Effective Characterizations of Simple Fragments of Temporal Logic Using Prophetic Automata

Sebastian Preugschat and Thomas Wilke

Christian-Albrechts-Universität zu Kiel
{preugschat,wilke}@ti.informatik.uni-kiel.de

Abstract. We present a framework for obtaining effective characterizations of simple fragments of future temporal logic (LTL) with the natural numbers as time domain. The framework is based on prophetic automata (also known as complete unambiguous Büchi automata), which enjoy strong structural properties, in particular, they separate the "finitary fraction" of a regular language of infinite words from its "infinitary fraction" in a natural fashion. Within our framework, we provide characterizations of all natural fragments of temporal logic, where, in some cases, no effective characterization had been known previously, and give lower and upper bounds for their computational complexity.

1 Introduction

Ever since propositional linear-time temporal logic (LTL) was introduced into computer science by Amir Pnueli in [18] it has been a major object of research. The particular line of research we are following here is motivated by the question how each individual temporal operator contributes to the expressive power of LTL. More precisely, our objective is to devise decision procedures that determine whether a given LTL property can be expressed using a given subset of the set of all temporal operators, for instance, the subset that includes "next" and "eventually", but not "until".

As every LTL formula interpreted in the natural numbers (the common time domain) defines a regular language of infinite words (ω-language), the aforementioned question can be viewed as part of a larger program: classifying regular ω-languages, that is, finding effective characterizations of subclasses of the class of all regular ω-languages. Over the years, many results have been established and specific tools have been developed in this program, the most fundamental result being the one that says that a regular ω-language is star-free or, equivalently, expressible in first-order logic or in LTL if, and only if, its syntactic semigroup is aperiodic [10,23,16].

The previous result is a perfect analogue of the same result for regular languages of finite words, that is, of the classical theorems by Schützenberger [19], McNaughton and Papert [13], and Kamp [10]. In general, the situation with infinite words is more complicated as with finite words; a good example for this is given in [5], where, for instance, tools from topology and algebra are used to settle characterization problems for ω-languages.

The first characterization of a fragment of LTL over finite linear orderings was given in [4], another one followed in [7], both following a simple and straightforward approach: to determine whether a formula is equivalent to a formula in a certain fragment, one computes the minimum reverse DFA for the corresponding regular language and verifies certain structural properties of this automaton, more precisely, one checks whether certain "forbidden patterns" do not occur. The first characterization for infinite words (concerning stutter-invariant temporal properties) [15] used sequential relations on ω-words; the second (concerning the nesting depth in the until/since operator) [22] used heavy algebraic machinery and did not shed any light on the computational complexity of the decision procedures involved.

In this paper, we describe a general, conceptually simple paradigm for characterizing fragments of LTL when interpreted in the natural numbers, combining ideas from [4,7] for finite words with the work by Carton and Michel on unambiguous Büchi automata [2,3]. The approach works roughly as follows. To determine whether a given formula is equivalent to a formula in a given fragment, convert the formula into what is called a "prophetic automaton" in [17], check that the automaton, when viewed as an automaton on finite words, satisfies certain properties, and check that languages of finite words derived from the accepting loops ("loop languages") satisfy certain other properties. In other words, we reduce the original problem for ω-languages to problems for languages of finite words. We show that the approach works for all reasonable fragments of future LTL and yields optimal upper bounds for the complexity of the corresponding decision procedures for all but one fragment.

A note on terminology. As just explained, we work with a variant (for details, see below) of the automaton model introduced by Carton and Michel in [2,3] and named CUBA model (**C**omplete **U**nambiguous **B**üchi **A**utomata). In [17], Pin uses "prophetic automata" to refer to CUBA's. We suggest to henceforth refer to these automata as "Carton–Michel automata" (CMA).

Structure of this extended abstract. In Section 2, we provide background on the topics relevant to this paper, in particular, CMA's and propositional linear-time temporal logic. In Section 3, we explain that a translation from temporal logic into CMA's is straightforward. In Section 4, we present our characterizations. In Section 5, we give a proof of the correctness of one of our characterizations, and in Section 6, we explain how our characterizations can be used effectively and deal with complexity issues. We conclude with open problems.

2   Basic Notation and Background

2.1   Reverse Deterministic Büchi Automata

A *Büchi automaton with a reverse deterministic transition function* is a tuple $(A, Q, I, \cdot, F)$ where A is a finite set of symbols, Q is a finite set of states, $I \subseteq Q$ is a set of initial states, $\cdot$ is a reverse transition function $A \times Q \to Q$, and $F \subseteq Q$ is
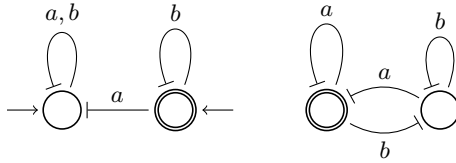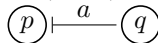
Fig. 1. CMA which recognizes $(a + b)^* b^\omega$

a set of final states. As usual, the transition function is extended to finite words by setting $\epsilon \cdot q = q$ and $au \cdot q = a \cdot (u \cdot q)$ for $q \in Q$, $a \in A$, and $u \in A^*$. For ease in notation, we write uq for $u \cdot q$ when the transition function $\cdot$ is clear from the context.

