;
- $\varepsilon^k(\langle s_0, \ldots, s_k\rangle) \stackrel{\text{def}}{=} \{A^i_j : 0 \leq i \leq k,\ 1 \leq j \leq n,\ A_j \in \varepsilon(s_i)\}$, and
- $\langle\langle s_0, \ldots, s_k\rangle, \langle s'_0, \ldots, s'_k\rangle\rangle \in R^k \stackrel{\text{def}}{\Leftrightarrow} \langle s_0, s'_0\rangle \in R$ and for any $j \in \{1, \ldots, k\}$, $s_j = s'_{j-1}$.

This peculiar unraveling is also called *bulldozing* (see e.g. [Seg71]). Fig. 7 contains a simple example. Observe that $|T^k|$ is in $\mathcal{O}(|T|^{k+1})$ and T^k can be computed in

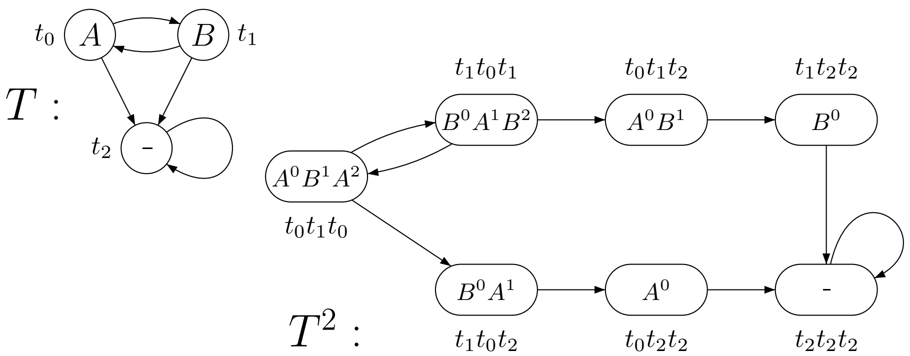


FIG. 7. An example of bulldozing: T and T^2 side by side

space $\mathcal{O}(\log(k + |T|))$.

Say a formula φ *has inner-nexts* if all occurrences of X are in subformulae of the form $\mathsf{XX}\ldots\mathsf{X}A$ (where A is a propositional variable).

If now φ has inner-nexts, with at most k nested X, and if we replace all $\mathsf{X}^i A_j$ in φ by propositions A^i_j, we obtain a new formula, denoted φ^k, such that

$$T, s \models \varphi \text{ iff } T^k, \overline{s} \models \varphi^k \text{ for some } \overline{s} \text{ starting with } s. \quad (1)$$

Both T^k and φ^k can be computed in space $\mathcal{O}(\log(|T| + |\varphi|))$.

Not all formulae have inner-nexts but, using the following equivalences

$$\mathsf{X}\neg\varphi \equiv \neg\mathsf{X}\varphi \qquad \mathsf{X}(\varphi \wedge \psi) \equiv (\mathsf{X}\varphi) \wedge (\mathsf{X}\psi) \qquad \mathsf{X}(\varphi \mathsf{U} \psi) \equiv (\mathsf{X}\varphi)\mathsf{U}(\mathsf{X}\psi)$$

as left-to-right rewrite-rules, it is possible to translate any PLTL formula into an equivalent one with inner-nexts. This translation may involve a quadratic blow-up in size but it does not modify the number of propositional variables or the temporal height of the formula [10].

COROLLARY 7.1. *For any $k \in \mathbb{N}$ and set $\mathsf{H}_1, \ldots$ of PLTL temporal combinators,* $MC(\mathsf{L}^k_\omega(\mathsf{X}, \mathsf{H}_1, \ldots)) \leq_L MC(\mathsf{L}^k_\omega(\mathsf{H}_1, \ldots))$.

Proof. Given φ in $\mathsf{L}^k_\omega(\mathsf{X}, \ldots)$, and some T, we transform φ into some equivalent ψ with inner-nexts and then evaluate ψ^k on T^k. ∎

COROLLARY 7.2. $MC(\mathsf{L}^k_\omega(\mathsf{X}))$ *is in* **L** *and* $MC(\mathsf{L}^k_\omega(\mathsf{F}, \mathsf{X}))$ *is in* **NP** *for any fixed* $k \geq 0$.

