Temporal Logic with Forgettable Past

François Laroussinie[1], Nicolas Markey[1,2], Philippe Schnoebelen[1]

1 Laboratoire Spécification et Vérification, ENS de Cachan & CNRS UMR 8643,
61, avenue de Président Wilson, 94235 Cachan Cedex, France.
2 Laboratoire d'Informatique Fondamentale d'Orléans, Univ. Orléans & CNRS FRE 2490,
rue Léonard de Vinci, BP 6759, 45067 Orléans Cedex 2, France.

E-mail: {fl,markey,phs}@lsv.ens-cachan.fr

Abstract

We investigate NLTL, a linear-time temporal logic with forgettable past. NLTL can be exponentially more succinct than LTL + Past (which in turn can be more succinct than LTL). We study satisfiability and model checking for NLTL and provide optimal automata-theoretic algorithms for these EXPSPACE*-complete problems.*

1. Introduction

Temporal logic and verification. Temporal logic provides a fundamental framework for formally specifying systems and reasoning about them [4, 18]. Model checking techniques further allow the automatic verification that a finite-state model of a system satisfies a temporal logic specification [2, 1].

Temporal logic with past. Usually, specification and verification is done with *pure-future* temporal logics, i.e. logics where the modalities only refer to the future of the current time. It is well-known that temporal logics combining past and future modalities make some specifications easier to write and more natural [17]. However, allowing past-time makes verification algorithms harder to implement (though not necessarily from a complexity-theoretic viewpoint). Additionally, all the main temporal logics with past-time admit translations to their pure-future fragment [11, 7, 6, 8, 24, 15, 26].

Forgettable past and the N modality. Being able to refer to past moments is often useful, but there also exist situations where it is convenient to *forget the past*. Consider for example, the following temporal formula

$$\mathsf{G}(\mathtt{alarm} \Rightarrow \mathsf{F}^{-1}\,\mathtt{problem}) \qquad \text{(Spec1)}$$

where "F^{-1}" means "at some past time". (Spec1) states that "*whenever the alarm rings, then there has been some problem in the past*", i.e. the alarm does not ring without due cause. If now the alarm has a *reset* button, and we want to state that (Spec1) holds after any reset, the formula

$$\mathsf{G}\big(\mathtt{reset} \Rightarrow \mathsf{G}(\mathtt{alarm} \Rightarrow \mathsf{F}^{-1}\,\mathtt{problem})\big) \qquad \text{(Spec2)}$$

is not exactly what we aim at. With (Spec2), a problem that occurred *before the reset* may account for the alarm ringing, which is probably not what we had in mind.

For this kind of situations, Laroussinie and Schnoebelen [15, 16] proposed to use a new modality, N (read "*Now*"[1], or "*from now on*") that forgets all the past moments (see below for the formal semantics). With N, one can state the intended property of the alarm example via e.g.

$$\mathsf{G}\big(\mathtt{reset} \Rightarrow \mathsf{NG}(\mathtt{alarm} \Rightarrow \mathsf{F}^{-1}\,\mathtt{problem})\big) \qquad \text{(Spec3)}$$

Not much is known about N, except that the translations of LTL+Past and CTL^*+Past into (resp.) LTL and CTL^* carry over to LTL+Past+Now and CTL^*+Past+Now [16].

Our contribution. In this paper, we investigate $NLTL$, i.e. LTL+Past+Now, and give automata-theoretic decision procedures for model checking and satisfiability. These algorithms run in EXPSPACE, which is optimal: we show that both satisfiability and model checking are EXPSPACE-complete for $NLTL$. This came as a surprise since $NLTL$ does not appear to be a big departure from $PLTL$, i.e. LTL+Past, for which verification is (as for LTL) PSPACE-complete [21].

The increased complexity of $NLTL$ is partly explained by its expressive power: we prove that $NLTL$ can be exponentially more succinct than $PLTL$, and that $PLTL$ can

[1] No connection with the Now of H. Kamp, Formal properties of 'now', *Theoria*, 227–263, 1971.

in turn be exponentially more succinct than LTL. This last result is the first direct proof of a succinctness gap between $PLTL$ and LTL [2].

Finally we show how the model checking of a path can be done in polynomial time for $NLTL$ formulae and is in fact PTIME-complete (observe that the precise complexity of model checking a path is still an open-problem for LTL and $PLTL$ [3]).

Related works. Past-time is more popular in linear-time settings but branching-time settings exist, e.g. [8, 15, 14, 16]. Automata-theoretic methods for temporal-logics were pioneered by Vardi and Wolper, and they were adapted for LTL + Past and mu-calculus + Past in [17, 24, 12, 13] (these logics have PSPACE-complete verification problems). N can be encoded in richer formalisms, e.g. $QPLTL$ ($PLTL$ with arbitrary quantification over propositional variables) but $QPLTL$ and $QLTL$ have non-elementary verification problems [22]. "Chop" is another modality where part of the computation can be forgotten: here too verification becomes non-elementary [10, 20].

Plan of the paper. We provide the necessary definitions in § 2, study succinctness issues in § 3, and provide automata-theoretic decision procedures for $NLTL$ in § 4. Model checking a path is investigated in § 5.

2. Temporal logic with past

We first define $PLTL$ and only introduce N later.

Syntax of $PLTL$. Let $AP = \{p_1, p_2, \ldots\}$ be a countable set of atomic propositions. $PLTL$ formulae are given by:

$$\phi, \psi ::= \psi \wedge \phi \mid \neg\phi \mid \mathsf{X}\phi \mid \psi \mathsf{U} \phi \mid \mathsf{X}^{-1}\phi \mid \psi\mathsf{S}\phi \mid p_1 \mid p_2 \mid \ldots$$

where U reads "until", S reads "since", X is "next" and X^{-1} "previous". Standard abbreviations include: $\top$, $\psi \vee \phi$, $\psi \Rightarrow \phi$, $\psi \Leftrightarrow \phi$, ... Moreover we let $\mathsf{F}\phi \stackrel{\text{def}}{=} \top\mathsf{U}\phi$ ("ϕ will eventually be true"), $\mathsf{G}\phi \stackrel{\text{def}}{=} \neg\mathsf{F}\neg\phi$ ("ϕ is always true in the future") and, symmetrically, $\mathsf{F}^{-1}\phi \stackrel{\text{def}}{=} \top\mathsf{S}\phi$ and $\mathsf{G}^{-1}\phi \stackrel{\text{def}}{=} \neg\mathsf{F}^{-1}\neg\phi$. The size $|\phi|$ of a formula is its length in symbols.

S, X^{-1}, F^{-1} and G^{-1} are the *past modalities*, while U, X, F and G are the *future modalities*. The classical logic LTL is the pure-future fragment of $PLTL$.

[2] Regarding upper bounds, the direct translations in [6] is non-elementary. An elementary upper bound can be obtained from the elementary translations of counter-free Büchi automata into LTL formulae (see [27]). Regarding lower bounds, the non-existence of succinct translations has been conjectured (e.g. [13]) but not yet proved.