A run of an automaton as above on an ω-word u over A is an ω-word r over Q satisfying the condition $r(i) = u(i)r(i + 1)$ for every $i < \omega$. Such a run is called *initial* if $r(0) \in I$; it is *final* if there exist infinitely many i such that $r(i) \in F$; it is *accepting* if it is initial and final. The language of ω-words recognized by such an automaton, denoted $\mathrm{L}(\boldsymbol{A})$ when $\boldsymbol{A}$ stands for the automaton, is the set of ω-words for which there exists an accepting run.

An automaton as above is called a *Carton–Michel automaton (CMA)* if for every ω-word over A there is exactly one final run. Such an automaton is *trim*, if every state occurs in some final run.— The original definition of Carton and Michel in [2,3] is slightly different, but for trim automata—the interesting ones— the definitions coincide.

As an example, consider the automaton depicted in Figure 1, which is a CMA for the language denoted by $(a + b)^* b^\omega$. Note that we depict $p = aq$ as

$$(p) \vdash\!\!\!\frac{a}{}\!\!\!- (q)$$

An initial state has an incoming edge $\longrightarrow$, a final state has a double circle $\bigcirc\!\!\!\bigcirc$.

The fundamental result obtained by Carton and Michel is the following.

Theorem 1 (Carton and Michel [2,3]). *Every regular ω-language is recognized by some CMA. More precisely, every Büchi automaton with n states can be transformed into an equivalent CMA with at most $(12n)^n$ states.*

Let $\boldsymbol{A}$ be a CMA over an alphabet A and $u \in A^+$. The word u is a *loop at q* if $q = uq$ and there exist $v, w \in A^*$ satisfying $vw = u$ and $wq \in F$. The set of loops at q is denoted $S(q)$. What Carton and Michel prove about loops is:

Lemma 1 (Carton and Michel [2,3]). *Let $\boldsymbol{A}$ be a CMA over some alphabet A. Then, for every $u \in A^+$, there is exactly one state q, denoted $u\natural$ and called anchor of u, such that u is a loop at q.*

In other words, the $S(q)$'s are pairwise disjoint and $\bigcup_{q \in Q} S(q) = A^+$.

A *generalized Carton–Michel automaton (GCMA)* is defined as expected. It is the same as a CMA except that the set F of final states is replaced by a set $\mathfrak{F} \subseteq 2^Q$ of final sets, just as with ordinary generalized Büchi automata. For such

an automaton, a run r is final if for every $F \in \mathfrak{F}$ there exist infinitely many i such that $r(i) \in F$.

The above definitions for CMA's can all be adapted to GCMA's in a natural fashion and all the statements hold accordingly. For instance, a word u is a loop at some state q in a GCMA if $q = uq$ and for every $F \in \mathcal{F}$ there exist $v, w \in A^*$ such that $u = vw$ and $wq \in F$. It is a theorem by Carton and Michel that every GCMA can be converted into an equivalent CMA.

2.2   Temporal Logic

In the following, it is understood that temporal logic refers to propositional linear-time future temporal logic where the natural numbers are used as the domain of time. For background on temporal logic, we refer to [6] and [8].

Given an alphabet A, the set of *temporal formulas over A,* denoted TL_A, is typically inductively defined by:

(i)   for every $a \in A$, the symbol a is an element of TL_A,
(ii) if $\varphi \in \mathrm{TL}_A$, so is $\neg\varphi$,
(iii) if $\varphi, \psi \in \mathrm{TL}_A$, so are $\varphi \vee \psi$ and $\varphi \wedge \psi$,
(iv) if $\varphi \in \mathrm{TL}_A$, so is $\mathsf{X}\varphi$ ("next φ"),
(v)   if $\varphi \in \mathrm{TL}_A$, so are $\mathsf{F}\varphi$ and $\mathsf{G}\varphi$ ("eventually φ" and "always φ"),
(vi) if $\varphi, \psi \in \mathrm{TL}_A$, so are $\varphi\mathsf{U}\psi$ and $\varphi\mathsf{R}\psi$ ("φ until ψ" and "φ releases ψ").

Often, the operators XF ("strictly eventually") and XG ("strictly always") are part of the syntax of temporal logic; we view them as abbreviations of XF and XG. (Obviously, F and G can be viewed as abbreviations of $(a \vee \neg a)\mathsf{U}$ and $(a \wedge \neg a)\mathsf{R}$, respectively.)

