An NP-complete fragment of LTL

Anca Muscholl[1] and Igor Walukiewicz[2]

[1] LIAFA, Université Paris 7
2, pl. Jussieu, case 7014
F-75251 Paris cedex 05, France
[2] LaBRI, Université Bordeaux-1
351, Cours de la Libération
F-33 405, Talence cedex, France

Abstract. A fragment of linear time temporal logic (LTL) is presented. It is proved that the satisfiability problem for this fragment is NP-complete. The fragment is larger than previously known NP-complete fragments. It is obtained by prohibiting the use of until operator and requiring to use only next operators indexed by a letter.

1 Introduction

Linear time temporal logic (LTL) is a well-studied and broadly used formalism for reasoning about events in time. It is equivalent to first-order logic over finite and infinite words [6]. The operators of the logic correspond to well-known semigroups which gives a starting point of the successful classification research [13]. LTL is used to formulate properties of finite or infinite words. Such a formalization permits to do model-checking – verify if the given model has the given property. It turns out that, for LTL and its fragments, in almost all cases the model-checking problem is equivalent to a satisfiability-checking problem. This is why the satisfiability problem for LTL and its fragments is so well-studied. It is well-known that the problem for whole LTL is PSPACE-complete [11]. It is known as well [11] that the fragment using only the "sometimes in the future" modality, denoted F, as well as the fragment using only the "next" modality, denoted X, have NP-complete satisfiability problems. Nevertheless, the fragment when both F and X are allowed is PSPACE-complete. This is a decidable fragment of LTL [1,13]. We show that restricting the next operator X to operators X_a ($a \in \Sigma$) that enforce the current letter to be a, we get a fragment with the satisfiability problem in NP.

Thus, this paper shows that the PSPACE-completeness of the $F + X$ fragment is in some sense an accident due to some syntactic conventions. A very common approach to formalization of LTL is to have propositions in the logic and to consider a model to be a sequence of valuations of propositions. Another approach is to consider models to be words over a given alphabet and to have next modalities indexed by the letters, i.e. $X_a\varphi$ says that the first letter of the model is a and after cutting a the rest of the model satisfies φ. Of course it is very easy to translate between the two conventions but the fragments that

look natural in one convention do not necessary do so in the other. In particular consider the next operator X. Having operators X_a we can express X as $X\varphi \equiv \bigvee_{a\in\Sigma} X_a\varphi$, where Σ is the alphabet. An important point about this translation is that it induces an exponential blow-up. We show that it is considering X as a primitive in the language that is the source of PSPACE-hardness. We prove that the fragment of LTL without until operator and using X_a operators instead of X is NP-complete.

Related work. We have mentioned already above the classic results on PSPACE-completeness of the full logic and NP-completeness of the fragments only with F and only with X, [11]. Matched with the PSPACE-completeness of $F + X$ fragment, these results were considered sharp and the later work has concentrated mostly on extensions of LTL [7,4,3,8]. Nevertheless the question about the fragment considered here was posed by the second author [12]. Recently the search of "easy" fragments of LTL has regained some interest [5,2,10]. The main motivation is to understand why the model-checkers (or the satisfiability-checkers) behave relatively well in practice despite the PSPACE lower bound. The fragments considered in recent papers put restrictions on the nesting of operators and on the number of propositions used [2].

2 Preliminaries

We will use Σ for a finite alphabet, the letters of which will be denoted by $a, b, c, \ldots$ As usual Σ^* denotes the set of finite and Σ^ω the set of infinite words over Σ. We use $u, v, w, \ldots$ to range over words.

Let $A \subseteq \Sigma^*$ be a finite set of words. The size of A is the sum of the lengths $|v|$ of all $v \in A$. We write $\Sigma^{\leq n}$ for the set of words of length $\leq n$.

Definition 1. *The set of subLTL formulas over an alphabet Σ is defined by the following grammar:*

$$\varphi ::= \mathsf{tt} \mid \mathsf{ff} \mid X_b\varphi \mid F\varphi \mid G\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2$$

where subscript b ranges over Σ and X_b, F and G are called "next", "finally" and "globally" modalities.

For an non-empty word $v = a_1 \cdots a_k$ we write for short $X_v\phi$ instead of the formula $X_{a_1} \ldots X_{a_k}\varphi$.