$MC(\mathsf{L}^1_\omega(\mathsf{F}))$ is **NP**-hard as can be seen from the proof of **NP**-hardness of $MC(\mathsf{L}^\omega_\omega(\mathsf{F}))$ in [SC85]. Hence for $k \geq 1$, $MC(\mathsf{L}^k_\omega(\mathsf{F}, \mathsf{X}))$ is **NP**-complete.

7.2. Elimination of X for satisfiability

Elimination of X for satisfiability relies on the same ideas. If φ is satisfiable, then, thanks to (1), φ^k is. The converse is not true: consider φ given as $\mathsf{G}A \wedge \mathsf{G}\neg\mathsf{X}A$, clearly not satisfiable. Here φ^1 is $\mathsf{G}A^0 \wedge \mathsf{G}\neg A^1$ which is satisfiable. This is because if φ^k is satisfiable, then it may be satisfiable in a model that is not a S^k for some S. But, using an $\mathsf{L}^2_\omega(\mathsf{F}, \mathsf{X})$ formula, we can express the fact that a given model is a S^k, so that

$$\varphi \text{ is satisfiable iff } \varphi^k \wedge \mathsf{G}(\textstyle\bigwedge_{j=1}^n \bigwedge_{i=1}^k A^i_j \Leftrightarrow \mathsf{X}A^{i-1}_j) \text{ is.}$$

Actually, this approach based on standard renaming techniques can get us further. We write $\varphi\{\psi \leftarrow A\}$ to denote a formula obtained by replacing all occurences

[10] These rules may introduce X's in the right-hand side of U^-'s but this will be repaired when we later replace the $X^i A_j$'s with the A^i_j's.

of ψ with A inside φ. If A does not occur in φ, then

$$\varphi \text{ is satisfiable iff } \varphi\{\psi \leftarrow A\} \wedge \mathsf{G}(A \Leftrightarrow \psi) \text{ is.}$$

By using this repeatedly and systematically, we can remove (by renaming) all subformulae ψ s.t. (1) $th(\psi) = 1$ and, (2) there exists at least one occurence of ψ in φ that is under the scope of two temporal combinators (or in the left-hand side of a U). For example, $\mathsf{F}(A\mathsf{U}(\mathsf{FG}B \Rightarrow \mathsf{G}B))$ is replaced by $\mathsf{F}(A\mathsf{U}(\mathsf{F}A^1_{new} \Rightarrow A^1_{new})) \wedge \mathsf{G}(A^1_{new} \Leftrightarrow \mathsf{G}B)$ in turn replaced by $\mathsf{F}(A\mathsf{U}(A^2_{new} \Rightarrow A^1_{new})) \wedge \mathsf{G}(A^1_{new} \Leftrightarrow \mathsf{G}B) \wedge \mathsf{G}(A^2_{new} \Leftrightarrow \mathsf{F}A^1_{new})$.

Starting from some φ, this repetitive construction eventually halts (when no ψ can be found), the resulting formula φ' has temporal height at most 2, uses flat until, and is satisfiable iff φ is. It can be computed in logspace, so that

PROPOSITION 7.1. *For any set* $\mathsf{H}_1, \ldots$ *of PLTL temporal combinators,* $SAT(\mathsf{L}(\mathsf{H}_1, \ldots)) \leq_L SAT(\mathsf{L}^2_\omega(\mathsf{F}, \mathsf{H}_1, \ldots))$.

COROLLARY 7.3. $SAT(\mathsf{L}^2_\omega(\mathsf{F}, \mathsf{X}))$, $SAT(\mathsf{L}^2_\omega(\mathsf{U}))$ *and* $SAT(\mathsf{L}^2_\omega(\mathsf{U}^-))$ *are* **PSPACE**-*hard.*

7.3. Satisfiability without temporal nesting

We now consider formulae in $\mathsf{L}^1_\omega(\mathsf{U}, \mathsf{X})$, i.e. without nesting of temporal operators. The main result is

PROPOSITION 7.2. *Assume* $\varphi \in \mathsf{L}^1_\omega(\mathsf{U}, \mathsf{X})$. *If* φ *is satisfiable then it is satisfiable in a model* $S' = s_0, s_1, \ldots$ *such that for any* $i, j \geq |\varphi|$, $\varepsilon(s_i) = \varepsilon(s_j)$.