Semantics of $PLTL$. A *(linear-time) structure* (also, a *path*) is a pair (π, ξ) of an ω-sequence $\pi = \pi(0), \pi(1), \ldots$ of *positions*, with a mapping $\xi : \{\pi(0), \pi(1), \ldots\} \to 2^{AP}$ labeling each position $\pi(i)$ with the propositions that hold locally. We often simply write π for a structure when the labeling can be inferred from the context. Let (π, ξ) be a structure, i a nonnegative integer, and ϕ a $PLTL$ formula. We inductively define the relation $\pi, i \models \phi$, read "ϕ holds at position i in π", by:

$$\begin{array}{lll}
\pi, i \models p & \text{iff} & p \in \xi(\pi(i)),\\
\pi, i \models \phi \wedge \psi & \text{iff} & \pi, i \models \phi \text{ and } \pi, i \models \psi,\\
\pi, i \models \neg\phi & \text{iff} & \pi, i \not\models \phi,\\
\pi, i \models \mathsf{X}\phi & \text{iff} & \pi, i+1 \models \phi,\\
\pi, i \models \psi\mathsf{U}\phi & \text{iff} & \pi, j \models \phi \text{ for some } j \geq i\\
 & & s.t.\ \pi, k \models \psi \text{ for all } i \leq k < j,\\
\pi, i \models \mathsf{X}^{-1}\phi & \text{iff} & i > 0 \text{ and } \pi, i-1 \models \phi,\\
\pi, i \models \psi\mathsf{S}\phi & \text{iff} & \pi, j \models \phi \text{ for some } 0 \leq j \leq i\\
 & & s.t.\ \pi, k \models \psi \text{ for all } j < k \leq i.
\end{array}$$

We write $\phi \equiv \psi$ when ϕ and ψ are *equivalent*, i.e. when for all π and i, we have $\pi, i \models \phi$ iff $\pi, i \models \psi$. A less discriminating equivalence is *initial equivalence*, denoted $\equiv_i$, and defined by: $\phi \equiv_i \psi$ iff for all π, $\pi, 0 \models \phi$ iff $\pi, 0 \models \psi$. "Global" equivalence ($\equiv$) is the natural notion of equivalence, and it is substitutive. Initial equivalence is less well-behaved and, e.g., it is not substitutive under past-time contexts. Using $\equiv_i$ is meaningful when we compare two temporal specifications of some system, since these specifications have to hold at the initial positions. Observe that Gabbay's theorem, stating that "any $PLTL$ formula can be translated into an equivalent LTL formula", refers to initial equivalence: saying that $a\mathsf{U}b$ and $\mathsf{F}(b \wedge \mathsf{G}^{-1}(a \vee b))$ are equivalent is only correct with initial equivalence in mind.

Verification problems. Satisfiability is the first verification problem we consider. The problem is "given a formula ϕ, is there some π and i s.t. $\pi, i \models \phi$?" A variant problem asks whether ϕ is *initially satisfiable*, that is, satisfiable in the initial position of some structure. Clearly, the two problems are inter-reducible: ϕ is satisfiable iff $\mathsf{F}\phi$ is initially satisfiable and ϕ is initially satisfiable iff $\phi \wedge \neg\mathsf{X}^{-1}\top$ is satisfiable.

Model checking is our second verification problem. Here we consider a *Kripke structure* [3] K and a formula ϕ and asks whether $K \models \phi$, that is whether $\pi, 0 \models \phi$ for all infinite paths π in K that start from an initial node (a path in K is naturally interpreted as a linear-time structure).

It is well known that both model checking and satisfiability are PSPACE-complete for LTL and $PLTL$ [21].

[3] i.e. a structure $\langle Q, \to, Q_0, l\rangle$ where Q is a set of nodes, $\to \subseteq Q \times Q$ is a total transition relation, $Q_0 \subseteq Q$ is a set of initial nodes and $l : Q \to 2^{AP}$ a labeling of the nodes.

The N modality. The semantics of $PLTL$ assumes that past is *cumulative*, i.e., when time progresses, history grows ever larger. As a consequence, the entire history is always used for evaluating formulae with past modalities.

The N modality was introduced for situations where at some point one wants to forget the past, and start anew [15, 16]. Such a situation can be, e.g., the alarm+reset example seen in the Introduction.

Formally, we define $NLTL$, or $PLTL$+Now, by extending the syntax of $PLTL$ with the unary modality N and by extending the semantics with the following clause:

$$\pi, i \models \mathsf{N}\phi \quad \text{iff} \quad \pi^i, 0 \models \phi$$

where π^i is the i-th suffix of π starting from $\pi(i)$. Then, all the notions that are meaningful for $PLTL$ (satisfiability, global and initial equivalences, ...) apply equally to $NLTL$.

The following useful equivalences hold for any $NLTL$ formulae:

$$\begin{array}{llll} \mathsf{N}(\phi \wedge \psi) \equiv \mathsf{N}\phi \wedge \mathsf{N}\psi & \mathsf{N}\neg\phi & \equiv \neg\mathsf{N}\phi & \\ \mathsf{N}\mathsf{X}^{-1}\phi \quad \equiv \bot & \mathsf{N}(\phi \mathsf{S} \psi) & \equiv \mathsf{N}\psi & (\text{N-Laws}) \\ \multicolumn{3}{c}{\mathsf{N}\phi \equiv \phi \text{ if } \phi \text{ is pure-future}} & \end{array}$$

Also, N allows defining initial equivalence in term of global equivalence: $\phi \equiv_i \psi$ iff $\mathsf{N}\phi \equiv \mathsf{N}\psi$.

3. Succinctness of temporal logics with past

Gabbay's (and Kamp's) theorem implies that $PLTL$ and LTL have the same expressive power [11, 7, 6]. Gabbay's proof associates with any $PLTL$ formula ψ a (globally) equivalent but separated ψ' (i.e. ψ' is a Boolean combination of pure-past and pure-future formulae). With this separation theorem, it is easy to prove that $NLTL$ also can be translated to LTL (modulo initial equivalence): one just applies the N-Laws on the separated formulae (see [15]).

In this section we show these three "equally expressive" logics can be distinguished in terms of succinctness. These results provide a formal justification of the claim that the addition of past-time modalities make some specifications easier to write. That there can exist a succinctness gap between $PLTL$ and LTL has been conjectured by many researchers in the field, but Theorem 3.1 provides, as far as we know, the first proof.

Theorem 3.1. *$PLTL$ can be exponentially more succinct than LTL.*

Proof. Assume $\{p_0, p_1, \ldots, p_n\}$ are atomic propositions and let ψ_n be the following $PLTL$ formula:

$$\mathsf{G}\Big(\big(\bigwedge_{i=1}^{n}(p_i \Leftrightarrow \mathsf{F}^{-1}(\neg\mathsf{X}^{-1}\top \wedge p_i))\big) \Rightarrow \big(p_0 \Leftrightarrow \mathsf{F}^{-1}(\neg\mathsf{X}^{-1}\top \wedge p_0)\big)\Big) \qquad (\psi_n)$$

ψ_n uses $\neg\mathsf{X}^{-1}\top$ to characterize the initial position of a path π, so that $\pi \models \psi_n$ iff all positions in π that agree with the initial position $\pi(0)$ on $p_1, \ldots, p_n$ also agree on p_0. Let φ_n be an LTL formula s.t. $\psi_n \equiv_i \varphi_n$ (it is easy to come up with such a φ_n, e.g. by considering all possible valuations for $p_0, p_1, \ldots, p_n$). Since φ_n is pure-future, $\mathsf{G}\varphi_n$ states that "any two future positions that agree on $p_1, \ldots, p_n$ also agree on p_0". But this latter property can only be expressed by $PLTL$ (or LTL) formulae of size $\Omega(2^n)$ [5, Theorem 2] [4]. Hence $|\varphi_n|$ is in $\Omega(2^n)$ while $|\psi_n|$ is in $O(n)$. ☐