Formulas of TL_A are interpreted in ω-words over A. For every such word u, we define what it means for a formula to hold in u, denoted $u \vDash \varphi$, where we omit the straightforward rules for Boolean connectives:

- $u \vDash a$ if $u(0) = a$,
- $u \vDash \mathsf{X}\varphi$ if $u[1, *) \vDash \varphi$, where, as usual, $u[1, *)$ denotes the word $u(1)u(2)\ldots$,
- $u \vDash \mathsf{F}\varphi$ if there exists $i \geq 0$ such that $u[i, *) \vDash \varphi$, similarly, $u \vDash \mathsf{G}\varphi$ if $u[i, *) \vDash \varphi$ for all $i \geq 0$,
- $u \vDash \varphi\mathsf{U}\psi$ if there exists $j \geq 0$ such that $u[j, *) \vDash \psi$ and $u[i, *) \vDash \varphi$ for all $i < j$, similarly, $u \vDash \varphi\mathsf{R}\psi$ if there exists $j \geq 0$ such that $u[j, *) \vDash \varphi$ and $u[i, *) \vDash \psi$ for all $j \leq i$ or if $u[i, *) \vDash \psi$ for all $i \geq 0$.

Clearly, a formula of the form $\neg\mathsf{F}\varphi$ is equivalent to $\mathsf{G}\neg\varphi$, and a formula of the form $\neg(\varphi\mathsf{U}\psi)$ is equivalent to $\neg\varphi\mathsf{R}\neg\psi$, which means F and G as well as U and R are dual to each other; X is self-dual.

Given a TL_A formula φ, we write $\mathrm{L}(\varphi)$ for the set of ω-words over A where φ holds, that is, $\mathrm{L}(\varphi) = \{u \in A^\omega : u \vDash \varphi\}$. This ω-language is called the *language defined by φ.*

Fragments of LTL. An *operator set* is a subset of the set of all basic temporal operators, $\{\mathsf{X}, \mathsf{F}, \mathsf{XF}, \mathsf{U}\}$. If A is an alphabet and O an *operator set*, then $\mathrm{TL}_A[O]$ denotes all LTL formulas that can be built from A using Boolean connectives

and the operators from O. We say a language $L \subseteq A^\omega$ is *O-expressible* if there is a formula $\varphi \in \mathrm{TL}_A[O]$ such that $\mathrm{L}(\varphi) = L$. The *O-fragment* is the set of all LTL-formulas φ such that $\mathrm{L}(\varphi)$ is O-expressible.

Observe that several operator sets determine the same fragment: $\{\mathsf{XF}\}$ and $\{\mathsf{F},\mathsf{XF}\}$; $\{\mathsf{U}\}$ and $\{\mathsf{F},\mathsf{U}\}$; $\{\mathsf{XF},\mathsf{U}\}$ and $\{\mathsf{F},\mathsf{XF},\mathsf{U}\}$; $\{\mathsf{X},\mathsf{F}\}$, $\{\mathsf{X},\mathsf{XF}\}$ and $\{\mathsf{X},\mathsf{F},\mathsf{XF}\}$; $\{\mathsf{X},\mathsf{U}\}$ and every superset of this.

What we are aiming at are decision procedures for each fragment except for the one determined by $\{\mathsf{XF},\mathsf{U}\}$, as this is kind of unnatural: a strict operator combined with a non-strict one.

Ehrenfeucht–Fraïssé Games for LTL. The statements of our results (Section 4.2) do not involve Ehrenfeucht–Fraïssé games (EF games), but we use them extensively in our proofs. In this extended abstract, we make use of them in Section 5.

In the following, we recall the basics of EF games for temporal logic, see [7] for details.

A play of a temporal logic EF game is played by two players, Spoiler and Duplicator, on two ω-words over some alphabet A, say u and v. The game is played in rounds, where in every round, Spoiler moves first and Duplicator replies. The basic idea is that Spoiler is trying to reveal a difference between u and v which can be expressed in temporal logic, while Duplicator is trying to show—by somehow imitating the moves of Spoiler—that there is no such difference.

There are different types of rounds, corresponding to the temporal operators considered. We explain the ones that we need:

X-*round.* Spoiler chooses either u or v, say v, and chops off the first letter of v, that is, he replaces v by $v[1, *)$. Duplicator does the same for u.

F-*round.* Spoiler chooses either u or v, say v, and chops off an arbitrary finite (possibly empty) prefix, that is, he replaces v by $v[i, *)$ for some $i \geq 0$. Duplicator replaces u (the other word) by $u[j, *)$ for some $j \geq 0$.

XF-*round.* Spoiler chooses either u or v, say v, and chops off an arbitrary non-empty finite prefix, that is, he replaces v by $v[i, *)$ for some $i > 0$. Duplicator replaces u (the other word) by $u[j, *)$ for some $j > 0$.

Before the first round, $u(0)$ and $v(0)$ are compared. If they are distinct, then this is a win (an early win) for Spoiler. After each round, the same condition is verified, and, again, if the two symbols are distinct, then this is a win for Spoiler. If, by the end of a play, Spoiler hasn't won, then this play is a win for Duplicator. For a fixed n, Duplicator wins the n-round game, if Duplicator has a strategy to win it.

When only rounds are allowed that correspond to operators in a temporal operator set $O \subseteq \{\mathsf{X},\mathsf{F},\mathsf{XF}\}$, then we speak of an *O-game*.

The fundamental property of EF games we are going to use is the following, which was essentially proved in [7].

