Electronic Notes in Theoretical Computer Science 68 No. 2 (2002)
URL: `http://www.elsevier.nl/locate/entcs/volume68.html` 18 pages

Past is for Free: on the Complexity of Verifying Linear Temporal Properties with Past

Nicolas Markey [1]

Laboratoire d'Informatique Fondamentale d'Orléans
Université Orléans & CNRS FRE 2490
6, rue Léonard de Vinci – BP 6759
45067 ORLEANS Cédex 2 – FRANCE

Abstract

We study the complexity of satisfiability and model-checking of the linear-time temporal logic with past (PLTL). More precisely, we consider several fragments of PLTL, depending on the allowed set of temporal modalities, the use of negations or the nesting of future formulae into past formulae. Our results show that "past is for free", that is it does not bring additional theoretical complexity, even for small fragments, and even when nesting future formulae into past formulae. We also remark that existential and universal model-checking can have different complexity for certain fragments.

Introduction

Temporal logics.

In 1977, Pnueli [20] introduced *temporal logics* for reasoning about concurrent programs. Those logics provide powerful methods for specifying and verifying properties of reactive systems. We refer to [3,5,18,19] for more motivations and background.

The temporal framework most used in research studies is linear-time propositional temporal logic (called LTL). An LTL formula expresses properties about the ordering of events along the runs of a system under study. For instance, that at all times a `request` eventually gives rise to a `grant` can be expressed with:

$$\mathbf{G}\,(\mathtt{request} \Rightarrow \mathbf{F}\,\mathtt{grant}) \tag{S1}$$

Temporal logics with past.

LTL is a *pure-future* temporal logic, *i.e.* a logic where modalities only refer to the future of the current state. It is possible, however, to define *past-time*

[1] Email: `markey@lifo.univ-orleans.fr`

modalities [10,8,16]. For example, for expressing that a $\texttt{grant}$ may only occur if some $\texttt{request}$ *has been* issued, we would write

$$\mathbf{G}\,(\texttt{grant} \Rightarrow \mathbf{F}^{-1}\,\texttt{request}) \tag{S2}$$

It is well-known, since [10], that past-time modalities do not increase expressiveness of LTL. [7] gives a method for translating LTL+Past formulae into equivalent pure-future LTL formulae. For instance, an equivalent pure-future formula for (S2) is

$$\mathbf{G}\,\neg\texttt{grant} \vee (\neg\texttt{grant})\,\mathcal{U}\,\texttt{request} \tag{S3}$$

expressing that either there is no grant at all, or there is no grant until the first request. By concern of minimality, since it does not add expressive power, past has not been widely studied, and model-checkers such as Spin or Cadence-SMV cannot handle LTL+Past specifications. Several methods have been proposed for model-checking LTL+Past [23,11], but as far as we know, these have not been implemented.

The benefits of the past.

Allowing past-time modalities makes specifications easier and more natural (*i.e.* closer to the natural language specifications) [16]. Furthermore, there is a sense in which LTL+Past really brings more expressive power: [14] shows a succinctness gap between LTL and LTL+Past, *i.e.* there exists LTL+Past formulae that only have LTL equivalents of exponential size. Finally, since model-checking and satisfiability are not more difficult for LTL+Past (PSPACE-complete in both cases [22]), one could argue that LTL+Past should be preferred.

These arguments seem to indicate that past is for free. Can this observation be made stronger and more systematic? In this paper, we investigate if this result still holds for different fragments of LTL+Past, in order to characterize fragments that are more expressive but not harder to verify.

Our contribution.

We provide a systematic study of fragments of LTL+Past obtained by three kinds of restrictions: on the set of allowed modalities, on the use of negations, and on nesting of past and future modalities. These results rely on a few basic techniques that are used throughout the paper.

Related work.

Several methods have been proposed for model-checking LTL+Past [23,11], but none have ever been implemented. As regards fragments of LTL+Past, [21] studies LUSAT, the fragment of LTL+Past with only $\mathcal{U}$ and $\mathcal{S}$, and provides an optimal (PSPACE) automata-theoretic algorithm. [4] studies the complexity of several fragments of LTL obtained by limiting the temporal height and the number of atomic propositions. Branching-time temporal logics with past have been investigated in [13,15].

Outline of the paper.

In the sequel, we first formally define the structures, logics and problems under study, and sum up our results. In section 2 we prove the NP-completeness results, and in section 3, the PSPACE-completeness results. We summarize our study and conclude in section 4.

1 PLTL: Linear Temporal Logic with Past

Syntax of PLTL.

Let $AP = \{P_1, P_2, \ldots\}$ be a countable set of atomic propositions. We define the syntax of PLTL as follows:

$$\text{PLTL} \ni \phi, \psi ::= \psi \vee \phi \mid \neg\phi \mid \mathbf{X}\phi \mid \psi\mathcal{U}\phi \mid \mathbf{X}^{-1}\phi \mid \psi\mathcal{S}\phi \mid P_1 \mid P_2 \mid \ldots$$

where $\mathcal{U}$ reads "until", $\mathcal{S}$ reads "since", $\mathbf{X}$ is "next" and $\mathbf{X}^{-1}$, "previous".

Some very useful abbreviations are commonly defined: $\top \equiv P_1 \vee \neg P_1$, $\Rightarrow$, $\Leftrightarrow$, ... As for temporal modalities, we will use the classical $\mathbf{F}$ and $\mathbf{G}$, as well as their past counterparts $\mathbf{F}^{-1}\phi \equiv \top\mathcal{S}\phi$ and $\mathbf{G}^{-1}\phi \equiv \neg\mathbf{F}^{-1}\neg\phi$, read "*eventually in the past*" and "*always in the past*" respectively.

Modalities $\mathcal{S}$, $\mathbf{X}^{-1}$, $\mathbf{F}^{-1}$ and $\mathbf{G}^{-1}$ are called "*past modalities*", while $\mathcal{U}$, $\mathbf{X}$, $\mathbf{F}$, $\mathbf{G}$ are "*future modalities*".

Semantics.

Formulas of PLTL are interpreted over paths. A path is a pair (π, ξ) in which π is an infinite sequence of states $\pi(0)$, $\pi(1)$, ... and ξ is a mapping from $\{\pi(0), \pi(1), \ldots, \pi(n), \ldots\} \to 2^{AP}$. This way, the states of π are labeled with atomic propositions.

Given a path (π, ξ), a natural i and a formula ϕ, we inductively define the relation $\pi, i \models \phi$ (read "*ϕ holds at position i along π*") as follows:

$\pi, i \models P$	if, and only if, $P \in \xi(\pi(i))$,
$\pi, i \models \phi \wedge \psi$	if, and only if, $\pi, i \models \phi$ and $\pi, i \models \psi$,
$\pi, i \models \neg\phi$	if, and only if, $\pi, i \not\models \phi$,
$\pi, i \models \mathbf{X}\phi$	if, and only if, $\pi, i+1 \models \phi$,
$\pi, i \models \psi\mathcal{U}\phi$	if, and only if, there exists some $j \geq i$ s.t. $\pi, j \models \phi$ and for all $i \leq k < j$, $\pi, k \models \psi$,
$\pi, i \models \mathbf{X}^{-1}\phi$	if, and only if, $i > 0$ and $\pi, i-1 \models \phi$,
$\pi, i \models \psi\mathcal{S}\phi$	if, and only if, there exists some $j \leq i$ s.t. $\pi, j \models \phi$ and for all $j < k \leq i$, $\pi, k \models \psi$.