Remark 3.2. *The statement of Theorem 3.1 (and 3.3) can be sharpened:*

1. *Observe that the succinctness gap already occurs with temporal formulae from the $L(\mathsf{F}, \mathsf{F}^{-1}, \mathsf{X}^{-1})$ fragment, and having a fixed temporal depth. We could adapt the proof and further restrict to $L(\mathsf{F}, \mathsf{F}^{-1})$.*
2. *The proof that $|\varphi_n|$ is in $\Omega(2^n)$ even shows that φ_n has $\Omega(2^n)$ distinct subformulae, so that φ_n cannot succinctly be represented as a dag (sharing common subformulae).*
3. *ψ_n uses $n+1$ distinct propositions but, using standard encoding techniques (see e.g. [3]), one shows the succinctness gap occurs even for formulae with a single proposition (at the cost of the fixed temporal depth).*

Adding N further allows more conciseness:

Theorem 3.3. *$NLTL$ can be exponentially more succinct than $PLTL$.*

Proof. Let ψ'_n be the $NLTL$ formula given by $\psi'_n \stackrel{\text{def}}{=} \mathsf{G}\mathsf{N}\psi_n$. ψ'_n too states that "any two future positions that agree on $p_1, \ldots, p_n$ also agree on p_0", so that it only has $PLTL$ equivalents of size $\Omega(2^n)$. ☐

Observe that a single occurence of N is sufficient for the succinctness gap.

4. An automata-theoretic approach to $NLTL$ verification

In this section we address satisfiability and model checking problems for $NLTL$. We start by an algorithm for satisfiability. This algorithm associates with an $NLTL$ formula ϕ an *alternating Büchi automaton* $\mathcal{A}_\phi$ that accepts the models of ϕ. Then $\mathcal{A}_\phi$ can also be used for model checking.

The literature contains many algorithms that build automata recognizing the models of temporal formulae. For

[4] A Büchi automaton for $\mathsf{G}\varphi_n$ must record all valuations that it sees and then requires $\Omega(2^{2^n})$ states. Since $PLTL$ formulae can be translated to Büchi automata with a single exponential blowup, the claim follows.

linear-time logics, one can distinguish between two different kinds of constructions. First there are methods based on classical (non-deterministic) Büchi automata whose size is exponential in the size of the formula (e.g. [23, 17]). Secondly, for pure future logics as LTL, there exist approaches based on *alternating* Büchi automata of only polynomial size [25]. All these methods offer (optimal) algorithms running in PSPACE since non-emptiness of a Büchi automaton can be decided in logarithmic space, while it requires polynomial space for alternating Büchi automata.

Our method produces an *alternating* Büchi automaton of *exponential size*. Then our algorithms for satisfiability and model checking run in EXPSPACE. We show in § 4.2 that these algorithms are optimal: satisfiability and model-checking are EXPSPACE-complete for $NLTL$.

4.1. Alternating Büchi automata for $NLTL$ formulae

Alternating automata. An alternating automaton with a generalized Büchi acceptance condition is a tuple $\mathcal{A} = \langle \Sigma, S, \rho, S_0, \mathcal{F} \rangle$ where Σ is a finite alphabet, S is a finite set of states, $\rho : S \times \Sigma \to \mathcal{B}^+(S)$ is a transition function ($\mathcal{B}^+(S)$ is the set of positive Boolean combinations over S and also contains $\perp$) whose semantics is described below, $S_0 \subseteq S$ is the set of initial states and $\mathcal{F} = \{F_1, \ldots\}$ is a set of sets of accepting states ($F_i \subseteq S$ for any i).
A run R of $\mathcal{A}$ over an infinite word $w \in \Sigma^\omega$ is an infinite S*-labeled tree* (viz $R = (\tau, r)$ where τ is a tree and $r : Nodes(\tau) \to S$ assigns an element of S to each node of τ). We further require that the root ε of τ is labeled with some initial state and that R respects the transition relation: that is, for any node x with children $x_1, \ldots, x_k$, if x has depth i then $\{r(x_1), \ldots, r(x_k)\} \models \rho(r(x), w_i)$. Here the formula $\theta = \rho(r(x), w_i)$ describes admissible sets of states in the obvious way. A run is *accepting* if, along every branch, the set of states that are visited infinitely often intersects every F_i non-vacuously.

W.l.o.g. the branching degree of accepting runs can be bounded: if the transition relation only contains Boolean formulae θ of the form $\bigvee_j \bigwedge_{i=1}^k s_{j,i}$, then an accepting run with branching degree $> k$ can be pruned to degree k and remain accepting.

Construction of $\mathcal{A}_\varphi$. Let φ be an $NLTL$ formula. We define $\mathrm{CL}(\varphi)$, the *closure of* φ, as the smallest set of formulae containing $\top$, $\mathsf{X}^{-1}\top$, all subformulae of φ, $\mathsf{X}(\psi_1 \mathsf{U} \psi_2)$ for any subformula $\psi_1 \mathsf{U} \psi_2$ of φ, $\mathsf{X}^{-1}(\psi_1 \mathsf{S} \psi_2)$ for any subformula $\psi_1 \mathsf{S} \psi_2$ of φ, and the negations of all these formulae (we identify $\neg\neg\psi$ with ψ). Classically, an *atom* A of φ is a locally coherent subset of $\mathrm{CL}(\varphi)$ [17]. For $NLTL$ the coherency conditions are:

- $\top \in A$,
- if $\psi \in \mathrm{CL}(\varphi)$ then $\psi \in A$ iff $\neg\psi \notin A$,
- if $\psi_1 \wedge \psi_2 \in \mathrm{CL}(\varphi)$ then $\psi_1 \wedge \psi_2 \in A$ iff $\psi_1 \in A$ and $\psi_2 \in A$,
- if $\psi_1 \vee \psi_2 \in \mathrm{CL}(\varphi)$ then $\psi_1 \vee \psi_2 \in A$ iff $\psi_1 \in A$ or $\psi_2 \in A$,
- if $\psi_1 \mathsf{U} \psi_2 \in \mathrm{CL}(\varphi)$ then $\psi_1 \mathsf{U} \psi_2 \in A$ iff $\psi_2 \in A$ or $\psi_1, \mathsf{X}(\psi_1 \mathsf{U} \psi_2) \in A$,
- if $\psi_1 \mathsf{S} \psi_2 \in \mathrm{CL}(\varphi)$ then $\psi_1 \mathsf{S} \psi_2 \in A$ iff $\psi_2 \in A$ or $\psi_1, \mathsf{X}^{-1}(\psi_1 \mathsf{S} \psi_2) \in A$,
- if $\mathsf{X}^{-1}\psi \in \mathrm{CL}(\varphi)$ then $\mathsf{X}^{-1}\psi \in A$ implies $\mathsf{X}^{-1}\top \in A$,
- if $\mathsf{N}\psi \in \mathrm{CL}(\varphi)$ and $\neg\mathsf{X}^{-1}\top \in A$ then $\mathsf{N}\psi \in A$ iff $\psi \in A$.

The set of atoms of φ is denoted $\mathrm{Atom}(\varphi)$. Since $|\mathrm{CL}(\varphi)| \leq 4|\varphi|$, there are at most $2^{4|\varphi|}$ atoms. We say that an atom is *initial* if it contains $\neg\mathsf{X}^{-1}\top$; the set of initial atoms is denoted $\mathrm{InitAtom}(\varphi)$.