Such a S' can be guessed and checked in polynomial time, hence

COROLLARY 7.4. *For any set* $\mathsf{H}_1, \ldots$ *of PLTL temporal combinators,* $SAT(\mathsf{L}^1_\omega(\mathsf{H}_1, \ldots))$ *is in* **NP**, *and hence is* **NP**-*complete.*

We now proceed with the proof of Prop. 7.2. Our main tool is a notion of *extracted structure*:

DEFINITION 7.1. An *extraction pattern* is an infinite sequence $n_0 < n_1 < n_2 < \ldots$ of increasing natural numbers. Given an extraction pattern $(n_i)_{i \in \mathbb{N}}$ and a structure S, the *extraction from* S *along* $(n_i)_{i \in \mathbb{N}}$ is the structure $s'_0, s'_1, \ldots$ where, for $i = 0, 1, 2, \ldots$, s'_i is a copy of s_{n_i}.

Now consider a formula $\varphi \in \mathsf{L}^1_\omega(\mathsf{U},\mathsf{X})$. Since φ has temporal height 1, it is a boolean combination of atomic propositions and of *temporal* subformulae of the form $\mathsf{X}\psi$ or $\psi\mathsf{U}\psi'$ where ψ and ψ' have temporal height 0. For example, with φ given as

$$\varphi \stackrel{\text{def}}{=} ((A \vee \neg C)\mathsf{U}B) \wedge \neg(A\mathsf{U}B) \wedge (\neg\mathsf{X}A \vee \neg(A\mathsf{U}C)) \wedge \neg B$$

the temporal subformulae of φ are $(A \vee \neg C)\mathsf{U}B$, $A\mathsf{U}B$, $\mathsf{X}A$ and $A\mathsf{U}C$.

DEFINITION 7.2. From any $S = s_0, s_1, \ldots$, and given $\varphi \in \mathsf{L}^1_\omega(\mathsf{U},\mathsf{X})$, we extract a set of positions, called *the witnesses for φ in S*. The rules are that 0 is always a witness, and that each temporal subformula of φ may require one witness:

1. for a temporal subformula $\mathsf{X}\psi$, 1 is the witness,
2. for a temporal subformula $\psi\mathsf{U}\psi'$, we have three cases

(i) if $S \models \psi\mathsf{U}\psi'$ and i is the smallest position such that $S, i \models \psi'$, then i is the witness. (Observe that for all $j < i$, $S, j \models \psi \wedge \neg\psi'$.)

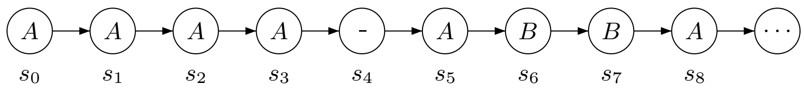
(ii) if $S \not\models \mathsf{F}\psi'$, then no witness is needed.

(iii) otherwise $S \not\models \psi\mathsf{U}\psi'$ and $S \models \mathsf{F}\psi'$. Let i be the smallest position such that $S, i \not\models \psi$, then i is the witness. (Observe that $S, i \not\models \psi'$ and for all $j < i$, $S, j \models \psi \wedge \neg\psi'$.)

Clearly, if $\{n_0, n_1, \ldots, n_k\}$ are the witnesses for φ, then $k < |\varphi|$.