Theorem 2. *Let L be a language of ω-words over some alphabet A and $O \subseteq \{\mathsf{X},\mathsf{F},\mathsf{XF}\}$ a temporal operator set. Then the following are equivalent:*

(A) L is O-expressible.

(B) There is some k such that for all words $u, v \in A^{\omega}$ with $u \in L \leftrightarrow v \in L$, Spoiler has a strategy to win the O-game on u and v within k rounds.

3   From Temporal Logic to CMA's

Several translations from temporal logic into Büchi and generalized Büchi automata are known, see, for instance, [24,21,9]. Here, we follow the same ideas and "observe" that the resulting automaton is a GCMA. This is supposed to be folklore,[1] but—to the best of our knowledge—has not been made precise yet.

First note that every formula can be assumed to be in negation normal form, which means (ii) from above is not used. So, without loss of generality, we only work with formulas of this form in what follows.

Let $\varphi \in \mathrm{TL}_A$ and let $\mathrm{sub}(\varphi)$ denote the set of its subformulas. We define a GCMA $\boldsymbol{A}[\varphi] = (A, 2^{\mathrm{sub}(\varphi)}, I, \cdot, \mathfrak{F})$. Our goal is to construct the automaton in such a way that in the unique final run r of this automaton on a given word u the following holds for every i and every $\psi \in \mathrm{sub}(\varphi)$:

$$u[i, *] \vDash \psi \qquad \text{iff} \qquad \psi \in r(i) \ . \tag{1}$$

First, we set $I = \{\Phi \subseteq \mathrm{sub}(\varphi) : \varphi \in \Phi\}$ (which is motivated directly by (1)).

Second, we define $a \cdot \Phi$ to be the smallest set Ψ satisfying the following conditions:

 (i) $a \in \Psi$,
 (ii) if $\psi \in \Psi$ and $\chi \in \Psi$, then $\psi \wedge \chi \in \Psi$,
 (iii) if $\psi \in \Psi$ or $\chi \in \Psi$, then $\psi \vee \chi \in \Psi$,
 (iv) if $\psi \in \Phi$, then $\mathsf{X}\psi \in \Psi$,
 (v) if $\psi \in \Psi$ or $\mathsf{F}\psi \in \Phi$, then $\mathsf{F}\psi \in \Psi$,
 (vi) if $\psi \in \Psi$ and $\mathsf{G}\psi \in \Phi$, then $\mathsf{G}\psi \in \Psi$,
 (vii) if $\chi \in \Psi$ or if $\psi \in \Psi$ and $\psi\mathsf{U}\chi \in \Phi$, then $\psi\mathsf{U}\chi \in \Psi$,
 (viii) if $\chi \in \Psi$ and if $\psi \in \Psi$ or $\psi\mathsf{R}\chi \in \Phi$, then $\psi\mathsf{R}\chi \in \Psi$.

This definition reflects the "local semantics" of temporal logic, for instance, $\mathsf{F}\psi$ is true now if, and only if, ψ is true now or $\mathsf{F}\psi$ is true in the next point in time. Observe, however, that the fulfillment of $\mathsf{F}\psi$ must not be deferred forever, which means that local conditions are not enough to capture the entire semantics of temporal logic. This is taken care of by the final sets.

Third, for every formula $\mathsf{F}\psi \in \mathrm{sub}(\varphi)$ the set $\{\Phi \subseteq \mathrm{sub}(\varphi) : \psi \in \Phi$ or $\mathsf{F}\psi \notin \Phi\}$ is a member of $\mathfrak{F}$. Similarly, for every formula $\psi\mathsf{U}\chi$ the set $\{\Phi \subseteq \mathrm{sub}(\varphi) : \chi \in \Phi$ or $\psi\mathsf{U}\chi \notin \Phi\}$ is a member of $\mathfrak{F}$. No other set belongs to $\mathfrak{F}$.

Proposition 1. Let A be an alphabet and $\varphi \in TL_A$. Then $\boldsymbol{A}_{\varphi}$ is a GCMA and $\mathrm{L}(\boldsymbol{A}_{\varphi}) = \mathrm{L}(\varphi)$.

---

[1] Personal communication of the second author with Olivier Carton: the observation can already be found in the notes by Max Michel which he handed over to Olivier Carton in the last millennium.

Proof. We first show that $\boldsymbol{A}_\varphi$ is a GCMA. To this end, let u be an ω-word over A. We show that the word r defined by (1), for every i and every $\psi \in \mathrm{sub}(\varphi)$, is a final run on u and the only one.

The ω-word r is a run on u. To see this, let $i \geq 0$ be arbitrary and observe that if we define Φ and Ψ by $\Phi = \{\psi \in \mathrm{sub}(\varphi) : u[i+1,*] \vDash \psi\}$ and $\Psi = \{\psi \in \mathrm{sub}(\varphi) : u[i,*] \vDash \psi\}$, then the implications (i)–(viii) not only hold, but also hold in the opposite direction. That is, $r(i) = u(i) \cdot r(i+1)$ for every i, in other words, r is a run on u.

The run r is final. Obvious from the semantics of temporal logic.