Before formally defining $\mathcal{A}_\varphi$, we explain the intuition behind its workings: assume that after reading the i first positions of some path π, $\mathcal{A}_\varphi$ is in a state labeled by A. This means that all $\psi \in A$ hold at π, i (and only these since A is coherent). The past-formulae in A have been *observed* to hold by $\mathcal{A}_\varphi$, the future-formulae have been *guessed* and will have to be checked later, the mixed past+future formulae combine observations and guesses. When $\mathcal{A}_\varphi$ moves from $\pi(i)$ to $\pi(i+1)$, it updates A while respecting the valuation for $\pi(i)$ and forks an alternative branch where the observation of past-formulae restarts with an empty history. The "update" branch goes on classically while the forked branch proceeds as if reading π^i with no past and verifies all N properties contained in A.

Definition 4.1. *$\mathcal{A}_\varphi = \langle \Sigma, S, \rho, S_0, \mathcal{F} \rangle$ is given by:*

- $\Sigma = 2^{AP}$,
- $S = \mathrm{Atom}(\varphi)$,
- $S_0 = \{A \in \mathrm{InitAtom}(\varphi) \mid \varphi \in A\}$,
- $\rho(A, \sigma) = \displaystyle\bigvee_{A' \in \mathrm{Succ}(A,\sigma)} \Big(A' \wedge \bigvee_{A'' \in \mathrm{Now}(A')} A'' \Big)$ *with:*

- $\mathrm{Now}(A') \stackrel{\text{def}}{=} \{A'' \in \mathrm{InitAtom}(\varphi) \mid \forall \mathsf{N}\psi \in \mathrm{CL}(\varphi),\ \mathsf{N}\psi \in A' \Leftrightarrow \mathsf{N}\psi \in A''\}$,

- *for A and σ s.t. for all $p \in AP$, $p \in A$ iff $p \in \sigma$:*

$$\mathrm{Succ}(A,\sigma) \stackrel{\text{def}}{=} \{A' \in \mathrm{Atom}(\varphi) \mid \mathsf{X}^{-1}\top \in A' \text{ and } \forall \mathsf{X}\psi \in \mathrm{CL}(\varphi),\ \mathsf{X}\psi \in A \Leftrightarrow \psi \in A' \text{ and } \forall \mathsf{X}^{-1}\psi \in \mathrm{CL}(\varphi),\ \psi \in A \Leftrightarrow \mathsf{X}^{-1}\psi \in A'\},$$

otherwise, $\mathrm{Succ}(A,\sigma) \stackrel{\text{def}}{=} \perp$,

- *if there is no formula φ of the form $\psi \mathsf{U} \psi'$ in $\mathrm{CL}(\varphi)$, $\mathcal{F}$ is $\{\mathrm{Atom}(\varphi)\}$. Otherwise, let $\{\varphi_1 \mathsf{U} \psi_1, \ldots, \varphi_k \mathsf{U} \psi_k\}$ be the set of all U-formulae in $\mathrm{CL}(\varphi)$. Then $\mathcal{F} = \{F_1, \ldots, F_k\}$ with: $F_i \stackrel{\text{def}}{=} \{A \in \mathrm{Atom}(\varphi) \,|\, \neg \mathsf{X}^{-1} \top \in A$ or $\psi_i \in A$ or $\neg(\varphi_i \mathsf{U} \psi_i) \in A\}$.*

Observe that the structure of ρ allows us to restrict our attention to *binary* runs of $\mathcal{A}_\varphi$ (where every node has exactly two successors) s.t. one and only one child of each node is labeled with an initial atom. In the following we identify an accepting binary run (τ, r) with its labeling function r.

We can now state and prove the correctness of our construction by formalizing the intuitions we gave before Def. 4.1. Given an accepting run r, we define the *level* of node x in r (written $\mathrm{l}_r(x)$) as the depth of the closest ancestor y of x that carries an initial atom (such an ancestor must exist since ε carries an initial atom). Clearly $\mathrm{l}_r(x) \leq |x|$. We have:

Lemma 4.2. *Let φ be an NLTL formula, ψ a formula in $\mathrm{CL}(\varphi)$, and w a word in Σ^ω. If there exists an accepting binary run r of $\mathcal{A}_\varphi$ over w, then for any node x with $|x| = i$, $\mathrm{l}_r(x) = i_0$ and $r(x) = A$, $\psi \in A$ iff $w^{i_0}, i - i_0 \models \psi$.*

Proof. By induction over ψ (full details can be found in Appendix A). ☐

Proposition 4.3. *Let φ be an NLTL formula, then φ is initially satisfiable iff there exists an accepting run in $\mathcal{A}_\varphi$.*

Proof. ($\Leftarrow$) Direct from Lemma 4.2.
($\Rightarrow$) See Appendix B. ☐

The corollary is that $\mathcal{A}_\varphi$ recognizes exactly the set of words over Σ for which φ is initially true.

Satisfiability checking. A formula φ is satisfiable if and only if $\mathsf{F}\varphi$ is initially satisfiable, and this can be checked by looking for accepting runs in $\mathcal{A}_{\mathsf{F}\varphi}$, then we have:

Theorem 4.4. *Satisfiability for NLTL formulae can be decided in* EXPSPACE.

Proof. The size of $\mathcal{A}_{\mathsf{F}\varphi}$ is exponential in φ and nonemptiness of alternating Büchi automaton can be solved in space polynomial in the size of the automaton [19]. ☐

Model checking. With $\mathcal{A}_\varphi$ available, deciding whether $K \models \varphi$ can be reduced to a language inclusion question, i.e. checking whether $L(\mathcal{A}_K) \subseteq L(\mathcal{A}_\varphi)$. Here $\mathcal{A}_K$ is simple K seen as a Büchi automaton (a classical automaton with trivial acceptance condition). Thus one has:

Theorem 4.5. *Model checking Kripke structures for NLTL formulae can be decided in* EXPSPACE.

As a corollary, we get that model checking for *CTL** +Past+Now is in EXPSPACE too.

Remark 4.6. *The* program complexity *of model checking NLTL formulae is* NLOGSPACE*-complete, as for LTL (it suffices to translate the NLTL formula into an initially equivalent LTL formula).*

The specification complexity *is clearly* EXPSPACE-*complete (the proof of* EXPSPACE*-hardness of model checking could use a fixed Kripke structure).*

4.2. EXPSPACE-hardness

The decision procedures for *NLTL* we just saw are optimal in the following sense:

Proposition 4.7. *Satisfiability and model checking for NLTL are* EXPSPACE*-hard.*

Proof. By reduction from a domino-tiling problem for grids with exponential size. Let $C = \{c_1, \ldots, c_k\}$ be a set of *colors*. A *domino-type* is a 4-tuple $\langle d_{\text{down}}, d_{\text{left}}, d_{\text{up}}, d_{\text{right}} \rangle$ of colors. Given a set $T \subseteq C^4$ of domino-types, and two integers m and n, "tiling the $m \times n$-grid with T" means finding a mapping $f\colon [0, m-1] \times [0, n-1] \to T$ s.t. for all i, j

$$\begin{aligned} f(i,j)_{right} &= f(i+1,j)_{left} && \text{if } i < m-1, \\ f(i,j)_{up} &= f(i,j+1)_{down} && \text{if } j < n-1. \end{aligned}$$

The problem of deciding, given a set T of domino-types, a natural number m (written in unary), and two domino-types $d_{\text{init}}, d_{\text{final}} \in T$, whether there exists a natural n $s.t.$ the $2^m \times n$-grid can be tiled, with the additional conditions that $f(0,0) = d_{\text{init}}$ and $f(2^m - 1, n-1) = d_{\text{final}}$, is EXPSPACE-complete [9].