Two formulas are *(globally) equivalent over a class* Π *of paths* (which we denote $\phi \equiv^{\Pi} \psi$) if for any path $\pi \in \Pi$ and any integer i, the equivalence $\pi, i \models$

$\phi \Leftrightarrow \pi, i \models \psi$ holds. The formulas are *initially equivalent* over Π ($\phi \equiv_i^\Pi \psi$) if for all paths $\pi \in \Pi$, $\pi, 0 \models \phi \Leftrightarrow \pi, 0 \models \psi$ is true. Whenever Π is not given, the equivalence has to hold along any possible path, namely $(2^{AP})^{\mathbb{N}}$.

Obviously, two equivalent formulas are initially equivalent. The converse does not hold. For instance, $P_1 \mathcal{S} P_2$ and P_2 are initially equivalent, but they clearly are not globally equivalent.

A formula ϕ is said to be *initially* (resp. *globally*) *valid* over Π if it is initially (resp. globally) equivalent to $\top$ over Π. It is *initially* (resp. *globally*) *satisfiable* over Π if its negation is not initially (resp. globally) valid over Π. This means that there exists a path $\pi \in \Pi$ (resp. and a position i along that path) such that $\pi, 0 \models \phi$ (resp. $\pi, i \models \phi$).

These definitions formalize the results we mentioned about expressive power in the introduction: that PLTL is as expressive as LTL [10,8,7] means that for any PLTL formula, there exists an initially equivalent LTL formula. The succinctness gap from [14] can be expressed as follows: there exists a sequence of PLTL formulas (ϕ_n), s.t. $|\phi_n| \in O(n)$, and for which any sequence of equivalent LTL formulas (ψ_n) verifies that $|\psi_n| \in \Omega(2^n)$.

Verification problems.

In this paper, we are concerned with the following questions:

- initial satisfiability, as defined above. Note that, generally speaking, this problem and global satisfiability are inter-reducible: a formula ϕ is globally satisfiable if, and only if, $\mathbf{F}\phi$ is initially satisfiable, and conversely, ϕ is initially satisfiable if, and only if, $\mathbf{G}^{-1}\mathbf{F}^{-1}\phi$ is globally satisfiable;
- universal model-checking, which is initial validity over a given set Π of paths;
- existential model-checking, which is initial satisfiability over a given set Π of paths.

For model-checking, the set Π is often defined through a Kripke Structure (KS for short), that is, a 4-tuple $K = (Q, Q_0, l, R)$ in which Q is a finite set of states, Q_0 is the set of initial states, $l \in (2^{AP})^Q$ indicates the propositions that are true in each state of Q, and $R \subseteq Q \times Q$ is a total relation representing the set of allowed transitions. The *size* of K, which we denote $|K|$, is $|Q| + |R|$. A KS generates a set Π of paths in the obvious way.

It should be remarked right now that, in the general case, the two model-checking problems are dual: indeed, there exists a path satisfying a formula ϕ if, and only if, it is not the case that every path satisfies the negation of ϕ. But this equivalence only holds for logics (or fragments of logics) allowing negation.

Fragments of PLTL.

We consider three types of restrictions: first of all, restrictions about the allowed modalities. For denoting the fragment of LTL where only $\mathbf{M}_1, \ldots, \mathbf{M}_p$ are allowed, we use the classical notation $\mathbf{L}(\mathbf{M}_1, \ldots, \mathbf{M}_p)$. For instance, $\mathbf{L}(\mathbf{F})$ is the logic where $\mathbf{F}$ is the only allowed temporal modality [2]. The second restriction we deal with affects negations: we write $\mathbf{L}^+(\mathbf{F},\mathbf{X})$, for example, for the logic where the only modalities are $\mathbf{F}$ and $\mathbf{X}$, and where modalities can not occur in the scope of a negation. Last, a formula is said to be *stratified* if it has no future modality in the scope of past modalities [17]. Sets of stratified formulas are denoted by $\mathbf{L}_s(\ldots)$. We write $\mathbf{L}_s^+(\mathbf{F},\mathcal{S})$ when combining restrictions about negation and stratification.

For example, $\mathbf{F}(a \wedge \mathbf{G}^{-1}(b \vee \mathbf{F}c))$ lies in $\mathbf{L}^+(\mathbf{F},\mathbf{G}^{-1})$, but it is not stratified. It is initially equivalent to $(b \vee \mathbf{F}c)\mathcal{U}a$, which is in $\mathbf{L}^+(\mathcal{U})$. And it is globally equivalent to $\mathbf{F}(a \wedge (\mathbf{F}c \vee \mathbf{G}^{-1}b \vee b\mathcal{S}c))$, which belongs to $\mathbf{L}_s(\mathbf{F},\mathcal{S})$.

Our results.

We get in the sequel the following results:

	Exist. model-ch.	Univ. model-ch.	Satisf.
$\mathbf{L}^+(\mathbf{F})$,$\mathbf{L}^+(\mathbf{G})$,$\mathbf{L}^+(\mathbf{X})$	**NP-c.**	**coNP-c.**	**NP-c.**
$\mathbf{L}^+(\mathbf{F},\mathbf{X})$	NP-c. [22]	**PSPACE-c.**	NP-c. [22]
$\mathbf{L}^+(\mathbf{G},\mathbf{X})$	**PSPACE-c.**	(coNP-c.)	**PSPACE-c.**
$\mathbf{L}^+(\mathcal{U})$	**PSPACE-c.**	**PSPACE-c.**	**PSPACE-c.**
$\mathbf{L}(\mathbf{X},\mathbf{X}^{-1},\mathcal{S})$	**NP-c.**	(coNP-c.)	**NP-c.**
$\mathbf{L}_s^+(\mathbf{F},\mathbf{X}^{-1})$	(NP-c.)	(PSPACE-c.)	(NP-c.)
$\mathbf{L}_s^+(\mathbf{G},\mathbf{X}^{-1})$	**PSPACE-c.**	(coNP-c.)	**PSPACE-c.**
$\mathbf{L}^+(\mathbf{F},\mathbf{X},\mathbf{F}^{-1},\mathbf{X}^{-1})$	**NP-c.**	(PSPACE-c.)	**NP-c.**
$\mathbf{L}^+(\mathbf{G},\mathbf{X},\mathbf{G}^{-1},\mathbf{X}^{-1})$	(PSPACE-c.)	(coNP-c.)	(PSPACE-c.)
$\mathbf{L}(\mathbf{F},\mathbf{F}^{-1})$	**NP-c.**	(coNP-c.)	**NP-c.** [6]
$\mathbf{L}_s^+(\mathbf{F},\mathcal{S})$	**PSPACE-c.**	**PSPACE-c.**	**PSPACE-c.**
$\mathbf{L}_s^+(\mathbf{G},\mathcal{S})$	**PSPACE-c.**	**PSPACE-c.**	(NP-c.)
$\mathbf{L}^+(\mathbf{G},\mathcal{S})$	(PSPACE-c.)	(PSPACE-c.)	**NP-c.**
PLTL	PSPACE-c. [22]	PSPACE-c. [22]	PSPACE-c. [22]

The results in bold are proved in this paper, the ones in parentheses are corollaries. For instance, existential model-checking for $\mathbf{L}_s^+(\mathbf{F},\mathbf{X}^{-1})$ is NP-complete since it is a subcase of existential model-checking for $\mathbf{L}^+(\mathbf{F},\mathbf{F}^{-1},\mathbf{X},\mathbf{X}^{-1})$, and since it is more general than existential model-checking for $\mathbf{L}^+(\mathbf{F})$. This entails, by duality, that universal model-checking of $\mathbf{L}_s^+(\mathbf{G},\mathbf{X}^{-1})$ is coNP-complete.