We continue our earlier example: let S be the structure

S:



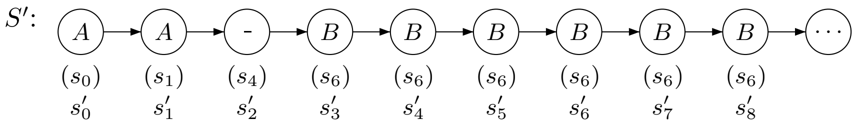
where C never holds. Here $S \models \varphi$. Indeed, $S \models (A \vee \neg C)\mathsf{U}B$, $S \not\models A\mathsf{U}B$, $S \models \mathsf{X}A$ and $S \not\models A\mathsf{U}C$. The witness for $\mathsf{X}A$ is 1. The witness for $(A \vee \neg C)\mathsf{U}B$ is 6 since we are in case (a) from definition 7.2, and s_6 is the first position where B holds. No witness is needed for $A\mathsf{U}C$ since we are in case (b). The witness for $A\mathsf{U}B$ is 4 since we are in case (c) and s_4 is the first position where A does not hold. Finally, the witnesses for φ are $\{0, 1, 4, 6\}$.

LEMMA 7.1. *Let $\varphi \in \mathsf{L}^1_\omega(\mathsf{U},\mathsf{X})$ and S be a structure. Let $(n_i)_{i\in\mathbb{N}}$ be an extraction pattern containing all witnesses for φ in S. Let S' be the structure extracted from φ along $(n_i)_{i\in\mathbb{N}}$. Then for any subformula ψ of φ, $S \models \psi$ iff $S' \models \psi$.*

Proof. By induction on the structure of ψ. Since all other cases are obvious, we only need deal with the case $\psi_1 \mathsf{U} \psi_2$ and show that $S' \models \psi_1 \mathsf{U} \psi_2$ iff $S \models \psi_1 \mathsf{U} \psi_2$. Assume $S \models \psi_1 \mathsf{U} \psi_2$. Let i be the witness for $\psi_1 \mathsf{U} \psi_2$. So $S, i \models \psi_2$ and, for any $j < i$, $S, j \models \psi_1$. (A copy of) s_i appears in S' as some s'_n and all $s'_{n'}$ for $n' < n$ are (copies of) s_j's for $j < i$, hence $S' \models \psi_1 \mathsf{U} \psi_2$ (remember that ψ_1 and ψ_2 have no temporal operator.)
Now assume $S \not\models \psi_1 \mathsf{U} \psi_2$. If $\psi_1 \mathsf{U} \psi_2$ has no witness, then no s_i satisfies ψ_2 and therefore no s'_n: then $S' \not\models \psi_1 \mathsf{U} \psi_2$. If i is the witness for $\psi_1 \mathsf{U} \psi_2$, then $S, i \not\models \psi_1$ and $S, j \not\models \psi_2$ for $j \leq i$. Assume s_i appears as s'_n in S': we have $S', n \not\models \psi_1$ and $S', m \not\models \psi_2$ for $m \leq n$, so that $S', n \not\models \psi_1 \mathsf{U} \psi_2$. ▮

We may now conclude the proof of Prop. 7.2: Consider now a satisfiable $\varphi \in \mathtt{L}^1_\omega(\mathsf{U}, \mathsf{X})$ and assume $S \models \varphi$. Let $\{n_0, \ldots, n_k\}$ be the witnesses for φ in S. We turn these into an extraction pattern by considering the sequence $n_0 < n_1 < \cdots < n_k$ prolongated by some $n_{k+1} < n_{k+2} < \ldots$ where the n_{k+i} are positions of states carrying the same valuation (there must be at least one valuation appearing infinitely often). The extracted S' has the form required for Prop. 7.2.

Continuing our previous example, and assuming the valuation of s_6 appears infinitely often, the resulting S' is made out of s_0, s_1, s_4 and s_6, and it satisfies φ:



7.4. Model checking without temporal nesting

We now consider model checking of formulae where the temporal height is at most 1.

PROPOSITION 7.3. *$MC(\mathtt{L}^1_\omega(\mathsf{U}, \mathsf{X}))$ is in* **NP**.

Proof. Consider $\varphi \in \mathtt{L}^1_\omega(\mathsf{U}, \mathsf{X})$ and assume $T, s \models \varphi$. Then there is a path S in T starting from s such that $S, s \models \varphi$.
The witnesses for φ in S are some $W = \{n_0, \ldots, n_k\}$. We consider an extraction pattern containing all witnesses of W and such that the extracted S' be a path in T: this may imply to retain some positions from S, between a $n_i \in W$ and the following n_{i+1}, to ensure connectivity in T. In any case, it is possible to find an extraction pattern where n_k appears as some position $l \leq k \times |T|$.