Let $\mathcal{I} = (C, T, m, d_{\text{init}}, d_{\text{final}})$ be such an instance. We build a Kripke structure $K_\mathcal{I}$ as follows: $\{b_1^+, \ldots, b_m^+, b_1^-, \ldots, b_m^-\}$ are $2m$ atomic propositions, that we will use to encode the value of a m-bits counter numbering the cells of one line of the grid. Each domino-type $d \in T$ is also an atomic proposition. $K_\mathcal{I}$ is depicted on figure 1.

Traversing $K_\mathcal{I}$ from left-to-right picks values $\pm$ for the bits $b_1, \ldots, b_m$ and a domino-type: this encodes the abscissa $0 \leq i \leq 2^m - 1$ of a cell (i,j) and its coloring (nb: b_1 is the least significant bit). A tiling of the grid is encoded by a path in $K_\mathcal{I}$, listing cells from left to right and from bottom to top, with an exit to E when n lines have been listed. Observe that, when cell (i,j) is listed, only i is given explicitly.

We now write an *NLTL* formula stating that a path in $K_\mathcal{I}$ does indeed encode a tiling. This combines several subformulae, where $b_i^\pm$ is an abbreviation for $b_i^+ \vee b_i^-$:

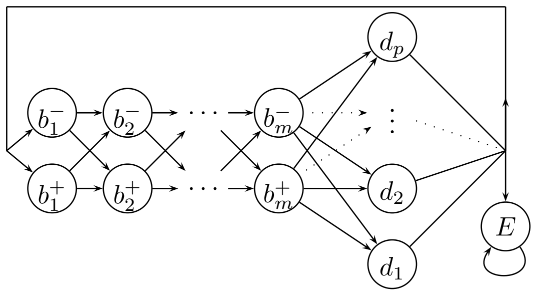


Figure 1. The Kripke structure associated with a tiling problem

(1) the path starts with a d_{init} numbered 0, and ends with a d_{final} numbered $2^m - 1$:

$$\bigwedge_{i=1}^{m} \mathsf{X}^{i-1} b_i^- \wedge \mathsf{X}^m d_{\text{init}}$$
$$\wedge\ \mathsf{F}\Big(\bigwedge_{i=1}^{m} \mathsf{X}^{i-1} b_i^+ \wedge \mathsf{X}^m d_{\text{final}} \wedge \mathsf{X}^{m+1} E\Big)$$

(2) the cells are listed in increasing order modulo 2^m, i.e. b_1 changes with each new cell, while b_{i+1} only changes if b_i switched from b_i^+ to b_i^-:

$$\mathsf{G}\Big[\big(b_1^{\pm} \wedge \neg \mathsf{X}^{m+1} E\big) \Rightarrow$$
$$(b_1^+ \Leftrightarrow \mathsf{X}^{m+1} b_1^-) \wedge \bigwedge_{i=1}^{m-1} \begin{array}{l}(\mathsf{X}^i b_{i+1}^+ \Leftrightarrow \mathsf{X}^{m+i+1} b_{i+1}^-)\\ \Leftrightarrow (\mathsf{X}^{i-1} b_i^+ \wedge \mathsf{X}^{m+i} b_i^-)\end{array}\Big]$$

(3) two adjacent cells (except at end of line) have the same color on the shared edge:

$$\bigwedge_{d \in T} \mathsf{G}\Big(d \Rightarrow \bigvee_{\substack{d' \in T \\ d'_{\text{left}} = d_{\text{right}}}} \mathsf{X}^{m+1} d' \ \vee\ \mathsf{X}E \ \vee\ \bigwedge_{i=1}^{m} \mathsf{X}^i b_i^-\Big)$$

(4) two neighbor cells (i, j) and $(i, j+1)$ *in a column* have the same color on the shared edge. This is where we use past-time modalities and N. We first define the following abbreviation:

$$\phi_{up} \ :=\ b_1^{\pm}\ \wedge\ \bigwedge_{i=1}^{m} (\mathsf{X}^{i-1} b_i^+ \Leftrightarrow \mathsf{F}^{-1}(\neg \mathsf{X}^{-1}\top \wedge \mathsf{X}^{i-1} b_i^+))$$

ϕ_{up} is inspired by the ψ_n formula from § 3: ϕ_{up} states that the value of the bits $b_1, \ldots, b_m$ coincide with the value they had at the beginning of the path (assuming we are at some $b_1^{\pm}$ and the path also starts at some $b_1^{\pm}$). Now, using N to forget everything before cell (i, j), we can use $\neg \phi_{up} \mathsf{U} \phi_{up}$ to find the next cell with same i, i.e. cell $(i, j+1)$:

$$\mathsf{G}\Big(b_1^{\pm} \Rightarrow$$
$$\mathsf{N}\mathsf{X}^m \bigwedge_{d \in T} \big(d \Rightarrow \neg \phi_{up} \mathsf{U}(E\ \vee\ \phi_{up} \wedge \bigvee_{\substack{d' \in T \\ d'_{\text{down}} = d_{\text{up}}}} \mathsf{X}^m d')\big)\Big)$$

Finally, $K_{\mathcal{I}}$ contains a path satisfying the $NLTL$ formula iff $\mathcal{I}$ has a solution to our tiling problem, hence model checking $NLTL$ formulas is EXPSPACE-hard.

Satisfiability for $NLTL$ is also EXPSPACE-hard since we can reduce model checking to satisfiability by encoding the Kripke structure K in a temporal formula. $\square$

Note that EXPSPACE-hardness already occurs with a single occurrence of N under the scope of one G. With standard encoding techniques, we could prove EXPSPACE-hardness for the fragment without $\mathsf{X}, \mathsf{X}^{-1}$, or for a fragment with a fixed number of atomic propositions.

5. Model checking a path

That model checking $NLTL$ formulae is EXPSPACE-complete may look daunting at first sight, especially when $PLTL$ is only PSPACE-complete. But there exist situations where $NLTL$ can be handled efficiently. Below we show that model checking $NLTL$ formulae on linear Kripke structures (i.e., structures where nodes have only one successor), can be done in polynomial-time. This result is interesting because the polynomial-time algorithm is not trivial, and it can be used e.g. for on-the-fly model checking.

But these questions also have a theoretical interest. Model checking linear structures is called "model checking a path" in [3]. We prove the problem is PTIME-complete for $NLTL$ while the precise complexity of model checking a path for LTL and $PLTL$ are still not known [5].

Theorem 5.1. *Model checking an NLTL-formula ϕ along an ultimately-periodic path can be done in polynomial-time.*

Note that the path π_L associated with a finite and linear Kripke structure L is ultimately-periodic. In the sequel, a *loop of type* (m, p) is a linear Kripke structure where the initial part of π_L has length m and the periodic part has length p. The proof of Theorem 5.1 relies on the following lemma.

Lemma 5.2. *For any loop L of type (m, p), for any NLTL formula ϕ with at most $h(\phi)$ nested past-time modalities, and any $k \geq m + p \cdot h(\phi)$, $\pi_L, k \models \phi$ iff $\pi_L, k + p \models \phi$.*

[5] In particular, it is not known whether model checking a path against LTL formulae can be done in NLOGSPACE or is PTIME-complete [3].

Proof. By induction on the structure of ϕ. We only consider the cases where ϕ has N or a past-time modality at its root. The other cases, Boolean operators or pure-future modalities, are easy.

- Assume ϕ is some $\mathsf{X}^{-1}\phi_1$. Then $h(\phi) = h(\phi_1) + 1$, and $k \geq m + p.h(\phi)$ implies $k - 1 \geq m + p.h(\phi_1)$. Now $\pi_L, k \models \phi$ iff $\pi_L, k-1 \models \phi_1$ iff (by ind. hyp.) $\pi_L, k-1+p \models \phi_1$ iff $\pi_L, k+p \models \phi$.