[2] In this case, we see $\mathbf{F}$ as a modality, and not as an abbreviation of $\top\mathcal{U}\cdot$.

Such techniques can be used to find the complexity of all the fragments we defined (not all of them are listed in the previous table).

2 NP-complete problems

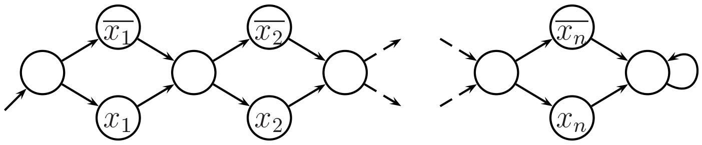
2.1 NP*-hardness of verifying linear temporal properties*

All the fragments we consider have at least NP-hard satisfiability problems since temporal logic encompasses boolean logic. Model-checking is somewhat different: boolean formulas can be evaluated in polynomial time in a given KS. Here we show that (existential) model-checking is NP-hard for all non-trivial fragments of PLTL.

Theorem 2.1 *Existential model-checking is* NP*-hard for* $\mathbf{L}^+(\mathbf{F})$*,* $\mathbf{L}^+(\mathbf{G})$ *and* $\mathbf{L}^+(\mathbf{X})$.

Proof

- We adapt the proof in [22] for NP-hardness of model-checking $\mathbf{L}(\mathbf{F})$. Intuitively, the satisfiability of a 3-SAT-instance $\bigwedge_i \bigvee_j \alpha_{i,j}$, where the $\alpha_{i,j}$ are literals on $\{x_1, x_2, \ldots, x_n\}$, is equivalent to the existence of a path verifying $\bigwedge_i \bigvee_j \mathbf{F}\alpha_{i,j}$ in the following structure:



- in the same way, the satisfiability of $\bigwedge_i \bigvee_j \alpha_{i,j}$ is equivalent to the existence of a path s.t. $\bigwedge_i \bigvee_j \mathbf{G}\neg(\overline{\alpha_{i,j}})$ in the same structure.
- for $\mathbf{L}^+(\mathbf{X})$, the idea is that a 3-SAT-instance $\bigwedge_i \bigvee_j \alpha_{i,j}$ is satisfiable if, and only if, the above structure contains a path verifying $\bigwedge_i \bigvee_j \mathbf{X}^{2n(\alpha_{i,j})-1}\alpha_{i,j}$, where $n(x_k) = n(\overline{x_k}) = k$. See [4] for more details. $\square$

By duality, we get

Theorem 2.2 *Universal model-checking for* $\mathbf{L}^+(\mathbf{G})$*,* $\mathbf{L}^+(\mathbf{F})$ *and* $\mathbf{L}^+(\mathbf{X})$ *are co-*NP*-hard.*

2.2 NP*-easy problems*

What we saw in the previous section entails that any non-trivial verification problem concerning linear temporal logics is NP-hard. We know from [22] that this lower bound is optimal for $\mathbf{L}(\mathbf{F})$ and $\mathbf{L}^+(\mathbf{F}, \mathbf{X})$, that is, the satisfiability and (existential) model-checking problems for these logics are NP-complete. In the rest of this section, we prove NP-easiness of satisfiability for four other fragments: $\mathbf{L}(\mathbf{X}, \mathbf{X}^{-1}, \mathcal{S})$, $\mathbf{L}(\mathbf{F}, \mathbf{F}^{-1})$, $\mathbf{L}^{+}(\mathbf{F}, \mathbf{F}^{-1}, \mathbf{X}, \mathbf{X}^{-1})$, and $\mathbf{L}^{+}(\mathbf{G}, \mathcal{S})$. These results carry on to existential model-checking, except for $\mathbf{L}^{+}(\mathbf{G}, \mathcal{S})$, for which we will prove in the next section that model-checking is PSPACE-complete.

We use a systematic method in order to do this:

- we first prove that those fragments have the so-called "polynomial witness property", *i.e.* every satisfiable formula can be satisfied in a path of polynomial "size" (formal definitions given below).
- we show that given a potential witness π, one can check in polynomial time whether $\pi \models \phi$.

This obviously gives an NP-algorithm for satisfiability: first guess the witness, and then check it. In several cases, this technique also provides a proof for NP-easiness of the model-checking problems: Indeed, we show that we can add arbitrary states in the polynomial witness, and this ensures that we can find a polynomial witness in the Kripke structure under study. In these cases, the algorithm for existential model-checking is as follows: first guess the witness, check that it is a run in the Kripke structure, and check that it satisfies the formula.

First of all, we need to introduce more formal definitions: A path (π, ξ) is said to be ultimately periodic if there exist two integers m and p, with $p > 0$, such that for any integer $n \geq m$, $\pi_n = \pi_{n+p}$. Such a path can be *finitely* represented by a *loop*, that is an ultimately-periodic *deterministic* Kripke structure. More precisely, given two integers m and $p > 0$, a loop of type (m, p) is the structure $(Q, \{0\}, l, R)$ such that $Q = [\![0; m+p-1]\!]$ and $R = \{(i, i+1) \mid i \in [\![0; m+p-2]\!]\} \cup \{(m+p-1, m)\}$. A loop is then a finite structure "encoding" an ultimately-periodic path. We write π_L the path associated to a loop L. The size of a loop of type (m, p) is the integer $m+p$. The size of an ultimately periodic path is the size of the smallest loop encoding that path.

2.2.1 Model checking a loop

We first recall the following result:

Theorem 2.3 *Given a pure-future formula ϕ and a loop L, one can check in time $O(|L| \cdot |\phi|)$ whether $\pi_L, 0 \models \phi$.*

Over deterministic KS, the CTL model-checking algorithm can be used for LTL formulas too, since quantification will always refer to the unique execution.

This simple approach does not extend to the problem of checking whether a loop satisfies a PLTL formula, hence a formula "with past time". In fact, in a loop, future is deterministic but past is not, that is any state has one successor but states have two (or even zero) predecessors in the transition relation.

The following lemma gives a way to overcome that problem. First remember that the temporal height (resp. future-temporal height, past-temporal height) of a formula is the maximal number of nested modalities (resp. future modalities, past modalities) in the formula. More generally, we define the temporal height w.r.t. one or several modalities as being the maximum number of times those modalities are nested. We denote it by $\mathrm{h}_{O_1,\ldots,O_n}(\phi)$. In the sequel, $h_P(\phi)$ denotes the past-temporal height of ϕ, *i.e.* $h_{\mathcal{S},\mathbf{X}^{-1}}(\phi)$.