The models are infinite words $v \in \Sigma^\omega$. The semantic is standard so we recall just the most important clauses:

- $v \models X_a\varphi$ if v can be factorized as av' and $v' \models \varphi$;
- $v \models F\varphi$ if there is a factorization uw of v, where $u \in \Sigma^*$, such that $w \models \varphi$;
- $v \models G\varphi$ if for all factorizations uw of v with $u \in \Sigma^*$, we have $w \models \varphi$.

Observe that there is no negation in the syntax. This is because we can define the negation of a formula using the equivalence rules $\neg(F\varphi) = G(\neg\varphi)$

and $\neg(X_a\,\varphi) = X_a(\neg\varphi) \vee \bigvee_{b\neq a} X_b\mathtt{tt}$. Note that these rules increase the size of the formula by a linear factor only.

In this paper we prove:

Theorem 1. *The satisfiability problem for subLTL is NP-complete.*

Let us compare subLTL with linear time temporal logic with propositional constants, that we call PTL here. In PTL instead of an alphabet we have a set of propositional constants $Prop = \{P, Q, \dots\}$. The formulas are built from propositions and their negations using the modalities X, F and G. There is also an until operator but we do not need it for our discussion here. The models are infinite sequences of valuations of propositions. When interested in satisfiability of a given formula φ one can restrict to the set of propositions that appear in the formula, call it $Prop_\varphi$. This way a model can be coded as a word over the finite alphabet $\Delta = 2^{Prop_\varphi}$. Given this, the semantics is the best explained by the translation to LTL:

- P is translated to $\bigvee\{X_a\mathtt{tt} \mid a \in \Delta,\ P \text{ true in } a\}$ (recall that letters are valuations),
- $X\varphi = \bigvee_{a\in\Delta} X_a\varphi$

The rest of the clauses being identities.

Having definitions of both subLTL and PTL we can make the comparisons. First, observe that the fragment of PTL without X corresponds to the fragment of subLTL where after X_a we can put only $\mathtt{tt}$. Next, observe that the translation of $X\varphi$ induces an exponential blowup. For example a formula $X^n\mathtt{tt}$ (X n-times followed by $\mathtt{tt}$) is translated to a formula of exponential size. Finally, observe that subLTL can express more properties than PTL without X. A simple example is $G(X_a\mathtt{tt} \Rightarrow X_{ab}\mathtt{tt})$ which states that after each a there is b. This property is not expressible in PTL without X if we have more than two letters. Another interesting formula is $G(X_{ab}\mathtt{tt} \vee X_{ba}\mathtt{tt})$. This formula has only the words $(ab)^\omega$ and $(ba)^\omega$ as models. This indicates that constructing a model for a subLTL formula may require a bit of combinatorics on words as the phenomena of interplay between different prefixes start to occur.

3 The lower-bound

Showing NP-hardness of the satisfiability problem for subLTL is quite straightforward.

We reduce SAT. Given a propositional formula α over variables $x_1, \dots, x_n$ we consider models over the letters $b, a_1, \dots, a_n$ were b will be used to fill the "empty spaces". A valuation of the variables $x_1, \dots, x_n$ will be encoded by a word in such a way that x_i is true iff a_i occurs in the word. Let φ_α be a formula obtained by replacing each occurrence of x_i in α by $FX_{a_i}\mathtt{tt}$. Then there is a valuation satisfying α iff there is a word which is a model for φ_α.

In this reduction the alphabet is not fixed. Nevertheless it is quite straightforward to modify the reduction so that it works also for a two letter alphabet. For example, one can code each letter a_i as a word ba^ib. We omit the details.

4 The upper-bound

To show that the satisfiability problem for subLTL is in NP we will prove the small model property. The algorithm will be then to guess the model, of the form uv^ω for $u, v \in \Sigma^*$ and to check if the formula holds. This latter task can be done in polynomial time [11]. As a digression let us mention that the precise complexity of this problem is not known [9].

Hence, our goal in this section is the following theorem:

Theorem 2. *Every satisfiable formula φ of subLTL has a model of the form uv^ω with $|u| + |v|$ polynomial in the size of φ.*

The proof will be split into two subsections. In the first we will consider periodic words, i.e., ones of the form v^ω. We will show that if φ has a model v^ω then there is short word w such that w^ω is also a model of φ. In the second subsection we consider the case of ultimately periodic words, i.e., of the form uv^ω and show how to shorten u. Putting the two together we will obtain a small model for any satisfiable formula.

4.1 Periodic words

We will first characterize the models of a subLTL formula that are periodic infinite words, i.e., words of the form v^ω for some $v \in \Sigma^*$.