- Assume ϕ is some $\phi_1 \mathsf{S} \phi_2$ so that $h(\phi) = \max(h(\phi_1), h(\phi_2)) + 1$. We first prove the ($\Rightarrow$) direction: assume $\pi_L, k \models \phi$, then there exists some $k' \leq k$ s.t. $\pi_L, k' \models \phi_2$, and for $k' < l \leq k$, $\pi_L, l \models \phi_1$. If $k' \geq k - p$, then by ind. hyp. $\pi_L, k' + p \models \phi_2$ and $\pi_L, l + p \models \phi_1$ for $k' < l \leq k$, so that $\pi_L, k + p \models \phi$. If $k' < k - p$ then by ind. hyp. $\pi_L, l \models \phi_1$ for all $l = k, \ldots, k + p$ so that again $\pi_L, k + p \models \phi$.

  For the ($\Leftarrow$) direction assume $\pi_L, k + p \models \phi$, so that there exists some $k' \leq k + p$ with $\pi_L, k' \models \phi_2$, and for $k' < l \leq k + p$, $\pi_L, l \models \phi_1$. If $k' \leq k$ one directly obtains $\pi_L, k \models \phi$. Otherwise $k' > k$ and by ind. hyp. $\pi_L, k' - p \models \phi_2$ and $\pi_L, l - p \models \phi_1$ for $k' < l \leq k + p$. Again $\pi_L, k \models \phi$.

- Assume ϕ is some $\mathsf{N}\phi_1$. Here $\pi_L, k \models \phi$ iff $\pi_L^k \models \phi_1$, and $\pi_L, k + p \models \phi$ iff $\pi_L^{k+p} \models \phi_1$. But since L has type (m, p), the suffixes π_L^k and π_L^{k+p} are isomorphic. Hence $\pi_L, k \models \phi$ iff $\pi_L, k + p \models \phi$.

$\square$

Proof of Theorem 5.1. Given a loop L of type (m, p), and an *NLTL* formula ϕ of past-height h, Lemma 5.2 allows reducing any question "does $\pi_L^l, k \models \psi$?" (where $l, k \in \mathbb{N}$ and ψ is a subformula of ϕ) to an equivalent "does $\pi_L^{l'}, k' \models \psi$?" where this time $0 \leq l' < m + p$ and $0 \leq k' < m + (h+1)p$. All these questions are solved once and for all by filling a Boolean array $V[l', k', \psi]$ in such a way that $V[l', k', \psi] = \top$ iff $\pi_L^{l'}, k' \models \psi$. Filling V is done through dynamic programming techniques, starting with the smallest subformulae ψ. For ψ of the form $\mathsf{N}\psi'$, we fill $V[l', k', \psi]$ with the (previously computed) value of $V[l'', 0, \psi']$, where l'' is $l' + k'$ if $l' + k' < m + p$, or $l' + k'$ brought back to the interval $[0, m + p)$ by subtracting p enough times. Dealing with the other cases requires something similar to the *CTL* model checking algorithm, e.g. $V[l', k', \mathsf{X}^{-1}\psi]$ receives true iff $k' > 0$ and $V[l', k' - 1, \psi] = \top$. Eventually, the algorithm finishes in time $O(h(\phi) \times |\phi| \times |L|)$, which is $O(|\phi|^2 \times |L|)$. $\square$

It turns out polynomial-time is also a lower bound for this problem.

Proposition 5.3. *Model checking NLTL along one path is* PTIME*-hard.*

Proof. By reduction from CIRCUIT-VALUE, where it is well known that only considering synchronous, alternating and monotone circuits is no loss of generality. We illustrate the reduction on an example. Consider the circuit $\mathcal{C}$ from Fig. 2 and write $E_\mathcal{C}$ for the set of edges linking one gate to one of its inputs (in our example, $E_\mathcal{C} = \{(n_1, n_2), (n_1, n_3), \ldots, (n_{12}, n_{14})\}$). We denote $v_\mathcal{C}(n)$ the (Boolean) value obtained by evaluating node n in the obvious way.

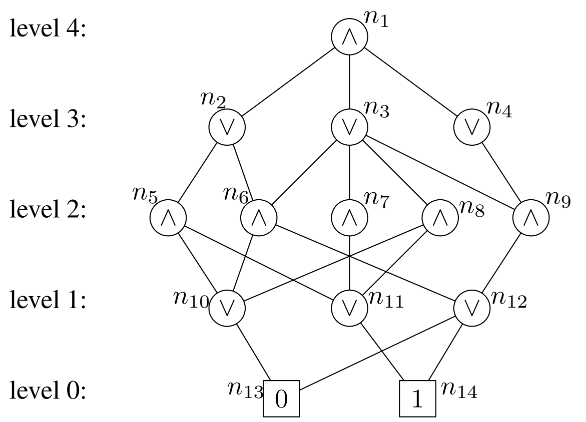
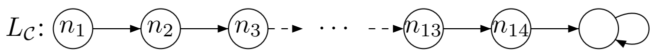


Figure 2. An instance of CIRCUIT-VALUE**.**

With $\mathcal{C}$, we associate a loop $L_\mathcal{C}$ listing all nodes in some height-respecting order, as illustrated in the following picture where node names are also used as propositions.



Let ϕ_{next} be defined as follows:

$$\phi_{next} := \bigvee_{(n,n') \in E_\mathcal{C}} (n' \wedge \mathsf{F}^{-1}(n \wedge \neg \mathsf{X}^{-1} \top)).$$

Then $L_\mathcal{C}^i, k \models \phi_{next}$ iff $(n_i, n_{i+k}) \in E_\mathcal{C}$. We now construct the following formulae, for $k > 0$:

$$\begin{aligned}
\phi_0 &:= n_{14} \\
\phi_{2k+1} &:= \mathsf{NF}\big(\phi_{next} \wedge \phi_{2k}\big) \\
\phi_{2k+2} &:= \mathsf{NG}\big(\phi_{next} \Rightarrow \phi_{2k+1}\big)
\end{aligned}$$

An easy induction on p shows that, for every node n_i at some level p in $\mathcal{C}$, $v_\mathcal{C}(n_i) = \top$ iff $L_\mathcal{C}, n_i \models \phi_p$. Thus, $v(\mathcal{C}) = v_\mathcal{C}(n_1) = \top$ iff $L_\mathcal{C} \models \phi_4$. Note that the reduction is clearly logspace, with a *NLTL* formula of size $O(|\mathcal{C}|^2)$. $\square$

6. Conclusions

We investigated *NLTL*, the linear-time temporal logic with past augmented with N, a new modality that allows forgetting the past.

Some specifications are easier and more natural in $NLTL$ than in $PLTL$ (i.e., LTL+Past). That $NLTL$ offers more expressive power can be stated formally as a succinctness gap between $NLTL$ and $PLTL$. An interesting byproduct of this study is a direct proof of the suspected succinctness gap between LTL+Past and LTL.

With any $NLTL$ formula φ, we associate an alternating Büchi automaton $\mathcal{A}_\varphi$ that accepts the models of φ. This provides automata-theoretic decision methods for satisfiability and model checking of $NLTL$ formulae. $\mathcal{A}_\varphi$ has exponential size, so that the decision methods are in EXPSPACE, but we show the problems are EXPSPACE-complete so that our decision methods are optimal.

References

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

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

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

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

[5] 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.

[6] 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.

[7] 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.