The run r is the only possible final run. Let s be a final run. We need to show $s = r$. To this end, one shows by an inductive proof on the structure of the elements of $\mathrm{sub}(\varphi)$ that for every i and every $\psi \in \mathrm{sub}(\varphi)$ condition (1) holds for s and hence $s = r$. The interesting part is the inductive step for formulas of the form $\mathsf{F}\psi$ and $\psi\mathsf{U}\chi$. We only show how the proof works for $\mathsf{F}\psi$.

If $\mathsf{F}\psi \in s(i)$, then $u[i,*] \vDash \mathsf{F}\psi$. Suppose $\mathsf{F}\psi \in s(i)$. Then, by definition of $\cdot$, $\psi \in s(i)$ or $\mathsf{F}\psi \in s(i+1)$. In the first case, we have $u[i,*] \vDash \psi$ by the induction hypothesis, so $u[i,*] \vDash \mathsf{F}\psi$ by the semantics of temporal logic, and we are done. In the second case, we can apply the same argument to the next point in time and get $u[i+1,*] \vDash \mathsf{F}\psi$, hence $u[i,*] \vDash \mathsf{F}\psi$, or we get $\mathsf{F}\psi \in s(i+2)$. Continuing like this, we eventually (!) get $u[i,*] \vDash \mathsf{F}\psi$ or that $\mathsf{F}\psi \in s(j)$ for every $j \geq i$. Because s is final, one of the states of the final set for $\mathsf{F}\psi$ must occur somewhere, that is, we get $\psi \in s(j)$ for some $j \geq i$, thus $u[j,*] \vDash \psi$ by induction hypothesis, hence $u[i,*] \vDash \mathsf{F}\psi$.

If $\mathsf{F}\psi \notin s(i)$, then $u[i,*] \nvDash \mathsf{F}\psi$. If $\mathsf{F}\psi \notin s(i)$, then, because of the definition of $\cdot$, $\psi \notin s(i)$ and $\mathsf{F}\psi \notin s(i+1)$. Continuing like this, we get $\psi \notin s(j)$ for every $j \geq i$, so, by induction hypothesis, $u[j,*] \nvDash \psi$ for every $j \geq i$, hence $u[i,*] \nvDash \mathsf{F}\psi$.

Correctness of the construction, that is, $\mathrm{L}(\varphi) = \mathrm{L}(\boldsymbol{A}_\varphi)$. This follows immediately from what was shown before because of the way I is defined.   □

4   General Approach and Individual Results

This section has two purposes: it explains our general approach and presents the characterizations we have found.

4.1   The General Approach

To describe our general approach, we first need to explain what we understand by the left congruence of a (G)CMA.

Let $\boldsymbol{A}$ be a (G)CMA. For every $q \in Q$, let L_q denote the set of words $u \in A^*$ such that $uq \in I$. The relation $\equiv_{\boldsymbol{A}}$ on Q, which we call the *left congruence of $\boldsymbol{A}$*, is defined by $q \equiv_{\boldsymbol{A}} q'$ when $L_q = L_{q'}$. The terminology is justified:

Remark 1. Let $\boldsymbol{A}$ be a (G)CMA. Then $\equiv_{\boldsymbol{A}}$ is a left congruence, that is, $uq \equiv_{\boldsymbol{A}} uq'$ whenever $u \in A^*$ and $q, q' \in A$ are such that $q \equiv_{\boldsymbol{A}} q'$.

In other words, we can define the *left quotient of* **A** with respect to $\equiv_{\boldsymbol{A}}$ to be the reverse semi DFA $\mathbf{A}/\!\!\equiv_{\boldsymbol{A}}$ given by

$$\mathbf{A}/\!\!\equiv_{\boldsymbol{A}} = (A, Q'/\!\!\equiv_{\boldsymbol{A}}, I/\!\!\equiv_{\boldsymbol{A}}, \circ) \tag{2}$$

where

- Q' is the set of all states that occur in some final run of $\boldsymbol{A}$ (active states), and
- $a \circ (q/\!\!\equiv_{\boldsymbol{A}}) = (a \cdot q)/\!\!\equiv_{\boldsymbol{A}}$ for all $a \in A$ and $q \in Q'$.

As usual, the attribute "semi" refers to the fact that this automaton has no final states nor final sets.

Next, we combine the left congruence of a (G)CMA with its loops. The *loop language of a state q* of a (G)CMA $\boldsymbol{A}$ is denoted $\mathrm{LL}(q)$ and defined by

$$\mathrm{LL}(q) = \bigcup_{q':q' \equiv_{\boldsymbol{A}} q} S(q') \ , \tag{3}$$

that is, $\mathrm{LL}(q)$ contains all loops at q and at congruent states.

Our general approach is to characterize a fragment of LTL as follows. To check whether a given formula φ is equivalent to a formula in a given fragment, we compute the GCMA $\boldsymbol{A}_\varphi$ and check various conditions on its left quotient and its loop languages. It turns out that this is sufficient; intuitively, the left quotient accounts for the "finitary fraction" of $\mathrm{L}(\boldsymbol{A}_\varphi)$, whereas the loop languages account for its "infinitary fraction".