Let $\mathrm{Swords}(w)$ be the set of finite factors of w.

Definition 2. *For $A, B \subseteq \Sigma^*$ and $p \in \Sigma^*$ we say that $w \in \Sigma^* \cup \Sigma^\omega$ is an (A, B, p)-word if*

- *p is a prefix of w,*
- *$A \subseteq Swords(w)$,*
- *$B \cap Swords(w) = \emptyset$.*

For the proof of the proposition below it is important to note that for any word $v \in \Sigma^*$ and any factorization $v = xy$ we have that $\mathrm{Swords}(v^\omega) = \mathrm{Swords}((yx)^\omega)$.

Proposition 1. *Let ϕ be a subLTL formula of size n. Then there exists a set $\mathcal{T}(\phi)$ of triples (A, B, p), where $A, B \subseteq \Sigma^{\leq n}$ are of polynomial size in n and $p \in \Sigma^{\leq n}$, such that for any word $v \in \Sigma^*$:*

$$v^\omega \models \phi \quad \textit{iff} \quad v^\omega \textit{ is an } (A, B, p)\textit{-word for some } (A, B, p) \in \mathcal{T}(\phi).$$

Proof. We show the assertion by induction on the given formula ϕ.

1. We have $\mathcal{T}(\mathsf{tt}) = \{(\emptyset, \emptyset, \lambda)\}$ and $\mathcal{T}(\mathsf{ff}) = \{(\emptyset, \Sigma, \lambda)\}$.
2. Suppose $\phi = \phi_1 \wedge \phi_2$. We define $\mathcal{T}(\phi)$ as the set of triples (A, B, p) constructed as follows. For every two triples $(A_1, B_1, p_1) \in \mathcal{T}(\phi_1)$ and $(A_2, B_2, p_2) \in \mathcal{T}(\phi_2)$ with $p_1 \leq p_2$ ($p_2 \leq p_1$ respectively) we let $A = A_1 \cup A_2$, $B = B_1 \cup B_2$ and $p = p_2$ ($p = p_1$ respectively). It is easy to check that v^ω is a (A_i, B_i, p_i)-word for $i = 1, 2$ if and only if it is a (A, B, p)-word.

3. Suppose $\phi = X_a\psi$. We define $\mathcal{T}(\phi) = \{(A, B, ap) \mid (A, B, p) \in \mathcal{T}(\psi)\}$. We have $v^\omega \models \phi$ if and only if $v = aw$ and $(wa)^\omega \models \psi$. By induction, this happens if and only if $(wa)^\omega$ is an (A, B, p)-word for some $(A, B, p) \in \mathcal{T}(\psi)$. But this is the case if and only if $(aw)^\omega$ is an (A, B, ap)-word.
4. Suppose $\phi = F\psi$. We define $\mathcal{T}(\phi) = \{(A \cup \{p\}, B, \lambda) \mid (A, B, p) \in \mathcal{T}(\psi)\}$. We have $v^\omega \models F\psi$ if and only if there exists some factorization $v = wx$ with $(xw)^\omega \models \psi$. By induction hypothesis this is equivalent to $(xw)^\omega$ being an (A, B, p)-word for some triple $(A, B, p) \in \mathcal{T}(\psi)$. It is now easy to see that v^ω is an $(A \cup \{p\}, B, \lambda)$-word iff there is a factorization xw of v with $(wx)^\omega$ being an (A, B, p)-word.
5. Let $\phi = G\psi$. For each subset $\{(A_0, B_0, p_0), \ldots, (A_k, B_k, p_k)\} \subseteq \mathcal{T}(\psi)$ with a distinguished element (A_0, B_0, p_0) we add the tuple (A, B, p_0) to $\mathcal{T}(\phi)$ where