Lemma 2.4 *Let ϕ be a* PLTL*-formula. For any loop L of type (m,p), for all $k \geq m + h_P(\phi)p$,*

$$\pi_L, k \models \phi \textit{ iff } \pi_L, k+p \models \phi.$$

Roughly speaking, this lemma states that after some initial fluctuations, past modalities in ϕ cannot distinguish how many times the loop has been unwound.

Proof The proof is by inuction on the structure of the formula ϕ:

- for $\phi = P$, $\phi = \neg\phi_1$ and $\phi = \phi_1 \vee \phi_2$, the result is obvious;
- if $\phi = \mathbf{X}\phi_1$ or $\phi = \phi_1 \mathcal{U} \phi_2$, we can apply the ind. hyp. to states occurring after the k-th one. This gives the result.
- if $\phi = \mathbf{X}^{-1}\phi_1$, then since $k - 1 \geq m + (h_P(\phi_1) + 1)p - 1 \geq m + h_P(\phi_1)p$, we get from the induction hypothesis the equivalence $\pi_L, k-1 \models \phi_1 \Leftrightarrow \pi_L, k-1+p \models \phi_1$. Thus $\pi_L, k \models \mathbf{X}^{-1}\phi_1 \Leftrightarrow \pi_L, k+p \models \mathbf{X}^{-1}\phi_1$;
- if $\phi = \phi_1 \mathcal{S} \phi_2$, we have $h_P(\phi) = \max(h_P(\phi_1), h_P(\phi_2)) + 1$. Suppose that $\pi_L, k \models \phi$. There exists some $k' \leq k$ s.t. $\pi_L, k' \models \phi_2$, and for $k' < l \leq k$, $\pi_L, l \models \phi_1$. Two cases may arise:
  - · if $k' < k - p$, then we know that the states from π_{k-p} to π_k satisfy ϕ_1. By induction hypothesis, so do the states from π_k to π_{k+p}. Thus, $\pi_L, k+p \models \phi$, since $\pi_L, k' \models \phi_2$ and all the states between $\pi_{k'+1}$ and π_{k+p} satisfy ϕ_1;
  - · otherwise, $k' \geq k - p \geq m + h_P(\phi_1)p$, and the induction hypothesis directly applies to the states between $\pi_{k'}$ and π_k.

  Thus $\pi_L, k \models \phi \Rightarrow \pi_L, k+p \models \phi$. The reverse implication may be proved similarly. □

Corollary 2.5 *Model-checking* PLTL *over loops can be done in polynomial time.*

Proof [Idea] Let ϕ be a formula with past temporal height h, and L a loop of type (m,p). We use dynamic programming in order to fill an array V of boolean values, where $V(n,\psi)$ is true if, and only if, the n-th state satisfies the subformula ψ. Lemma 2.4 enables to deal only with the first $m + (h+1)p$ states of the loop. For all natural n less than $m + (h+1)p$, and for all subformulas ψ of ϕ, we can inductively compute whether $\pi_L, n \models \psi$ (this is an easy extension of the labelling algorithm for CTL). □

2.2.2 Looking for ultimately periodic paths

We recall that an ultimately-periodic witness exists for any satisfiable formula of PLTL.

Theorem 2.6 *A pure-future formula $\phi \in$* LTL *is satisfiable if, and only if, it is satisfiable in a loop. A KS K "existentially" satisfies a formula ϕ if, and only if, it contains an ultimately-periodic path satisfying ϕ. These results also hold for* PLTL *formulas.*

Proof For LTL, the first statement is shown in [22]. The second one can be shown by a classical reduction from model-checking to satisfiability (see [22, lemma 4.3]).

The result is extended to PLTL thanks to the following theorem:

Theorem 2.7 ([10,7]) *For any formula $\phi \in$ PLTL, there exists a boolean combination $\tilde{\phi}$ of pure-future and pure-past formulas, s.t. $\phi \equiv \tilde{\phi}$.* □

2.2.3 NP*-easy fragments*

There simply remains to find the fragments having the "polynomial witness property". We show that for these fragments, we get a polynomial witness by selecting polynomialy many states from an arbitrary witness loop, while keeping the satisfaction of the temporal property.

This first requires some new definitions: Given a path (π, ξ), a *subpath* is a path $(\pi', \xi_{|\pi'})$ where π' is a subsequence of π. We equivalently say that π' is a subpath of π, or that π contains π'. If π' is a subpath of π, there exists an increasing function f such that, for all i, $\pi'_i = \pi_{f(i)}$. We will write $\pi' \sqsubseteq_f \pi$ when we need the function f. Otherwise, we simply write $\pi' \sqsubseteq \pi$.

In the same way, given a loop L, a *subloop* is a loop L' whose associated path is a subpath of the path associated to L. We also write $L' \sqsubseteq_f L$ in that case. Note that a subloop could be bigger than its original loop. For instance, from a loop $(ab)^\omega$ (whose size is 2), we can extract the loop $aaba(aab)^\omega$, with size 7.

Let $L = uv^\omega$ be a loop. We say that a subloop $L' = u'v'^\omega$ of L is *acceptable*, and we write $L' \preccurlyeq_f L$, whenever $f([1, |u'|]) \subseteq [1, |u|]$ and $f([|u'| + 1, |u'v'|]) \subseteq [|u| + 1, |uv|]$. The size of an acceptable subloop is always lower than or equal to the size of the original loop.

The following four technical lemmas directly entail the polynomial witness property for the four PLTL fragments under study. They prove NP-easiness of satisfiability for these fragments. But only the first three give NP-easiness of existential model-checking.

Lemma 2.8 *The truth of an $\mathbf{L}(\mathbf{X}, \mathbf{X}^{-1}, \mathcal{S})$-formula ϕ in the initial state of a path π only depends on the $\mathrm{h}_{\mathbf{X}}(\phi)$ first states of π.*

This result is obvious.

Lemma 2.9 *Let $\phi \in \mathbf{L}(\mathbf{F}, \mathbf{F}^{-1})$, and L be a loop s.t. $\pi_L, i \models \phi$ for some integer i. Then there exists an acceptable subloop $L' \preccurlyeq L$, whose size is polynomial in $|\phi|$, containing π_i, and s.t. any acceptable subloop L'' s.t. $L' \preccurlyeq L'' \preccurlyeq_f L$ satisfies $\pi_{L''}, f^{-1}(i) \models \phi$.*

Proof We suppose that the loop L is of type (m, p), and that, for some i, $\pi_L, i \models \phi$. We write $h = h_P(\phi)$.