Therefore, if $T, s \models \varphi$ then this can be seen along a path S of the form $s_0 \dots s_l(s_{l+1} \dots s_{l+m})^\omega$ with $l \leq |\varphi| \times |T|$ and $m \leq |T|$. Guessing this path and checking it can be done in non deterministic polynomial-time. ∎

Since $MC(\mathtt{L}^1_\omega(\mathsf{F}))$ is **NP**-hard [SC85], we get:

COROLLARY 7.5. *For any set* $\mathsf{H}_1, \dots$ *of PLTL temporal combinators,* $MC(\mathtt{L}^1_\omega(\mathsf{F}, \mathsf{H}_1, \dots))$ *is* **NP**-*complete.*

8. CONCLUDING REMARKS

In this article we have measured the complexity of model checking and satisfiability for all fragments of PLTL obtained by bounding (1) the number of atomic propositions, (2) the temporal height, and (3) restricting the temporal operators one allows. Table 1 provides a complete summary.

In this table we use $\mathsf{U}^?$ to denote any of U and U^- since one outcome of our study is that all the problems we considered have the same computational complexity when “Until” is replaced by the weaker “flat Until”, thereby ruining some hopes of [Dam99].

Some general conclusions can be read in the table. In most cases no reduction in complexity occurs when two propositions are allowed, or with temporal height two. Moreover, in most cases, for equal fragments, satisfiability and model checking belong to the same complexity class. Still the table displays some exceptions, two of which deserve comments :

1. Model checking and satisfiability for $\mathtt{L}^\omega_1(\mathsf{U})$ (only one proposition) are in **P**. Admittedly this fragment is not very relevant when it comes to, say, protocol verification. Moreover, it is open whether those problems are **P**-hard or in **NL**, to quote a few possibilities.

2. Model checking for $\mathtt{L}^k_\omega(\mathsf{F}, \mathsf{X})$ is only **NP**-complete. This shows that F+X can be simpler than U. Because **NP**-hardness is already intractable, this result does not immediately suggest improved deterministic algorithms. However, the isolated fragment is very relevant.

Another way to see our results is to focus on the general techniques that we developed: we provided a simple transformation from QBF into model checking problems, and we formalized a number of logspace transformations leading to a few basic rules of thumb:

(1) when the number of propositions is fixed, satisfiability can be transformed into model checking,

TABLE 1.