[8] T. Hafer and W. Thomas. Computation tree logic CTL* and path quantifiers in the monadic theory of the binary tree. In *Proc. 14th Int. Coll. Automata, Languages, and Programming (ICALP'87), Karlsruhe, FRG, July 1987*, volume 267 of *Lecture Notes in Computer Science*, pages 269–279. Springer, 1987.

[9] D. Harel. *Algorithmics: The Spirit of Computing*. Addison-Wesley, 2nd edition, 1992.

[10] D. Harel and D. Peleg. Process logic with regular formulas. *Theoretical Computer Science*, 38:307–322, 1985.

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

[12] 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.

[13] O. Kupferman, N. Piterman, and M. Y. Vardi. Extended temporal logic revisited. In *Proc. 12th Int. Conf. Concurrency Theory (CONCUR'2001), Aalborg, Denmark, Aug. 2001*, volume 2154 of *Lecture Notes in Computer Science*, pages 519–535. Springer, 2001.

[14] 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.

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

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

[17] 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.

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

[19] S. Miyano and T. Hayashi. Alternating finite automata on omega-words. *Theoretical Computer Science*, 32(3):321–330, 1984.

[20] R. Rosner and A. Pnueli. A choppy logic. In *Proc. 1st IEEE Symp. Logic in Computer Science (LICS'86), Cambridge, MA, USA, June 1986*, pages 306–313. IEEE Comp. Soc. Press, 1986.

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

[22] A. P. Sistla, M. Y. Vardi, and P. Wolper. The complementation problem for Büchi automata with applications to temporal logic. *Theoretical Computer Science*, 49(2–3):217–237, 1987.

[23] M. Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proc. 1st IEEE Symp. Logic in Computer Science (LICS'86), Cambridge, MA, USA, June 1986*, pages 332–344. IEEE Comp. Soc. Press, 1986.

[24] M. Y. Vardi. A temporal fixpoint calculus. In *Proc. 15th ACM Symp. Principles of Programming Languages (POPL'88), San Diego, CA, USA, Jan. 1988*, pages 250–259, 1988.

[25] M. Y. Vardi. Alternating automata and program verification. In *Computer Science Today. Recent Trends and Developments*, volume 1000 of *Lecture Notes in Computer Science*, pages 471–485. Springer, 1995.

[26] M. Y. Vardi. Reasoning about the past with two-way automata. In *Proc. 25th Int. Coll. Automata, Languages, and Programming (ICALP'98), Aalborg, Denmark, July 1998*, volume 1443 of *Lecture Notes in Computer Science*, pages 628–641. Springer, 1998.

[27] T. Wilke. Classifying discrete temporal properties. In *Proc. 16th Ann. Symp. Theoretical Aspects of Computer Science (STACS'99), Trier, Germany, Mar. 1999*, volume 1563 of *Lecture Notes in Computer Science*, pages 32–46. Springer, 1999.

Technical appendix

A. Proof of Lemma 4.2

First note that for any $i \geq i_0$, there exists some node x in any binary accepting run r with $|x| = i$ and $\mathrm{l}_r(x) = i_0$. We prove the following result:

$$\begin{array}{rcr} & \forall x \in r,\ |x| = i \text{ and } \mathrm{l}_r(x) = i_0 \text{ implies } \psi \in r(x) & (1) \\ \Leftrightarrow & \exists x \in r \text{ s.t. } |x| = i \text{ and } \mathrm{l}_r(x) = i_0 \text{ and } \psi \in r(x) & (2) \\ \Leftrightarrow & w^{i_0}, i - i_0 \models \psi & (3) \end{array}$$

Clearly the remark above implies that $(1) \Rightarrow (2)$. The proof that $(2) \Rightarrow (3)$ and $(3) \Rightarrow (1)$ is done by induction over ψ:

- $\psi = p \in AP$: if r is accepting, then x has two successors satisfying $\rho(x, w_i)$ and then $r(x)$ and w_i agree over AP.
- $\psi = \psi_1 \wedge \psi_2$: If $\psi \in r(x)$, then $\psi_1, \psi_2 \in r(x)$ (because $r(x)$ is an atom), and then by i.h. $w^{i_0}, i - i_0 \models \psi_j$ for $j = 1, 2$, and then $w^{i_0}, i - i_0 \models \psi_1 \wedge \psi_2$. Then $(2) \Rightarrow (3)$. Now suppose $w^{i_0}, i - i_0 \models \psi_1 \wedge \psi_2$, then $w^{i_0}, i - i_0 \models \psi_j$ for $j = 1, 2$, and by i.h. the labeling of any node x at depth i with $\mathrm{l}_r(x) = i_0$ contains ψ_j and then it contains $\psi_1 \wedge \psi_2$.
- $\psi = \neg\psi_1$: If $\psi \in r(x)$, then $\psi_1 \notin r(x)$ and then (by i.h.) $w^{i_0}, i - i_0 \not\models \psi_1$. The converse is similar.
- $\psi = \mathsf{X}\psi_1$: $((2) \Rightarrow (3))$ Let x_1 and x_2 be the successors of x in r over w. Given the definition of $\rho(x, \sigma)$, one and only one x_i is labeled by an initial atom. Assume $r(x_2)$ is initial. Then x_1 belongs to $\mathrm{Succ}(x, w_i)$, and then $\psi_1 \in r(x_1)$ which entails by i.h. $((2) \Rightarrow (3))$ that $w^{i_0}, i - i_0 + 1 \models \psi_1$ (x_1 is not initial and is at depth $i + 1$) and then $w^{i_0}, i - i_0 \models \mathsf{X}\psi_1$.
$((3) \Rightarrow (1))$ is similar: Assume $w^{i_0}, i - i_0 \models \mathsf{X}\psi_1$, then $w^{i_0}, i - i_0 + 1 \models \psi_1$. By i.h. any node x' of depth $i + 1$ with $\mathrm{l}_r(x') = i_0$ satifies $\psi_1 \in r(x')$. Given a node x at depth i with $\mathrm{l}_r(x) = i_0$, x has two successors in r, one of them is not initial, belongs to $\mathrm{Succ}(x', w_i)$, is at depth $i + 1$ and has a level equal to i_0, and then it is labeled by an atom containing ψ_1, and then $r(x) \ni \mathsf{X}\psi_1$ (by definition of Succ) and we have $(3) \Rightarrow (1)$.
- $\psi = \psi_1 \mathsf{U} \psi_2$: $((2) \Rightarrow (3))$ Assume $\psi \in r(x)$. Consider the infinite branch b from x which never visits initial atoms (except possibly x), such a branch exists due to the definition of ρ (at each step one successor is labeled by an non-initial atom). This branch has to visit infinitely many atoms satisfying ψ_2 or $\neg(\psi_1 \mathsf{U} \psi_2)$. Moreover the definition of Succ ensures that an atom labeled by $\psi_1 \mathsf{U} \psi_2$ is labeled by ψ_2 or its successor (along b) by $\psi_1 \mathsf{U} \psi_2$. This entails that a node on b is labeled by ψ_2 and every intermediary node is labeled by ψ_1 and then, (by i.h.) that $w^{i_0}, i - i_0 \models \psi_1 \mathsf{U} \psi_2$.
$((3) \Rightarrow (1))$ Assume $w^{i_0}, i - i_0 \models \psi_1 \mathsf{U} \psi_2$. Then there exists $j \geq 0$ s.t. $w^{i_0}, i - i_0 + j \models \psi_2$ and for any $j' < j$ we have $w^{i_0}, i - i_0 + j' \models \psi_1$. This entails that there exists a branch from x whose nodes are labeled with $\psi_1 \mathsf{U} \psi_2$.
- $\psi = \mathsf{X}^{-1}\psi_1$: $((2) \Rightarrow (3))$ Assume $\psi \in r(x)$. Then $\mathsf{X}^{-1}\top \in r(x)$ and there exists y s.t. x is a child of y and $r(x) \in \mathrm{Succ}(r(y), w_{i-1})$. Therefore $\psi_1 \in r(y)$ and by i.h. $w^{i_0}, i - i_0 - 1 \models \psi_1$, and then $w^{i_0}, i - i_0 \models \mathsf{X}^{-1}\psi_1$.
$((3) \Rightarrow (1))$ Assume $w^{i_0}, i - i_0 \models \mathsf{X}^{-1}\psi_1$. Then $i_0 < i$. Moreover $w^{i_0}, i - i_0 - 1 \models \psi_1$ and then by i.h. the labeling of any node x' of depth $i - 1$ with $\mathrm{l}_r(x') = i_0$ contains ψ_1. Now any node x of depth i with $\mathrm{l}_r(x) < i$ has a predecessor at depth $i - 1$ and $\mathrm{l}_r(x') = \mathrm{l}_r(x)$ and then is labeled by ψ_1, this entails that the labeling x contains $\mathsf{X}^{-1}\psi_1$.
- $\psi = \psi_1 \mathsf{S} \psi_2$: If x is initial, then $\psi \in r(x)$ entails $\psi_2 \in r(x)$. Otherwise due to the definition of Succ, there exists a node y along the branch from ε to x which is labeled by ψ_2 and all intermediary nodes between y and x are labeled by ψ_1, this gives $w^{i_0}, i - i_0 \models \psi_1 \mathsf{S} \psi_2$. The converse is similar.
- $\psi = \mathsf{N}\psi_1$: $((2) \Rightarrow (3))$ If x is labeled with an initial atom, then it also contains ψ_1 (by def. of an atom), and then by i.h. $w^{i_0}, 0 \models \psi_1$, and then $w^{i_0}, 0 \models \mathsf{N}\psi_1$. If x is not initial, then its "brother" (remember that the run is binary) is, and, by i.h., it is also labeled by ψ_1. This entails that $w^i, 0 \models \psi_1$ and then $w^{i_0}, i - i_0 \models \mathsf{N}\psi_1$ for any $i_0 = 0, \ldots, i$.
$((3) \Rightarrow (1))$ Assume $w^{i_0}, i - i_0 \models \mathsf{N}\psi_1$. Then $w^i, 0 \models \psi_1$, and by i.h. we have $\psi_1 \in r(x)$ if $|x| = i$ and $\mathrm{l}_r(x) = i$. Any node x' at depth i which is not labeled by an initial atom ($\mathrm{l}_r(x') < i$) has a predecessor which has a child labeled by an initial atom (because r is binary) similar to x and then x' is labeled by $\mathsf{N}\psi$ (by def. of Now).