For each subformula of ϕ of the form $\mathbf{F}^{-1}\xi$, if there exists a position where ξ is satisfied, then we know from lemma 2.4 that there is one less than $m + hp$. If it exists, we write

$$i_{\mathbf{F}^{-1}\xi} = \min\{i \mid \pi_L, i \models \xi\}$$

The same holds for subformulas of the form $\mathbf{F}\xi$: if there exists a state satisfying ξ, we write

$$i_{\mathbf{F}\xi} = \max\{j \in [\![0; m+hp-1]\!] \mid \pi_L, j \models \xi\}$$

For each $\mathbf{F}$- or $\mathbf{F}^{-1}$-subformula ψ, we define j_ψ to be either equal to i_ψ if $i_\psi \leq m$, or congruent to i_ψ modulo p and between m and $m+p-1$ otherwise. We define L' to be the acceptable subloop of L built by keeping state j_ψ for all $\mathbf{F}$- and $\mathbf{F}^{-1}$-subformulas ψ of ϕ. We also add the current state π_i, and possibly some other states. We let f be the function s.t. $\pi_{L'} \preccurlyeq_f \pi_L$. Remark that L' has type (m', p') with $m' \leq m$ and $p' \leq p$, and that we have $f(m') \leq m \leq f(m'+1)$ and, for all k, $f(m'+kp') \leq m+kp \leq f(m'+kp'+1)$.

The example shown on figure 1 explains this construction.

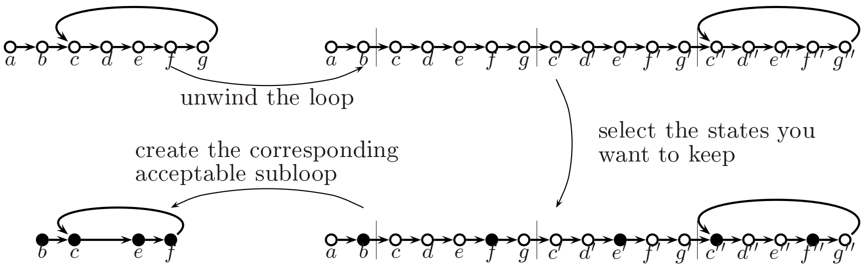


Figure 1. Construction of L'

We now have to prove that this construction is correct. For this, we show that

$$\forall \psi \in \mathrm{SF}(\phi), \forall j \in \mathbb{N},\ \pi_{L'}, j \models \psi \Leftrightarrow \pi_L, f(j) \models \psi$$

where we write $\mathrm{SF}(\phi)$ for the set of subformulas of ϕ and their negation.

We prove this by induction on the structure of ψ:

- for atomic propositions and boolean combinations, the result is straightforward;
- if $\psi = \mathbf{F}\psi_1$, suppose that for some i, $\pi_{L'}, i \models \mathbf{F}\psi_1$. Then for some state $j \geq i$, we have $\pi_{L'}, j \models \psi_1$. By ind. hyp. and since f is increasing, we get a state $f(j) \geq f(i)$ s.t. $\pi_L, f(j) \models \psi_1$, and $\pi_L, f(i) \models \mathbf{F}\psi_1$.

  Conversely, if we suppose that $\pi_L, f(i) \models \mathbf{F}\psi_1$, then there exists a state $j \geq f(i)$ s.t. $\pi_L, j \models \psi_1$. Two cases may arise:
  - either $j \leq m+hp-1$, then we have $j \leq i_{\mathbf{F}\psi_1}$. By ind. hyp. we get that $\pi_{L'}, f^{-1}(i_{\mathbf{F}\psi_1}) \models \psi_1$, and since f is increasing, we have $f^{-1}(i_{\mathbf{F}\psi_1}) \geq i$.
  - either $j \geq m+hp$. In that case, lemma 2.4 ensures that there exists a state k satisfying ψ_1 s.t. k is between $m+(h-1)p$ and $m+hp-1$. Then $k \leq i_{\mathbf{F}\psi_1}$, and $\pi_{L'}, f^{-1}(i_{\mathbf{F}\psi_1}) \models \psi_1$. The remark above ensures that $f^{-1}(i_{\mathbf{F}\psi_1})$ lies between $m'+(h-1)p'$ and $m'+hp'$. Since lemma 2.4 also applies to L', we get that for all $l \geq 0$, $\pi_{L'}, f^{-1}(i_{\mathbf{F}\psi_1}) + lp' \models \psi_1$. Thus for all m, $\pi_{L'}, m \models \mathbf{F}\psi_1$, especially for $m = i$.
- if $\psi = \mathbf{F}^{-1}\psi_1$, the proof is similar. □

Lemma 2.10 *Let $\phi \in \mathbf{L}^+(\mathbf{F}, \mathbf{F}^{-1}, \mathbf{X}, \mathbf{X}^{-1})$ be a satisfiable formula, and L a loop s.t. $\pi_L, i \models \phi$ for some integer i. Then there exists an acceptable subloop L' of*

L, whose size is polynomial in $|\phi|$, containing π_i, and s.t. any loop L'' for which $L' \preccurlyeq L'' \preccurlyeq_f L$ satisfies $\pi_{L''}, f^{-1}(i) \models \phi$.

Proof

The proof is similar to the proof of Lemma 2.9: we first unwind the loop $h_P(\phi)$ times. Then, by induction on the structure of the formula, we build a set S of "witness states". At each step, we prove that

$$\forall j \leq m + h_P(\phi) \cdot p, \forall \psi \in \mathsf{SubF}(\phi), \quad (j, \psi) \in S \Rightarrow \pi_L, j \models \psi \tag{1}$$

- initially, S contains $\{(i, \phi)\}$. The property (1) is satisfied by hypothesis;
- while S contains pairs of the form (j, ψ) where ψ is not (a negation of) an atomic proposition, we remove (j, ψ) from S, put it in T and
  - · if $\psi = \alpha \vee \beta$, then either $\pi_L, j \models \alpha$ or $\pi_L, j \models \beta$. We add (j, α) or (j, β) to S in order to keep (1) true;
  - · if $\psi = \alpha \wedge \beta$, then add (j, α) and (j, β) to S;
  - · if $\psi = \mathbf{X}\alpha$, then add $(j + 1, \alpha)$ (or $(j + 1 - p, \alpha)$ if $j + 1 > m + h_P(\phi) \cdot p$) to S;
  - · if $\psi = \mathbf{X}^{-1}\alpha$, then add $(j - 1, \alpha)$ to S. We know that $j \geq 1$ since $\pi_L, j \models \mathbf{X}^{-1}\alpha$. Thus (1) still holds;
  - · if $\psi = \mathbf{F}\alpha$, we know that α is true in some state k greater than j and smaller than $m+(h(\phi)+1)p$. If k is greater than $m+h(\phi)p+1$, then we can substract p in order to remains lower than $m + h(\phi)p + 1$ (thanks to lemma 2.4). Thus we add (k, α) to S, so that (1) is still satisfied;
  - · if $\psi = \mathbf{F}^{-1}\alpha$, the argument is the same.

This process clearly ends, since the sum of sizes of formulas in S decreases at each step. Moreover, $|S \cup T| \leq |\phi|$ at the end.

Now consider the acceptable subloop L' of L containing the states we kept in $S \cup T$. We also possibly add some other states (this construction is the same as the one shown in figure 1). We write f for the function s.t. $\pi_{L'} \preccurlyeq_f \pi_L$. The remarks of the previous proof still apply. Then $\pi_{L'}, f^{-1}(i) \models \phi$.