$$A = \bigcup_{i=0,\ldots,k} A_i, \qquad B = \bigcup_{i=0,\ldots,k} B_i \cup Y$$

and Y is the set of minimal words that are neither prefixes nor contain as a prefix any of the words $p_0, \ldots, p_k$. It is easy to see that Y is of the size polynomial in n. A word belongs to Y if it is of the form va with v a prefix of one of the words $p_0, \ldots, p_n$ and va neither a prefix of any of these words nor containing any of them.
Suppose that $v^\omega \models G\psi$. For every factorization xw of v we know, by the induction hypothesis, that $(wx)^\omega$ is a (A_w, B_w, p_w)-word for some (A_w, B_w, p_w) in $\mathcal{T}(\psi)$. Let $(A, B, p_v) \in \mathcal{T}(\phi)$ be the triple constructed as above from the set $\{(A_w, B_w, p_w) \mid w \text{ suffix of } v\}$. A direct verification shows that v^ω is a (A, B, p_v)-word. For example, let us show that $\text{Swords}(v^\omega) \cap B = \emptyset$. Directly from the definition we have that $\text{Swords}(v^\omega) \cap B_w$ for every w a suffix of v. For the set Y defined as above we have $\text{Swords}(v^\omega) \cap Y = \emptyset$ because all suffixes of v^ω have some p_w as a prefix.
For the opposite direction suppose that v is an (A, B, p)-word constructed from some set $\{(A_0, B_0, p_0), \ldots, (A_k, B_k, p_k)\} \subseteq \mathcal{T}(\psi)$. Take any factorization xw of v. We want to show that $(wx)^\omega \models \psi$. Because of the set Y, as defined above, the word $(wx)^\omega$ has some p_i as a prefix. From the definition we know that $A_i \subseteq A$ and $B_i \subseteq B$. Hence $(wx)^\omega$ is a (A_i, B_i, p_i)-word and consequently $(wx)^\omega \models \psi$.

□

Example 1. Consider the formula $\phi = G\psi$, where $\psi = X_{ab}\mathsf{tt} \vee X_{ba}\mathsf{tt}$ and $\Sigma = \{a, b\}$. We have $\mathcal{T}(\psi) = \{(\emptyset, \emptyset, ab), (\emptyset, \emptyset, ba)\}$. The construction above yields $\mathcal{T}(\phi) = \{(\emptyset, \{b, aa\}, ab), (\emptyset, \{a, bb\}, ba), (\emptyset, \{aa, bb\}, ab), (\emptyset, \{aa, bb\}, ba)\}$. Clearly, only for the last two triples of $\mathcal{T}(\phi)$ there can be a solution.

The next two lemmas show that finite (periodic, resp.) (A, B, p)-words can be chosen of polynomial length.

Lemma 1. *Let $A, B \subseteq \Sigma^*$ and $p \in \Sigma^*$. If there is a finite (A, B, p)-word then there is one of size polynomial in the sizes of A, B and p.*

Proof. We construct a (deterministic) finite automaton $\mathcal{A}$ accepting (A, B, p)-words. Then we show that it accepts a short word.

As a preparation observe that we may assume that no word in A is a factor of some other word in A; if it is the case then we simply delete the smaller one. Similarly for B but here we can delete the bigger one.

The states of $\mathcal{A}$ will be triples (u, v, r) where u is a prefix of a word in A, v is a prefix of a word in B and r is a prefix of p. Intuitively such a state will say that u and v are suffixes of the word being read and they are the longest possible suffixes for the words in A and B respectively. The last component r is used for testing that the word starts with p. The initial state will be $(\lambda, \lambda, \lambda)$.

The transitions of $\mathcal{A}$ are deterministic. We have

$$(u, v, r) \xrightarrow{a} (u', v', r')$$

when

- either $u' = ua$, or if ua is not a prefix of a word in A then u' is the longest suffix of ua that is a prefix of word from A.
- for v' we have exactly the same rule but with respect to B.
- either $r = p = r'$, or $r' = ra$ if ra is a prefix of p.

A state (u, v, r) is *rejecting* if $v \in B$. It is a u-state if its first component is u.

Our first claim is that a word w is an (A, B, p)-word iff $\mathcal{A}$ has a run on this word that does not visit a rejecting state, passes through a u-state for every $u \in A$ and ends in a state where the last component is p (called a p-state). This follows from the observation that if there is a B-factor v in w then after reading the last letter of v the automaton $\mathcal{A}$ enters in the rejecting state. If $u \in A$ is a factor of w, then after reading u the state of $\mathcal{A}$ is a u-state.

It remains to see that there is a short (A, B, p)-word if there is one at all. Consider an (A, B, p)-word u and an accepting run $\rho = (s_0, s_1, \ldots, s_m)$ of $\mathcal{A}$ on $u = a_1 \cdots a_m$. Thus, s_m is p-state and for each $u \in A$ there is some position j such that s_j is a u-state. Let us fix for each $u \in A$ such a position $j(u)$ and let $J = \{j(u) \mid u \in A\}$. We can delete now any loop contained between two consecutive positions in J, and the run obtained is still accepting. The length of the run is at most $O(|A||\mathcal{A}|)$, hence polynomial in $|A|, |B|, |p|$.