A complete summary of complexity measures

	$n-1, k<\omega$	Model checking	Satisfiability
$\mathtt{L}(\ldots)$	$\mathtt{L}_n^0(\ldots)$	L	L
	$\mathtt{L}_\omega^0(\ldots)$	L [Lyn77]	NP-complete [Coo71]
$\mathtt{L}(\mathsf{F})$	$\mathtt{L}(\mathsf{F})$	NP-complete [SC85]	NP-complete [ON80]
	$\mathtt{L}_\omega^1(\mathsf{F})$	NP-complete	NP-complete
	$\mathtt{L}_2^\omega(\mathsf{F})$	NP-complete	NP-complete
	$\mathtt{L}_1^\omega(\mathsf{F})$	in P, NL-hard	P
	$\mathtt{L}_n^{k+1}(\mathsf{F})$	NL-complete	L
$\mathtt{L}(\mathsf{U}^?)$	$\mathtt{L}(\mathsf{U}^?)$	PSPACE-complete [SC85]	PSPACE-complete [SC85, HR83]
	$\mathtt{L}_\omega^2(\mathsf{U}^?)$	PSPACE-complete	PSPACE-complete
	$\mathtt{L}_\omega^1(\mathsf{U}^?)$	NP-complete	NP-complete
	$\mathtt{L}_2^\omega(\mathsf{U}^?)$	PSPACE-complete	PSPACE-complete
	$\mathtt{L}_1^\omega(\mathsf{U}^?)$	in P, NL-hard	P
	$\mathtt{L}_n^{1+k}(\mathsf{U}^?)$	NL-complete	L
$\mathtt{L}(\mathsf{X})$	$\mathtt{L}(\mathsf{X})$	NP-complete	NP-complete
	$\mathtt{L}_\omega^k(\mathsf{X})$	L	NP-complete
	$\mathtt{L}_1^\omega(\mathsf{X})$	NP-complete	NP-complete
	$\mathtt{L}_n^k(\mathsf{X})$	L	L
$\mathtt{L}(\mathsf{F},\mathsf{X})$	$\mathtt{L}(\mathsf{F},\mathsf{X})$	PSPACE-complete [SC85]	PSPACE-complete [SC85, HR83]
	$\mathtt{L}_\omega^{2+k}(\mathsf{F},\mathsf{X})$	NP-complete	PSPACE-complete [Har85, Spa93]
	$\mathtt{L}_\omega^1(\mathsf{F},\mathsf{X})$	NP-complete	NP-complete
	$\mathtt{L}_1^\omega(\mathsf{F},\mathsf{X})$	PSPACE-complete	PSPACE-complete
	$\mathtt{L}_n^{1+k}(\mathsf{F},\mathsf{X})$	NL-complete	L
$\mathtt{L}(\mathsf{U}^?,\mathsf{X})$	$\mathtt{L}(\mathsf{U}^?,\mathsf{X})$	PSPACE-complete [SC85]	PSPACE-complete [SC85, HR83]
	$\mathtt{L}_\omega^2(\mathsf{U}^?,\mathsf{X})$	PSPACE-complete	PSPACE-complete [Har85, Spa93]
	$\mathtt{L}_\omega^1(\mathsf{U}^?,\mathsf{X})$	NP-complete	NP-complete
	$\mathtt{L}_1^\omega(\mathsf{U}^?,\mathsf{X})$	PSPACE-complete	PSPACE-complete
	$\mathtt{L}_n^{1+k}(\mathsf{U}^?,\mathsf{X})$	NL-complete	L

(2) n propositional variables can be encoded into

(2.1) only one if F (sometimes) and X (next) are allowed,

(2.2) only two if U (until) is allowed,

(3) when arbitrarily many propositions are allowed, temporal height can be reduced to 2 if F is allowed, and

(4) model checking for logics with X can be transformed into model checking without X.

(5) Besides, when the formula φ has temporal height at most 1, knowing whether $S \models \varphi$ only depends on $\mathcal{O}(|\varphi|)$ places in S.

Most of the time, these techniques are used to strengthen earlier hardness results, showing that they also apply to specific fragments. In some cases we develop specific arguments showing that the complexity really decreases under the identified threshold values.

The general situation in our study is that lower bounds are preserved when fragments are taken into account. Hence our investigations do not give a formal justification of the alleged simplicity of "simple practical applications". Rather, we show that several natural suggestions are not sufficient.

Understanding and taming the complexity of linear temporal logics remains an important issue and the present work can be seen as some additional contribution. The ground is open for further investigations. We think future work could investigate

- different, finer definitions of fragments (witness [EES90]) that can be inspired by practical examples, or that aim at defeating one of our hardness proofs, e.g. forbidding the renaming technique we use in sections 7.1 and 7.2,
- restrictions on the models rather than the formulae,
- other complexity measures: e.g. average complexity, or separated complexity measure for models and formulae, or analysis of hard and easy distributions.

Additionaly, it must be noted that we only considered satisfiability and model checking, and ignored other problems that are important for verification: module checking, semantic entailment, ...

REFERENCES

BBC+96. N. Bjørner, A. Browne, E. Chang, M. Colón, A. Kapur, Z. Manna, H. B. Sipma, and T. E. Uribe. STeP: deductive-algorithmic verification of reactive and real-time systems. In *Proc. 8th Int. Conf. Computer Aided Verification (CAV'96), New Brunswick, NJ, USA, July-Aug. 1996*, volume 1102 of *Lecture Notes in Computer Science*, pages 415–418. Springer, 1996.