Therefore, we show that for any $(j, \psi) \in T \cup S$, we have $\pi_{L'}, f^{-1}(j) \models \psi$: clearly, for each (j, ψ) in S, we have $\pi_{L'}, f^{-1}(j) \models \psi$ since ψ is an atomic proposition. For $(j, \psi) \in T$, several cases may arise:

- if $\psi = \alpha \vee \beta$, then either (j, α) or (j, β) is in $T \cup S$, and the result comes by ind. hyp.,
- if $\psi = \alpha \wedge \beta$, then (j, α) and (j, β) are in $T \cup S$, and the result also comes from the ind. hyp.,
- if $\psi = \mathbf{X}\alpha$, then $(j + 1, \alpha)$ (or $(j + 1 - p, \alpha)$) is in $T \cup S$. In the first case, the result is immediate. In the second case, it comes from lemma 2.4,
- for the other modalities $\mathbf{F}$,$\mathbf{F}^{-1}$ and $\mathbf{X}^{-1}$, the argument is the same. □

Lemma 2.11 *Let $\phi \in \mathbf{L}^+(\mathbf{G}, \mathcal{S})$ be a satisfiable formula, and L a loop s.t. $\pi_L, 0 \models \phi$. Let π_0 be the initial state of π_L, and $(a_k)_{k=0..i-1}$ be any sequence of i states, for an arbitrary natural i. We build the loop $L' = a_0 \cdot a_1 \cdots a_{i-1} \cdot (\pi_0)^\omega$. Then*

$\pi_{L'}, i \models \phi$.

This lemma ensures that we can find a witness with only one state for the satisfiable formula under study (by taking $i = 0$).

Proof The proof is by structural induction on ϕ:

- The result clearly holds for atomic propositions, and for conjunction and disjunction of smaller formulas;
- Assume that $\pi_L, 0 \models \mathbf{G}\psi$. Let $(a_k)_{k=0..i-1}$ be a sequence of i states. Then of course $\pi_L, 0 \models \psi$, and by induction hypothesis, for any integer j, we can build the loop
$$L_j = a_0 \cdot a_1 \cdots a_{i-1} \cdot \underbrace{\pi_0 \cdots \pi_0}_{j \text{ times}} \cdot (\pi_0)^\omega.$$
Then $\pi_{L_j}, i + j \models \psi$, by induction, for any natural j. Thus $\pi_{L'}, i \models \mathbf{G}\psi$.
- If $\pi_L, 0 \models \psi_1 \mathcal{S} \psi_2$, then obviously $\pi_L, 0 \models \psi_2$. By induction, $\pi_{L'}, i \models \psi_2$ for any sequence of i states $(a_k)_{k=0..i-1}$. This clearly entails that $\pi_{L'}, i \models \psi_1 \mathcal{S} \psi_2$.

□

Theorem 2.12 *Satisfiability and existential model-checking are* NP*-complete for* $\mathbf{L}(\mathbf{X}, \mathbf{X}^{-1}, \mathcal{S})$, $\mathbf{L}(\mathbf{F}, \mathbf{F}^{-1})$, $\mathbf{L}^+(\mathbf{F}, \mathbf{F}^{-1}, \mathbf{X}, \mathbf{X}^{-1})$, *and for their non-trivial fragments. Satisfiability is* NP*-complete for* $\mathbf{L}^+(\mathbf{G}, \mathcal{S})$ *and its non-trivial fragments.*

NP-easiness is a direct consequence of the previous results. NP-hardness was proved in Section 2.1.

3 PSPACE-complete problems

In this section, we prove PSPACE-hardness of verification problems for several fragments of PLTL.

The proofs are reductions from a tiling problem. Let C be a finite set of colors. A domino-type is a 4-tuple $\langle d^{up}, d^{down}, d^{left}, d^{right} \rangle$ of colors of C. Given a set $T \subseteq C^4$ of domino-types, and two integers m and n, tiling the $m \times n$-grid amounts to finding a function $f \colon [1, m] \times [1, n] \to T$ s.t.

$$\forall (i, j) \in [1, m-1] \times [1, n], f(i, j)^{right} = f(i+1, j)^{left}$$
$$\forall (i, j) \in [1, m] \times [1, n-1],\ f(i, j)^{up} =\ f(i, j+1)^{down}$$

We consider the following tiling problem, which is a slightly modified version of [9, prob. B_2]: given a set T of domino-types, a natural m (in unary), and two colors c_0 and c_1 of C, does there exist a natural n s.t. the $m \times n$-grid can be tiled, with the additional conditions that $f(1,1)^{down} = c_0$ and $f(m, n)^{up} = c_1$? This problem is PSPACE-complete.

Let $(C, T = \{d_1, \ldots d_p\}, m, c_0, c_1)$ be an instance of B_2. W.l.o.g., we may assume that the domino-types whose d^{up}-color is c_1 are numberred from 1 to q, and the other ones from $q + 1$ to p.

We build the Kripke structure shown on figure 2. The set of atomic propositions is $T \cup \{E\} \cup \{i = k \mid k = 1, \ldots, m\}$. The initial states are those where the

d^{down}-color is c_0 and the value of i is 1. All the transitions from a state labeled with $i = k$ to a state labeled with $i = k + 1$ are enabled for $k \leq m - 1$. For $i = m$, if the d^{up}-color is not c_1, then it is only possible to go to states labeled with $i = 1$, else it is only possible to go to state E.

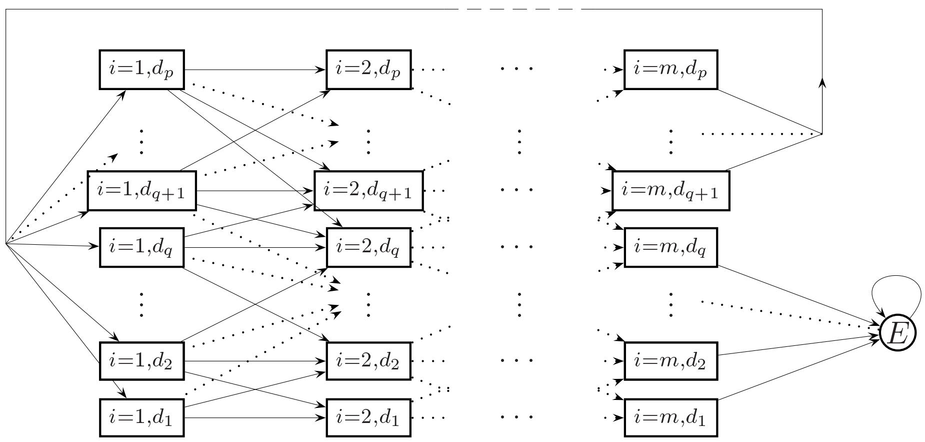


Figure 2. The Kripke structure K associated with our tiling problem

We now have to write formulas stating that

- colors are respected from left to right (ϕ_{horiz});
- colors are respected from top to bottom (ϕ_{vert}).

The initial and final conditions are true whenever the path eventually arrives in E.

3.1 The fragment $\mathbf{L}^+(\mathcal{U})$

[22] proves that model-checking and satisfiability are PSPACE-complete for $\mathbf{L}(\mathcal{U})$. The result here is a little stronger since we cannot, for instance, encode the $\mathbf{G}$ modality in $\mathbf{L}^+(\mathcal{U})$.