□

Lemma 2. *If there is a periodic (A, B, p)-word then there is one of the form s^ω with $|s|$ polynomial in the sizes of A, B and p.*

Proof. The construction is as in the previous lemma but now we need to start the automaton in some state of the form (u, v, λ) and to require that it reaches the state (u, v, p).

Suppose that we have a run from (u, v, λ) to (u, v, p) for some u, v, and let s be the word defining this run. Then s defines also a run from (u, v, p) back to itself, thus s^ω is the desired periodic (A, B, p)-word. By the same argument as in the lemma before we can also see that there is always a run of polynomial

length, if any. For the other direction, suppose that s^ω is a periodic (A, B, p)-word. We can consider the run of $\mathcal{A}$ on this word starting in $(\lambda, \lambda, \lambda)$. Since s^ω is an (A, B, p)-word, this run never blocks. Hence, there must be two indices $i < j$ such that after reading s^i and s^j the automaton is in the same state, say (u, v, p). But then, there is a run of automaton $\mathcal{A}$ from (u, v, λ) to (u, v, p) on s^{j-i}. □

4.2 Ultimately periodic words

We turn now to the case where the model of ϕ is ultimately periodic, i.e., of the form uv^ω for some $u, v \in \Sigma^*$. For the rest of the section, n denotes the length of the given formula φ.

Lemma 3. *Any subLTL formula ϕ using only the operators X_a ($a \in \Sigma$) is equivalent to a disjunction of the form $\bigvee_{v \in V} X_v \mathsf{tt}$, for some set $V \subseteq \Sigma^{\leq |\phi|}$ of at most $|\phi|$ words.*

Proof. We show the assertion by induction on the given formula ϕ. For $\phi = X_a \psi$ we suppose that ψ is equivalent to $\bigvee_{v \in V} X_v \mathsf{tt}$, hence ϕ is equivalent to $\bigvee_{v \in V'} X_v \mathsf{tt}$ for $V' = aV$. Let now ϕ_i be equivalent to $\vee_{v \in V_i} X_v \mathsf{tt}$ for $i = 1, 2$. Then $\phi_1 \wedge \phi_2$ is equivalent to $\bigvee_{v \in V} X_v \mathsf{tt}$, for V defined as follows: a word $v \in V_i$ belongs to V if and only if there exists $v' \in V_j$, $j \neq i$, such that $v' \leq v$. □

An *F-formula* is a formula that begins with F. Similarly for X and G-*formulas*. The set $\mathrm{El}(\phi)$ of elementary subformulas of ϕ is the set of those subformulas of φ that are either F- or G-formulas.

For any word we write $w[i, j]$ for the factor $a_i \dots a_j$. For an infinite word $w = a_1 a_2 \dots$ we write $w[j, \infty]$ for the suffix $a_j a_{j+1} \dots$ We use $w(i)$ to denote a_i, the i-th letter a_i.

For the rest of the section we fix a model uv^ω of a formula φ. With each position $i \leq |u|$ in the word uv^ω we associate the set of subformulas:

$$S_i = \{\psi \in \mathrm{El}(\varphi) \mid u[i, |u|]\, v^\omega \models \psi\}$$

Remark 1. If a formula $G\alpha$ is in S_i then it is in all S_j for $j \geq i$. Analogously if $F\alpha \in S_i$ then $F\alpha \in S_j$ for all $j \leq i$.

A position $i \leq |u|$ is called *important* if there is a formula $F\alpha$ in $S_i \setminus S_{i+1}$ or there is a formula $G\alpha$ in $S_{i+1} \setminus S_i$. Let VIP be the set of important positions in u. Clearly the number of important positions is bounded by $n = |\varphi|$.

We will show how to reduce distances between consecutive important positions, in order to obtain a short word u. From now on we fix two consecutive important positions $i, j \in \mathrm{VIP}$. This means that $S_{i+1} = \dots = S_j$ contain the same F- and G-subformulas and $S_i \neq S_{i+1}$.