4.2   Characterization of the Individual Fragments

The formal statement of our main result is as follows.

Theorem 3. *Let A be some alphabet, φ an LTL-formula, and O a temporal operator set as listed in Table 1. Then the following are equivalent:*

(A)   The formula φ belongs to the O-fragment.

(B)   The left quotient of $\boldsymbol{A}_\varphi$ and its loop languages satisfy the respective conditions listed in Table 1. (Information on how to read this table follows.)
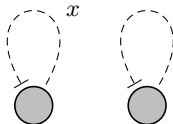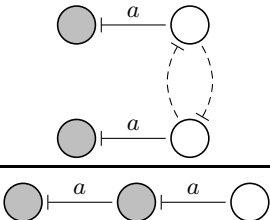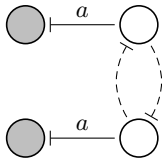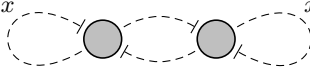
Conditions on the left quotient of $\boldsymbol{A}_\varphi$ are phrased in terms of "forbidden patterns" (also called "forbidden configurations" in [4]). To explain this, let $\boldsymbol{A} = (A, Q, I, \circ)$ be any reverse semi DFA. Its *transition graph*, denoted $\mathrm{T}(\boldsymbol{A})$, is the A-edge-labeled directed graph (Q, E) where $E = \{(a \circ q, a, q) : a \in A, q \in Q\}$.

Now, the conditions depicted in the second column of Table 1 are to be read as follows: the displayed graph(s) do not (!) occur as subgraphs of the transition graph of the left quotient of $\boldsymbol{A}_\varphi$, that is, as subgraphs of $\mathrm{T}(\boldsymbol{A}_\varphi/\!\!\equiv_{\boldsymbol{A}_\varphi})$. Vertices filled gray must be distinct, the others may coincide (even with gray ones); dashed arrows stand for non-trivial paths.

For instance, the condition for the left quotient in the case of the $\{\mathsf{X}\}$-fragment requires that the following is not true for $\mathrm{T}(\boldsymbol{A}_\varphi/\!\!\equiv_{\boldsymbol{A}_\varphi})$: there exist distinct states q and q' and a word $x \in A^+$ such that $q = x \circ q$ and $q' = x \circ q'$.

Table 1. Characterizations of the individual fragments of LTL

fragment	left quotient	loop languages
X		no condition
F		1-locally testable
XF		1-locally testable
X, F		locally testable
U		stutter-invariant

Note that for the $\{X\}$-fragment one forbidden pattern consisting of two strongly connected components is listed, whereas for the $\{F\}$-fragment two forbidden patterns (indicated by the horizontal line) are listed.

The conditions listed in the third column of Table 1 are conditions borrowed from formal language theory, which we explain in what follows. For a word $u \in A^*$ and $k \geq 0$, we let $\mathrm{prf}_k(u)$, $\mathrm{sffx}_k(u)$, and $\mathrm{occ}_k(u)$ denote the set of prefixes, suffixes, and infixes of u of length $\leq k$, respectively. For words $u, v \in A^*$, we write $u \equiv_{k+1} v$ if $\mathrm{prf}_k(u) = \mathrm{prf}_k(v)$, $\mathrm{occ}_{k+1}(u) = \mathrm{occ}_{k+1}(v)$, and $\mathrm{sffx}_k(u) = \mathrm{sffx}_k(v)$. A language L is called $(k+1)$-locally testable if $u \in L \leftrightarrow v \in L$, whenever $u \equiv_k v$, and it is called *locally testable* if it is k-locally testable for some k, see [1].

A language $L \subseteq A^+$ is *stutter-invariant* if $uav \in L \leftrightarrow uaav \in L$ holds for all $a \in A$, $u, v \in A^*$.

4.3   Proof Techniques

For each fragment dealt with in Theorem 3, we have a separate proof, some of them are similar, others are completely different. In this section, we give a brief overview of our proofs.

For the operator set $\{X\}$, the proof is more or less a simple exercise, given that $\{X\}$-expressibility means that there is some k such that $u \vDash \varphi$ is determined by $\mathrm{prf}_k(u)$.

For the operator sets $\{F\}$, $\{XF\}$, and $\{X, F\}$, we use similar proofs. As an example, we treat the simplest case, $\{XF\}$, in the next section.

For $\{U\}$, we use a theorem from [14], which says that an LTL formula over some alphabet A is equivalent to a formula in $\mathrm{TL}_A[U]$ if the language defined by the formula is stutter-invariant, where stutter invariance is defined using an appropriate notion of stutter equivalence on ω-words.

5   Characterization of the $\{XF\}$-Fragment

We present the characterization of the $\{XF\}$-fragment in detail. Since every GCMA can obviously be turned into an equivalent trim GCMA, all GCMA are assumed to be trim subsequently.

We start with a refined version of Theorem 3 for the $\{XF\}$-fragment.