Theorem 3.1 *Existential model-checking for* $\mathbf{L}^+(\mathcal{U})$ *is* PSPACE-*hard.*

Proof

We simply have to express the three properties stated before with $\mathbf{L}^+(\mathcal{U})$ formulas:

- the "initial" and "final" conditions are satisfied: $\top\mathcal{U}E$
- the sequence of colors from left to right is correct:

$$\left(\bigwedge_{k=1}^{m-1}\bigwedge_{d\in T}(i=k\wedge d)\Rightarrow \Big(i=k\,\mathcal{U}(i=k+1\wedge \bigvee_{\substack{d'\in T\\ d'^{left}=d^{right}}} d')\Big)\right)\mathcal{U}E$$

- the sequence is also correct from bottom to top:

$$\left(\bigwedge_{k=1}^{m}\bigwedge_{d\in T}(i=k\wedge d)\Rightarrow\left(i=k\,\mathcal{U}\Big(\neg i=k\wedge\right.\right.$$
$$\left.\left.(\neg i=k)\mathcal{U}\big(E\vee(i=k\wedge\bigvee_{\substack{d'\in T\\ d'^{down}=d^{up}}}d')\big)\Big)\right)\right)\mathcal{U}E$$

A path in K satisfying the conjunction of those formulas eventually reaches E, after having run n times through a state where $i=1$ holds. The path gives rise to a function $f\colon[1,m]\times[1,n]\to T$ in the obvious way. This function is a tiling function since the path satisfies the ϕ_{horiz} and ϕ_{vert} conditions. Thus the (PSPACE-complete) problem B_2 is (polynomialy) reducible to model-checking $\mathbf{L}^+(\mathcal{U})$, and model-checking $\mathbf{L}^+(\mathcal{U})$ is PSPACE-hard. □

Corollary 3.2 *Satisfiability for $\mathbf{L}^+(\mathcal{U})$ is* PSPACE-*hard.*

Proof A classical method for such a proof is to reduce model-checking problem to satisfiability problem. However, since we cannot use or express $\mathbf{G}$ in $\mathbf{L}^+(\mathcal{U})$, we cannot encode the behaviour of a (general) Kripke structure. Thus we will reduce our tiling problem to the satisfiability problem, by encoding the Kripke structure of figure 2 into an $\mathbf{L}^+(\mathcal{U})$ formula.

We have to express that, until it reaches E, a path should meet the following properties:

- there is exactly one value for i in each state, as well as exactly one $d\in T$, unless we are in the state E;
- any path starts in a $i=1$ state, with $d^{down}=c_0$;
- i increases from 1 to m, and then goes back to 1 unless $d^{up}=c1$;
- we may go to the E state only from a $i=m$ state;

We define

$$uniq(S)\stackrel{\text{def}}{=}(\bigvee_{s\in S}s)\wedge(\bigwedge_{s\in S}\bigwedge_{s'\in S\smallsetminus\{s\}}s\Rightarrow\neg s').$$

The first property thus writes

$$\phi_{uniq}\stackrel{\text{def}}{=}\neg E\Leftrightarrow(uniq(\{i=k\}\wedge uniq(T)).$$

The third and fourth properties can be expressed through:

$$\phi_{incr}\stackrel{\text{def}}{=}\bigwedge_{k=1}^{m-1}(i=k\Rightarrow(i=k\,\mathcal{U}\,i=k+1))\wedge$$
$$((i=m\wedge\bigvee_{\substack{d\in T\\ d^{up}=c_1}}d)\Rightarrow(i=m\,\mathcal{U}(E)))\wedge$$
$$((i=m\wedge\neg\bigvee_{\substack{d\in T\\ d^{up}=c_1}}d)\Rightarrow(i=m\,\mathcal{U}\,i=1))$$

Thus the global reduction can be written as:

$$(i = 1 \land d^{down} = c_0) \land (\phi_{uniq} \land \phi_{incr} \land \phi_{vert} \land \phi_{horiz})\mathcal{U}E$$

□

Theorem 3.3 *Universal model-checking is* PSPACE*-hard for* $\mathbf{L}^+(\mathcal{U})$.

Proof This proof considers a slightly modified tiling problem: the input is the same, but the question is whether all correct tilings having c_0 as leftmost bottom color will eventually have c_1 as rightmost top color. Since PSPACE and co-PSPACE coincide, this problem clearly is PSPACE-complete. We will write a formula expressing that each path either does not represent a correct tiling, or eventually reaches the E state. Thus we write

$$\phi_{horiz} \stackrel{\text{def}}{=} \bigvee_{k=1}^{m-1} \bigvee_{d \in T} \top\mathcal{U}(i = k \land d \land (i = k\mathcal{U}(i = k+1 \land \bigwedge_{\substack{d' \in T \\ d'^{left} = d^{right}}} \neg d')))$$

$$\phi_{vert} \stackrel{\text{def}}{=} \bigvee_{k=1}^{m} \bigvee_{d \in T} \top\mathcal{U}(i = k \ \land d \land (i = k\mathcal{U}(\neg i = k \ \land$$
$$(\neg i = k\mathcal{U}(i = k \ \land \bigwedge_{\substack{d' \in T \\ d'^{down} = d^{up}}} \neg d')))))$$

A path satisfying those properties does not correspond to a correct tiling. Thus checking that all the paths satisfy $\phi_{horiz} \lor \phi_{vert} \lor \top\mathcal{U}E$ amounts to solving our tiling problem. □

3.2 PSPACE*-hardness for* $\mathbf{L}^+(\mathbf{F}, \mathbf{X})$, $\mathbf{L}^+(\mathbf{G}, \mathbf{X})$ *and* $\mathbf{L}_s^+(\mathbf{G}, \mathbf{X}^{-1})$

[22] shows that satisfiability and existential model-checking for $\mathbf{L}^+(\mathbf{F}, \mathbf{X})$ are NP-complete. We show here that universal model-checking is harder for that fragment.

Theorem 3.4 *Universal model-checking for* $\mathbf{L}^+(\mathbf{F}, \mathbf{X})$ *is* PSPACE*-hard.*

Proof The reduction is similar to the previous one, and formulas are even easier to write. □

By duality, we get

Corollary 3.5 *Existential model-checking and satisfiability are* PSPACE*-hard for* $\mathbf{L}^+(\mathbf{G}, \mathbf{X})$.

Proof For existential model-checking, the result comes by duality from the previous theorem. It was already proved in [4]. A reduction from existential model-checking to satisfiability for $\mathbf{L}(\mathbf{F}, \mathbf{X})$ is given in [22], and that reduction also applies to $\mathbf{L}^+(\mathbf{G}, \mathbf{X})$. □

Theorem 3.6 *Existential model-checking and satisfiability for* $\mathbf{L}_s^+(\mathbf{G}, \mathbf{X}^{-1})$ *are* PSPACE*-hard.*

Proof For existential model-checking, we consider the dual problem of the one we used for the proof of Theorem 3.3: given the same input, the question is whether there exists a correct tiling that never satisfies the “final” condition. For this, we simply have to write that the considered path satisfies the tiling conditions:

$$\phi_{horiz} \stackrel{\text{def}}{=} \bigwedge_{k=2}^{m} \bigwedge_{d\in T} \mathbf{G}(i = k \wedge d \Rightarrow \bigvee_{\substack{d'\in T \\ d'^{right}=d^{left}}} \mathbf{X}^{-1} d')$$

$$\phi_{vert} \stackrel{\text{def}}{=} \bigwedge_{k=1}^{m} \bigwedge_{d\in T} \mathbf{G}(i = k \wedge d \Rightarrow (\mathbf{X}^{-1^k} \bot \vee \bigvee_{\substack{d'\in T \\ d'^{up}=d^{down}}} \mathbf{X}^{-1^n} d'))$$

Checking that there exists a path satisfying $\phi_{horiz} \wedge \phi_{vert} \wedge \mathbf{G}\neg E$ amounts to solving the initial PSPACE-hard problem.