B. Proof of Proposition 4.3

Only the $(\Rightarrow)$ direction needs be proved.

Given a word w and an *NLTL* formula φ, we define an $\mathrm{Atom}(\varphi)$-labeled binary run r^w_φ and we show that it is an accepting run of $\mathcal{A}_\varphi$ if $w, 0 \models \varphi$.

Definition of r^w_φ. Given $w \in \Sigma^\omega$, $\varphi \in$ *NLTL*, and $i, j \in \mathbb{N}$, we define the atom $\mathrm{C}^{w^j}_\varphi(i)$ as the set of $\mathrm{CL}(\varphi)$

formulae which hold for w^j, i, that is: $\mathrm{C}_\varphi^{w^j}(i) \stackrel{\text{def}}{=} \{\psi \in \mathrm{CL}(\varphi) \mid w^j, i \models \psi\}$

Now we define the Atom-labeled binary tree r_φ^w where every branch is infinite and which is labeled with a $\mathrm{C}_\varphi^{w^j}(k)$ as follows:

- $\mathrm{r}_\varphi^w(\varepsilon) \stackrel{\text{def}}{=} \mathrm{C}_\varphi^{w^0}(0)$,
- A node x of r_φ^w labeled by $\mathrm{C}_\varphi^{w^j}(k)$ has two children x_1 and x_2 such that:
  - x_1 is labeled by $\mathrm{C}_\varphi^{w^j}(k+1)$ and
  - x_2 is labeled by $\mathrm{C}_\varphi^{w^{j+k+1}}(0)$.

It is easy to verify [6] that any node x of r_φ^w of depth i is labeled by $\mathrm{C}_\varphi^{w^{i_0}}(i - i_0)$ with $i_0 = \mathrm{l}_{\mathrm{r}_\varphi^w}(x)$. Then we have:

Lemma B.1. *Given an NLTL formula φ and a word $w \in \Sigma^\omega$ such that $w, 0 \models \varphi$, the* Atom*-labeled tree r_φ^w is an accepting run of $\mathcal{A}_\varphi$ over w.*

Proof. $\mathrm{r}_\varphi^w(\varepsilon)$ is labeled by an initial atom containing φ, then $\mathrm{r}_\varphi^w(\varepsilon) \in S_0$. Now, given a node x of depth i labeled by $\mathrm{C}_\varphi^{w^{i_0}}(i - i_0)$ with two successors x_1 and x_2 as described above, the labeling of x_1 and x_2 satisfy the formula $\rho(\mathrm{C}_\varphi^{w^{i_0}}(i - i_0), w_i)$ because:

- x_1 is labeled $\mathrm{C}_\varphi^{w^{i_0}}(i+1)$ which clearly belongs to $\mathrm{Succ}(\mathrm{C}_\varphi^{w^{i_0}}(i - i_0))$ and
- x_2 is labeled by $\mathrm{C}_\varphi^{w^{i+1}}(0)$ which belongs to $\mathrm{Now}(\mathrm{C}_\varphi^{w^{i_0}}(i+1))$ by definition of C_φ^w.

It remains to show that every branch satisfies the acceptance condition $\mathcal{F}$:

- A branch which visits infinitely many (nodes labeled with) initial atoms is accepting because $\neg\mathsf{X}^{-1}\top$ belongs to every set F_i.
- Other branches visit a finite number of second transitions (leading to some x_2). Let x be a node on such a branch b with no initial atom in its descendants (in b). Assume x is at depth i and x is labeled with $\mathrm{C}_\varphi^{w^{i_0}}(i - i_0)$. The nodes along b are labeled by $\mathrm{C}_\varphi^{w^{i_0}}(i - i_0)$, $\mathrm{C}_\varphi^{w^{i_0}}(i - i_0 + 1)$, $\mathrm{C}_\varphi^{w^{i_0}}(i - i_0 + 2)$, etc. By definition of $\mathrm{C}_\varphi^{w^{i_0}}(i - i_0)$, for any $\psi_1 \mathsf{U} \psi_2 \in \mathrm{C}_\varphi^{w^{i_0}}(i - i_0)$, there exists $j \geq i - i_0$ s.t. $w^{i_0}, j \models \psi_2$ and for any $i - i_0 \leq j' < j$ we have $w^{i_0}, j' \models \psi_1$. Therefore there exists infinitely many nodes on b labeled by an atom containing ψ_2 whenever $\psi_1 \mathsf{U} \psi_2$ occurs in infinitely many atoms, otherwise there are infinitely many nodes labeled with $\neg(\psi_1 \mathsf{U} \psi_2)$. Therefore the branch b satifies every acceptance condition in F.

□

Therefore, if φ is initially satisfiable, there exsists an accepting run of $\mathcal{A}_\varphi$.

[6] it holds for the root, and the property is maintained for $\rho(x, w_i)$ successors.