Theorem 4. *The following are equivalent for a given GCMA $\boldsymbol{A}$:*

(A) $L(\boldsymbol{A})$ is XF-expressible.
(B) (a) The transition graph $\mathrm{T}(\boldsymbol{A}/\equiv_{\boldsymbol{A}})$ doesn't have a subgraph of the following form (in the above sense):

$$(T1)$$

(b) For all $u, v \in A^+$ with $\mathrm{occ}(u) = \mathrm{occ}(v)$, it holds that $u\text{\textbackslash} \equiv_{\boldsymbol{A}} v\text{\textbackslash}$.
(C) (a) The same as in (B)(a).
   (b)   (i) For all $u, v \in A^*$, $a \in A$, it holds that $uav\text{\textbackslash} \equiv_{\boldsymbol{A}} uaav\text{\textbackslash}$.
       (ii) For all $u, v \in A^*$, $a, b \in A$, it holds that $uabv\text{\textbackslash} \equiv_{\boldsymbol{A}} ubav\text{\textbackslash}$.

Observe that (B)(b) means that the loop languages are 1-locally testable. In other words, the above theorem implies that the characterization of the $\{XF\}$-fragment given in Theorem 3 is correct.

Before we get to the proof of Theorem 4 we provide some more notation and prove some useful lemmas. For ease in notation, we often write $\bar{q}$ for $q/\equiv_{\boldsymbol{A}}$. When $u \in A^\omega$, then $u \cdot \infty$ denotes the first state of the unique final run of $\boldsymbol{A}$ on u, and $\inf(u) = \{a \in A : \exists^\infty i(u(i) = a)\}$. For $a \in A$ and $u \in A^*$, $|u|_a$ denotes the number of occurrences of a in u.

Lemma 2. *Assume* $T(\boldsymbol{A}/{\equiv}_{\boldsymbol{A}})$ *has a subgraph of type (T1). Then for every k there exist words $u, v \in A^{\omega}$ such that Duplicator wins the k-round XF-game on u and v, but $u \in L(\boldsymbol{A}) \leftrightarrow v \in L(\boldsymbol{A})$.*

Proof. Assume $T(\boldsymbol{A}/{\equiv}_{\boldsymbol{A}})$ has a subgraph of type (T1). That is, there are states $\bar{p} \neq \bar{q}, \bar{r}, \bar{s}$, words $x, y \in A^{+}$, and a letter $a \in A$ such that $\bar{p} = a \cdot \bar{r}$, $\bar{q} = a \cdot \bar{s}$, $\bar{s} = y \cdot \bar{r}$ and $\bar{r} = x \cdot \bar{s}$. We find states $r_0, r_1, \ldots,$ and $s_0, s_1, \ldots$ such that
- $\bar{r}_i = \bar{r}$ and $\bar{s}_i = \bar{s}$ for all $i < \omega$, and
- $x \cdot s_i = r_i$ and $y \cdot r_i = s_{i+1}$ for all $i < \omega$.

Because Q is a finite set, we find $l > 0$ and i such that $r_i = r_{i+l}$. Since $\boldsymbol{A}$ is trim, we find v such that $v \cdot \infty = r_i$ and u such that $ua \cdot r_i \in I$ iff $ua \cdot s_i \notin I$. This means that $ua(yx)^{lm} v \in L \leftrightarrow uax(yx)^{lm} v \in L$ for all $m \geq 1$.

Clearly, if we choose $lm > k$, then the two resulting words cannot be distinguished in the k-round XF-game. $\qquad\square$

Lemma 3. *Let $\boldsymbol{A}$ be a GCMA such that $T(\boldsymbol{A}/{\equiv}_{\boldsymbol{A}})$ doesn't have a subgraph of type (T1). Further, let r and s be the unique final runs of $\boldsymbol{A}$ on words $u, v \in A^{\omega}$ and define $\bar{r}$ and $\bar{s}$ by $\bar{r}(i) = r(i)/{\equiv}_{\boldsymbol{A}}$ and $\bar{s}(i) = s(i)/{\equiv}_{\boldsymbol{A}}$ for all $i < \omega$.*

If $\bar{r}(0) \neq \bar{s}(0)$ and $\inf(\bar{r}) \cap \inf(\bar{s}) \neq \varnothing$, then Spoiler wins the k-round XF-game on u and v where k is twice the number of states of $\boldsymbol{A}/{\equiv}_{\boldsymbol{A}}$.

Proof. In the following, we use SCC as an abbreviation for strongly connected component. In our context, a state which is not reachable by a non-trivial path from itself is considered to be an SCC by itself. For every $i < \omega$, let R_i and S_i be the SCC's of $\bar{r}(i)$ and $\bar{s}(i)$ in $\boldsymbol{A}/{\equiv}_{\boldsymbol{A}}$, respectively. Observe that because of $\inf(\bar{r}) \cap \inf(\bar{s}) \neq \varnothing$ there is some l such that the R_i's and S_j's are all the same for $i, j \geq l$.