For a subformula ψ of φ let $\widehat{\psi}$ be a formula obtained by substituting tt for every F- or G-subformula of ψ appearing in S_j and ff for all other F or G subformulas. By Lemma 3, for every subformula ψ there exists a polynomial-size

set of words $V_\psi \subseteq \Sigma^{\leq n}$ such that $\widehat{\psi}$ is $\bigvee_{v \in V_\psi} X_v \mathtt{tt}$ (in consequence, if $V_\psi = \emptyset$ then $\widehat{\psi} = \mathtt{ff}$ and if $V_\psi = \{\lambda\}$ then $\widehat{\psi} = \mathtt{tt}$).

Example 2. $X_a \widehat{G(X_b \mathtt{tt})}$ is $X_a \mathtt{tt}$ or $X_a \mathtt{ff}$ depending on $G(X_b \mathtt{tt})$ being in S_j or not. Moreover $G(\widehat{X_a G(X_b \mathtt{tt})})$ is just $\mathtt{tt}$ or $\mathtt{ff}$. (Both hats range over whole formula).

Our goal is to replace $u[i+1, j]$ in uv^ω by a short word so that the result is still a model of φ. In order to do this we will use (Y, p, s)-words, that we define in the following. Let $Y \subseteq \Sigma^*$ and $p, s \in \Sigma^*$. A finite word w is a (Y, p, s)-word, if it starts with p, finishes with s and for all positions $1 \leq k \leq |w| - |s|$, the suffix $w[k, |w|]$ starts with a word from Y.

Lemma 4. *If there exists some (Y, p, s) word, then there exists some of length polynomial in the sizes of Y, p, s.*

Proof. Consider some (Y, p, s)-word w and a position $k \leq |w| - |s|$. By definition, there exists some $l > k$ such that $w[k, l] \in Y$. With k we associate the position $r(k)$ defined by $r(k) = \max\{l' \geq l \mid \exists k' \leq k : w[k', l'] \in Y\}$. That is, $r(k)$ is the rightmost end of a word in Y that begins at the left of k. By definition, $w[l+1, r(k)]$ is a suffix (possibly empty) of a word in Y. Thus, there are at most $|Y|^2$ different words $w[k, r(k)]$.

Suppose now that $|p| < k < k' \leq |w| - |s|$ are such that $w[k, r(k)] = w[k', r(k')]$. Obviously, the word $w[1, k]w[k'+1, |w|]$ obtained by cutting out $w[k+1, k']$ is still a (Y, p, s)-word. Thus, by the above remark we know that there exists a (Y, p, s)-word of length at most $|Y|^2 + |ps|$. □

For each $G\alpha \in S_j$ let V_α be the set of words obtained by Lemma 3 such that $\widehat{\alpha} = \bigvee_{v \in V_\alpha} X_v \mathtt{tt}$. Applying again Lemma 3 we obtain a set Y such that $\bigwedge_{G\alpha \in S_j} \widehat{\alpha} = \bigvee_{v \in Y} X_v \mathtt{tt}$. Note that Y is of polynomial size and contains only words of length at most n, since $Y \subseteq \bigcup_{G\alpha \in S_j} V_\alpha$. Moreover, let $p = u[i+1, i+n]$, $s = u[j-n+1, j]$ be the prefix and suffix, resp., of length n of $u[i, j]$.

The next lemma follows immediately from the definition of Y:

Lemma 5. *Let $i, j \in VIP$ be consecutive important positions with $j - i > n$, and let Y, p, s be defined as above. Then the word $u[i+1, j]$ is a (Y, p, s)-word. Moreover, each (Y, p, s)-word w starts with $u[i+1, i+n]$, finishes with $u[j-n+1, j]$ and for all $k \leq |w| - n$ the word $w[k, |w|]$ has a word from V_α as a prefix for all subformulas $G\alpha \in S_j$.*

Our goal is to show that $wu[j+1, |u|]v^\omega \models S_{i+1}$ for any (Y, p, s)-word w, as this will imply $u[1, i]wu[j+1, |u|]v^\omega \models \varphi$. To do this we will need a definition and several lemmas. Consider a formula α and let

$$\gamma_\alpha = \bigwedge \{\delta \in S_{i+1} \mid \delta \text{ is a } F\text{- or } G\text{-subformula of } \alpha\}$$

Hence γ_α is the conjunction of all F- and G-formulas that appear in $S_{i+1} = S_j$ and that are subformulas of α. By Lemma 3 the formula $\widehat{\alpha}$ is equivalent to

$\bigvee_{r\in R} X_r\mathsf{tt}$. We define $\widetilde{\alpha}$ to be the formula $\bigvee_{r\in R}\bigwedge_{s\leq r} X_s\gamma_\alpha$. The definition of $\widetilde{\alpha}$ is important because it gives an underapproximation of α that is easy to work with.