BCG88. M. C. Browne, E. M. Clarke, and O. Grumberg. Characterizing finite Kripke structures in propositional temporal logic. *Theoretical Computer Science*, 59(1–2):115–131, 1988.

BK98. D. Basin and N. Klarlund. Automata based symbolic reasoning in hardware verification. *Formal Methods in System Design*, 13(3):253–288, 1998.

Bul65. R. A. Bull. An algebraic study of Diodorean modal systems. *The Journal of Symbolic Logic*, 30(1):58–64, 1965.

CC00. H. Comon and V. Cortier. Flatness is not a weakness. In *Proc. 14th Int. Workshop Computer Science Logic (CSL'2000), Fischbachau, Germany, Aug. 2000*, volume 1862 of *Lecture Notes in Computer Science*, pages 262–276. Springer, 2000.

CGH97. E. M. Clarke, O. Grumberg, and K. Hamaguchi. Another look at LTL model checking. *Formal Methods in System Design*, 10(1):47–71, 1997.

CL93. Cheng-Chia Chen and I-Peng Lin. The computational complexity of satisfiability of temporal Horn formulas in propositional linear-time temporal logic. *Information Processing Letters*, 45(3):131–136, 1993.

Coo71. S. Cook. The complexity of theorem-proving procedures. In *Proc. 3rd ACM Symp. Theory of Computing (STOC'71), Shaker Heights, Ohio, USA, May 1971*, pages 151–158, 1971.

Dal96. M. Dalal. An almost quadratic class of satisfiability problems. In *Proc. 12th European Conf. Artificial Intelligence (ECAI'96), Budapest, Hungary, Aug. 1996*, pages 355–359. John Wiley & Sons, 1996.

Dam99. D. R. Dams. Flat fragments of CTL and CTL*: Separating the expressive and distinguishing powers. *Logic Journal of the IGPL*, 7(1):55–78, 1999.

Dem96. S. Demri. A simple tableau system for the logic of elsewhere. In *Proc. 5th Int. Workshop on Theorem Proving with Analytical Tableaux and Related Methods (TABLEAUX'96), Terrasini, Palermo, Italy, May 1996*, volume 1071 of *Lecture Notes in Artificial Intelligence*, pages 177–192. Springer, 1996.

DFR00. C. Dixon, M. Fisher, and M. Reynolds. Execution and proof in a Horn-clause temporal logic. In H. Barringer, M. Fisher, D. Gabbay, and G. Gough, editors, *Advances in Temporal Logic*, volume 16 of *Applied Logic Series*, pages 413–433. Kluwer Academic, 2000.

DG99. V. Diekert and P. Gastin. An expressively complete temporal logic without past tense operators for Mazurkiewicz traces. In *Proc. 13th Int. Workshop Computer Science Logic (CSL'99), Madrid, Spain, Sep. 1999*, volume 1683 of *Lecture Notes in Computer Science*, pages 188–203. Springer, 1999.

DLNN97. F. M. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. *Information and Computation*, 134(1):1–58, 1997.

DS98. S. Demri and Ph. Schnoebelen. The complexity of propositional linear temporal logics in simple cases (extended abstract). In *Proc. 15th Ann. Symp. Theoretical Aspects of Computer Science (STACS'98), Paris, France, Feb. 1998*, volume 1373 of *Lecture Notes in Computer Science*, pages 61–72. Springer, 1998.

EES90. E. A. Emerson, M. Evangelist, and J. Srinivasan. On the limits of efficient temporal decidability. In *Proc. 5th IEEE Symp. Logic in Computer Science (LICS'90), Philadelphia, PA, USA, June 1990*, pages 464–475, 1990.

EL87. E. A. Emerson and Chin-Laung Lei. Modalities for model checking: Branching time logic strikes back. *Science of Computer Programming*, 8(3):275–306, 1987.

Eme90. E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, vol. B*, chapter 16, pages 995–1072. Elsevier Science, 1990.

FV98. J. Franco and A. Van Gelder. A perspective on certain polynomial time solvable classes of Satisfiability. *Discrete Applied Mathematics*, 2002. In press.

Gor94. R. Goré. Cut-free sequent and tableau systems for propositional Diodorean modal logics. *Studia Logica*, 53:433–457, 1994.