The reduction from existential model-checking to satisfiability given in [22] for $\mathbf{L}(\mathbf{F},\mathbf{X})$ can easily be adapted for $\mathbf{L}^+_s(\mathbf{G},\mathbf{X}^{-1})$. □

3.3 PSPACE*-hardness for* $\mathbf{L}^+_s(\mathbf{F},\mathcal{S})$ *and* $\mathbf{L}^+_s(\mathbf{G},\mathcal{S})$

Theorem 3.7 *Model-checking is* PSPACE*-hard for* $\mathbf{L}^+_s(\mathbf{F},\mathcal{S})$ *and* $\mathbf{L}^+_s(\mathbf{G},\mathcal{S})$*. Satisfiability is* PSPACE*-hard for* $\mathbf{L}^+_s(\mathbf{F},\mathcal{S})$*.*

Proof Those proofs are very similar to the previous ones:

- for existential model-checking of $\mathbf{L}^+_s(\mathbf{F},\mathcal{S})$ and $\mathbf{L}^+_s(\mathbf{G},\mathcal{S})$, the reduction is almost the same as in the proof of existential model-checking for $\mathbf{L}^+(\mathcal{U})$;
- for satisfiability, we simply encode the structure of figure 2 into $\mathbf{L}^+_s(\mathbf{F},\mathcal{S})$ formulae;
- For universal model-checking, the reductions are similar to the one of Theorem 3.3. □

Theorem 3.8 *Model-checking and satisfiability problems for the fragments* $\mathbf{L}^+(\mathcal{U})$*,* $\mathbf{L}^+(\mathbf{G},\mathbf{X})$*,* $\mathbf{L}^+_s(\mathbf{G},\mathbf{X}^{-1})$*,* $\mathbf{L}^+_s(\mathbf{F},\mathcal{S})$*, and for fragments of* PLTL *containing one of them, are* PSPACE*-complete. Model-checking is* PSPACE*-complete for* $\mathbf{L}^+_s(\mathbf{G},\mathcal{S})$*.*

Proof This is a direct consequence of [22, Theorem 4.1], and of Theorems in this section. □

4 Concluding remarks

The results we got are sufficient to completely classify all the considered fragments of PLTL w.r.t. the complexity of (existential and universal) model-checking and satisfiability problems.

This exhaustive case study led to several surprising results. We showed that existential and universal model-checking might have the different complexity for positive fragments (NP vs. PSPACE). We found only one case where existential model-checking and satisfiability have different theoretical complexity. On the other hand, we observe that using the symmetric past-time modalities of the allowed future modalities does not increase the complexity of verification problems. The same remark holds for the use of future modalities in the scope of past-time modalities. This all boils down to the conclusion that past is really cheap.

After this study on the effect of adding past into *fragments* of LTL, it would be interesting to look into when "past is for free" for *extensions* of that logic, such as CTL* (as far as we know, the complexity of model-checking for CTL* with linear past is still open [13]) or timed temporal logics ([2] proves that, for the validity problem over timed state sequences, past can be added for free in the Metric Temporal Logic from [12], but not in the Timed Propositional Temporal Logic of [1]).

References

[1] R. Alur and T. A. Henzinger. A really temporal logic. In *Proc. 30th IEEE Symp. Foundations of Computer Science (FOCS'89), Research Triangle Park, NC, USA, Oct. 1989*, pages 164–169, 1989.

[2] R. Alur and T. A. Henzinger. Real-time logics: Complexity and expressiveness. *Information and Computation*, 104(1):35–77, 1993.

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

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

[5] 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.

[6] K. Etessami, M. Y. Vardi, and T. Wilke. First order logic with two variables and unary temporal logic. In *Proc. 12th IEEE Symp. Logic in Computer Science (LICS'97), Warsaw, Poland, June–July 1997*, pages 228–235. IEEE Comp. Soc. Press, 1997.

[7] D. M. Gabbay. The declarative past and imperative future: Executable temporal logic for interactive systems. In *Proc. Workshop Temporal Logic in Specification, Altrincham, UK, Apr. 1987*, volume 398 of *Lecture Notes in Computer Science*, pages 409–448. Springer, 1989.

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

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

[10] J. A. W. Kamp. *Tense Logic and the Theory of Linear Order.* PhD thesis, UCLA, Los Angeles, CA, USA, 1968.

[11] Y. Kesten, Z. Manna, H. McGuire, and A. Pnueli. A decision algorithm for full propositional temporal logic. In *Proc. 5th Int. Conf. Computer Aided Verification (CAV'93), Elounda, Greece, June 1993*, volume 697 of *Lecture Notes in Computer Science*, pages 97–109. Springer, 1993.

[12] R. Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990.

[13] O. Kupferman and A. Pnueli. Once and for all. In *Proc. 10th IEEE Symp. Logic in Computer Science (LICS'95), San Diego, CA, USA, June 1995*, pages 25–35. IEEE Comp. Soc. Press, 1995.

[14] F. Laroussinie, N. Markey, and Ph. Schnoebelen. Temporal logic with forgettable past. In *Proc. 17th IEEE Symp. Logic in Computer Science (LICS'2002), Copenhagen, Denmark, July 2002*, pages 383–392. IEEE Comp. Soc. Press, 2002.

[15] F. Laroussinie and Ph. Schnoebelen. A hierarchy of temporal logics with past. *Theoretical Computer Science*, 148(2):303–324, 1995.

[16] O. Lichtenstein, A. Pnueli, and L. D. Zuck. The glory of the past. In *Proc. Logics of Programs Workshop, Brooklyn, NY, USA, June 1985*, volume 193 of *Lecture Notes in Computer Science*, pages 196–218. Springer, 1985.

[17] Z. Manna and A. Pnueli. Completing the temporal picture. *Theoretical Computer Science*, 83:97–130, 1991.

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

[19] Z. Manna and A. Pnueli. *Temporal Verification of Reactive Systems: Safety*. Springer, 1995.

[20] A. Pnueli. The temporal logic of programs. In *Proc. 18th IEEE Symp. Foundations of Computer Science (FOCS'77), Providence, RI, USA, Oct.-Nov. 1977*, pages 46–57, 1977.

[21] Y. S. Ramakrishna, L. E. Moser, L. K. Dillon, P. M. Melliar-Smith, and G. Kutty. An automata-theoretic decision procedure for propositional temporal logic with Since and Until. *Fundamenta Informaticae*, 17(3):271–282, 1992.

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

[23] M. Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994.