Lemma 6. *If α is a subformula of the initial formula then $\widetilde{\alpha}\Rightarrow\alpha$ holds.*

Proof. By induction on α:

If α is an F- or G-formula then $\widehat{\alpha}$ is either tt or ff and $\widetilde{\alpha}$ is either γ_α or ff, respectively. We have that $\widetilde{\alpha}\Rightarrow\alpha$ in this case.

If $\alpha = X_a\beta$ then $\gamma_\alpha\Rightarrow\gamma_\beta$. Let $\widehat{\beta} = \bigvee_{r\in R} X_r\mathsf{tt}$, then $\widetilde{\beta} = \bigvee_{r\in R}\bigwedge_{s\leq r} X_s\gamma_\beta$. We have that $\widetilde{\alpha} = \bigvee_{ar\in aR}\bigwedge_{s\leq ar} X_s\gamma_\alpha = \bigvee_{r\in R}(\gamma_\alpha\wedge\bigwedge_{s\leq r} X_aX_s\gamma_\alpha) = \gamma_\alpha\wedge X_a(\bigvee_{r\in R}\bigwedge_{s\leq r} X_s\gamma_\alpha)\Rightarrow X_a(\bigvee_{r\in R}\bigwedge_{s\leq r} X_s\gamma_\beta) = X_a\widetilde{\beta}\Rightarrow X_a\beta=\alpha$.

For $\alpha=\beta_0\vee\beta_1$ the argument is straightforward. It remains to consider the case when α is of the form $\beta_0\wedge\beta_1$. We have that $\widehat{\beta}_0$ is of the form $\bigvee_{r\in R_0} X_r\mathsf{tt}$ and $\widehat{\beta}_1$ is $\bigvee_{r\in R_1} X_r\mathsf{tt}$. Now by construction $\widehat{\beta_0\wedge\beta_1}$ is $\bigvee_{s\in S} X_s\mathsf{tt}$ where $s\in S$ if there is $i\in\{0,1\}$ with $s\in R_i$ and some prefix of s in R_{1-i}. We have by definition that $\widetilde{\beta_0\wedge\beta_1}$ is $\bigvee_{s\in S}\bigwedge_{w\leq s} X_w\gamma_\alpha$. It is easy to check that this implies $\bigvee_{s\in R_i}\bigwedge_{w\leq s} X_w\gamma_\alpha$ for $i\in\{0,1\}$, hence also $\widetilde{\beta_0}\wedge\widetilde{\beta_1}$. □

We have now all ingredients to show that $wu[j+1,|u|]v^\omega\models S_{i+1}$ for any (Y,p,s)-word w. To shorten the notation we will write z for the suffix $u[j+1,|u|]v^\omega$. The proof goes through two lemmas because we need to consider G-formulas separately.

Lemma 7. *Let w be a (Y,p,s)-word. For every G-formula $G\alpha\in S_{i+1}$ and every non-empty suffix w' of w we have $w'z\models G\alpha$.*

Proof. The proof is by induction on the size of $G\alpha$. Take a formula $G\alpha\in S_{i+1}$ and any suffix w' of w. We want to show that $w'z\models\alpha$. Since $z\models G\alpha$, this suffices for showing that $w'z\models G\alpha$.

If $|w'|\leq n$ then w' is a suffix of $u[j-n+1,j]$ and we have $w'z\models G\alpha$ by definition of S_{i+1}. Hence also $w'z\models\alpha$.

If $|w'|>n$ then we know by Lemma 5 that w' starts with a word from V_α, say $r\in V_\alpha$. We will show that $w'z\models\widetilde{\alpha}$. Consider now the conjunct $\bigwedge_{s\leq r} X_s\gamma_\alpha$ of $\widetilde{\alpha}$, where γ_α is, as before, the conjunction of all F and G-subformulas of α from S_{i+1}. We know that $|w'|>n$ and $|r|\leq n$, hence we can use the induction hypothesis and we obtain that $w'z\models\bigwedge_{s\leq r} X_s\gamma_\alpha$. But then we have $w'z\models\widetilde{\alpha}$ which implies $w'z\models\alpha$ by Lemma 6. □

Lemma 8. *If w is a (Y,p,s)-word then $wz\models S_{i+1}$.*

Proof. For formulas $F\alpha\in S_{i+1}$ we know that $u(j)z\models S_j = S_{i+1}$ and we are done as w ends with the letter $u(j)$. The case of G-formulas follows from the previous lemma.