Hal95. J. Y. Halpern. The effect of bounding the number of primitive propositions and the depth of nesting on the complexity of modal logic. *Artificial Intelligence*, 75(2):361–372, 1995.

Har85. D. Harel. Recurring dominos: Making the highly undecidable highly understandable. *Annals of Discrete Mathematics*, 24:51–72, 1985.

Hem00. E. Hemaspaandra. The complexity of Poor Man's logic. In *Proc. 17th Ann. Symp. Theoretical Aspects of Computer Science (STACS'2000), Lille, France, Feb. 2000*, volume 1770 of *Lecture Notes in Computer Science*, pages 230–241. Springer, 2000.

Hol97. G. J. Holzmann. The model checker Spin. *IEEE Transactions on Software Engineering*, 23(5):279–295, 1997.

HR83. J. Y. Halpern and J. H. Reif. The propositional dynamic logic of deterministic, well-structured programs. *Theoretical Computer Science*, 27(1–2):127–165, 1983.

Jon75. N. D. Jones. Space-bounded reducibility among combinatorial problems. *Journal of Computer and System Sciences*, 11(1):68–85, 1975.

KV98. O. Kupferman and M. Y. Vardi. Relating linear and branching model checking. In *Proc. IFIP Working Conference on Programming Concepts and Methods (PROCOMET'98), Shelter Island, NY, USA, June 1998*, pages 304–326. Chapman & Hall, 1998.

Lam83. L. Lamport. What good is temporal logic? In *Information Processing'83. Proc. IFIP 9th World Computer Congress, Sep. 1983, Paris, France*, pages 657–668. North-Holland, 1983.

LMS01. F. Laroussinie, N. Markey, and Ph. Schnoebelen. Model checking CTL^+ and $FCTL$ is hard. In *Proc. 4th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS'2001), Genova, Italy, Apr. 2001*, volume 2030 of *Lecture Notes in Computer Science*. Springer, 2001.

LP85. O. Lichtenstein and A. Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *Proc. 12th ACM Symp. Principles of Programming Languages (POPL'85), New Orleans, LA, USA, Jan. 1985*, pages 97–107, 1985.

LS00. F. Laroussinie and Ph. Schnoebelen. Specification in CTL+Past for verification in CTL. *Information and Computation*, 156(1/2):236–263, 2000.

Lyn77. N. Lynch. Log space recognition and translation of parenthesis languages. *Journal of the ACM*, 24(4):583–590, 1977.

MC85. B. Mishra and E. M. Clarke. Hierarchical verification of asynchronous circuits using temporal logic. *Theoretical Computer Science*, 38(2–3):269–291, 1985.

MP92. Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer, 1992.

ON80. H. Ono and A. Nakamura. On the size of refutation Kripke models for some linear modal and tense logics. *Studia Logica*, 39(4):325–333, 1980.

Pap94. C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

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

Seg71. K. Segerberg. An essay in classical modal logic (three vols.). Technical Report Filosofiska Studier nr 13, Uppsala Universitet, 1971.

Spa93. E. Spaan. *Complexity of Modal Logics*. PhD thesis, ILLC, Amsterdam University, NL, March 1993.

SR99. P.-Y. Schobbens and J.-F. Raskin. The logic of "initially" and "next": Complete axiomatization and complexity. *Information Processing Letters*, 69(5):221–225, 1999.

Var94. M. Y. Vardi. Nontraditional applications of automata theory. In *Proc. Int. Symp. Theoretical Aspects of Computer Software (TACS'94), Sendai, Japan, Apr. 1994*, volume 789 of *Lecture Notes in Computer Science*, pages 575–597. Springer, 1994.

Wol83. P. Wolper. Temporal logic can be more expressive. *Information and Control*, 56(1/2):72–99, 1983.

WVS83. P. Wolper, M. Y. Vardi, and A. P. Sistla. Reasoning about infinite computation paths (extended abstract). In *Proc. 24th IEEE Symp. Foundations of Computer Science (FOCS'83), Tucson, AZ, USA, Nov. 1983*, pages 185–194, 1983.