It remains to consider an X-formula α. The first observation is that $wz\models\widehat{\alpha}$. This is because $u[i+1,j]z\models\alpha$, the size of α is not bigger than n, and w starts

with $u[i+1, i+n]$. Now, by the same reasoning as in the previous lemma we get that $wz \models \widetilde{\alpha}$. Finally, by Lemma 6 we have $wz \models \alpha$. □

We obtain:

Proposition 2. *If $uv^\omega \models \varphi$ and $i, j \in VIP$ are successive important positions then there is a word w of size polynomial in $n = |\varphi|$ such that $u[1, i]\, w\, u[j+1, |u|]v^\omega \models \varphi$.*

Proof. From Lemma 8 we know that $wz \models S_{i+1}$. By induction on $k = i, \ldots, 1$ it is easy to see that $u[k, i]\, w\, [j+1, |u|]v^\omega \models S_k$. Lemma 4 gives the bound on the length of w. □

Using Proposition 2 repetitively we can shorten the size of u. From the previous subsection, Proposition 1 and Lemma 2, we know that we can shorten v. This proves the small model theorem, Theorem 2.

5 Conclusions

We have show the small model property for subLTL which implies that the satisfiability problem for the logic is NP-complete. This indicates that the temporal logic formalism based on word models is more subtle than the one based on propositions and on sequences of valuations. This also indicates that the X operator of LTL may be a source of complexity. On the other hand it seems that X_a operators are easier algorithmically, but the proofs become much more involved because some combinatorics on words becomes necessary. As further work we would like to investigate global trace logics with X_a instead of X. The hope being to have a reasonable such logic with the complexity lower than EXPSPACE. The other question is the complexity of the model-checking problem for subLTL with respect to a single path uv^ω. This question has been asked for general LTL in [9]. For general LTL it can be solved in polynomial time, but no good lower bound is known.

References

1. J. Cohen, D. Perrin, and J.-E. Pin. On the expressive power of temporal logic. *Journal of Computer and System Sciences*, 46(3):271–294, 1993.
2. S. Demri and Ph. Schnoebelen. The complexity of propositional linear temporal logics in simple cases. *Information and Computation*, 174(1):84–103, 2002.
3. C. Dixon, M. Fisher, and M. Reynolds. Execution and proof in a horn-clause temporal logic. In H. Barringer, M. Fisher, D. Gabbay, and G. Gough, editors, *Advances in Temporal Logic*, volume 16 of *Applied Logic Series*, pages 413–433. Kluwer Academic, 2000.
4. E. A. Emerson and C. Lei. Modalities for model-checking: Branching-time logic strikes back. *Science of Computer Programming*, 8(3):275–306, 1987.
5. K. Etessami, M. Vardi, and Th. Wilke. First-order logic with two variables and unary temporal logic. In *LICS'97 - 12th Annual IEEE Symposium on Logic in Computer Science, Warsaw, Poland*, pages 228–235, 1997.

6. J. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, 1968.
7. O. Lichtenstein and A. Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *POPL'85 - 12th Annual ACM Symposium on Principles of Programming Languages, New Orleans, Louisiana*, pages 97–107, 1985.
8. N. Markey. Past is for free: On the complexity of verifying linear temporal properties with past. *Acta Informatica*, 40(6-7):431–458, 2004.
9. N. Markey and Ph. Schnoebelen. Model checking a path. In *CONCUR'03 - Concurrency Theory, 14th International Conference, Marseille, France*, number 2761 in Lecture Notes in Computer Science, pages 248–262. Springer, 2003.
10. Ph. Schnoebelen. The complexity of temporal logic model checking. *Advances in Modal Logic*, 4:437–459, 2003.
11. A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logic. *Journal of the ACM*, 32(3):733–749, 1985.
12. I. Walukiewicz. Difficult configurations – on the complexity of LTrL. In *ICALP'98 - Automata, Languages and Programming, 25th International Colloquium, Aalborg (Denmark)*, number 1443 in Lecture Notes in Computer Science, pages 140–151, 1998.
13. Th. Wilke. Classifying discrete temporal properties. In *STACS'99, 16th Annual Symposium on Theoretical Aspects of Computer Science, Trier (Germany)*, number 1563 in Lecture Notes in Computer Science, pages 32–46, 1999.