Let $\mathfrak{R} = \{R_i : i > 0\}$, $\mathfrak{S} = \{S_i : i > 0\}$, $m = |\mathfrak{R}| - 1$, and $n = |\mathfrak{S}| - 1$. We show that Spoiler wins the XF-game in at most $m + n$ rounds. The proof is by induction on $m + n$.

Base case. Let $m = n = 0$. Then $R_1 = S_1$. Because of the absence of (T1), we have $u(0) \neq v(0)$, and Spoiler wins instantly.

Induction step. Note that if r is the unique final run of $\boldsymbol{A}$ on u, then $r[i, *)$ is the unique final run of $\boldsymbol{A}$ on $u[i, *)$ for every i.

Let $m + n > 0$. If $u(0) \neq v(0)$, then Spoiler wins instantly. If $u(0) = v(0)$, we proceed by a case distinction as follows.

Case 1, $R_1 = S_1$. This is impossible because of the absence of (T1).

Case 2, $R_1 \neq S_1$, $R_1 \notin \mathfrak{S}$. Since $R_1 \notin \mathfrak{S}$ and $\inf(\bar{r}) \cap \inf(\bar{s}) \neq \varnothing$ we have $m > 0$. So there must be some $i \geq 1$ such that $\bar{r}(i) \in R_1$ and $\bar{r}(i+1) \notin R_1$. Spoiler chooses the word u and replaces u by $u[i, *)$.

Now Duplicator has to replace v by $v[j, *)$ for some $j > 0$. Since $R_1 \notin \mathfrak{S}$ we have $\bar{r}(i) \neq \bar{s}(j)$ and the induction hypothesis applies.

Case 3, $R_1 \neq S_1$, $S_1 \notin \mathfrak{R}$. Symmetric to Case 2.

Case 4, $R_1 \neq S_1$, $R_1 \in \mathfrak{S}$, and $S_1 \in \mathfrak{R}$. Impossible, because R_1 would be reachable from S_1 and vice versa, which would mean R_1 and S_1 coincide. $\qquad\square$

Lemma 4. *Let A be a (G)CMA. Then the following are equivalent:*

(A) For all $u, v \in A^+$ with $\mathrm{occ}(u) = \mathrm{occ}(v)$, it holds that $u\rtimes \equiv_A v\rtimes$.
(B) (a) For all $u, v \in A^*$, $a \in A$, it holds that $uav\rtimes \equiv_A uaav\rtimes$.
(b) For all $u, v \in A^*$, $a, b \in A$, it holds that $uabv\rtimes \equiv_A ubav\rtimes$.

Proof. That (A) implies (B) is obvious. For the converse, let $u, v \in A^+$ with $\mathrm{occ}(u) = \mathrm{occ}(v)$. Let $\mathrm{occ}(u) = \{a_0, a_1, \ldots, a_n\}$. Now, we have

$$u\rtimes \equiv_A a_0^{|u|_{a_0}} a_1^{|u|_{a_1}} \ldots a_n^{|u|_{a_n}} \rtimes \equiv_A a_0^{|v|_{a_0}} a_1^{|v|_{a_1}} \ldots a_n^{|v|_{a_n}} \rtimes \equiv_A v\rtimes \ ,$$

where the first and the last equivalence are obtained by iterated application of (b), and the second equivalence is obtained by iterated application of (a). □

In what follows, we need more notation and terminology. A word $u \in A^\omega$ is an *infinite loop* at q if $q = u \cdot \infty$ and $q \in \inf(r)$ where r is the unique final run of A on u.

Proof of Theorem 4. The implication from (A) to (B)(a) is Lemma 2. We prove that (A) implies (B)(b) by contraposition. Assume (B)(b) does not hold, that is, there are $u, v \in A^+$ with $\mathrm{occ}(u) = \mathrm{occ}(v)$, and $u\rtimes \not\equiv_A v\rtimes$. Then there exists $x \in A^*$ such that $x \cdot u\rtimes \in I \nleftrightarrow x \cdot v\rtimes \in I$, that is, $xu^\omega \in L \nleftrightarrow xv^\omega \in L$. It is easy to see that Duplicator wins the XF-game on xu^ω and xv^ω for any number of rounds, which, in turn, implies L is not XF-expressible.

For the implication from (B) to (A), let n be the number of states of $A/{\equiv_A}$. We show that whenever $u, v \in A^\omega$ such that $u \in L \leftrightarrow v \in L$, then Spoiler wins the $2n$-round XF-game on u and v.

Assume $u, v \in A^\omega$ are such that $u \in L \leftrightarrow v \in L$ and let r and s be the unique final runs of A on u and v, respectively, and $\bar{r}$ and $\bar{s}$ defined as in Lemma 3. We distinguish two cases.

First case, $\inf(u) \neq \inf(v)$. Then Spoiler wins within at most 2 rounds.

Second case, $\inf(u) = \inf(v)$. Then there are i, i' and j, j' such that

- $\mathrm{occ}(u[i, j]) = \mathrm{occ}(v[i', j'])$,
- $u[i, *) \cdot \infty$ is an infinite loop at $u[i, j]\rtimes$, and
- $v[i', *) \cdot \infty$ is an infinite loop at $v[i', j']\rtimes$.

From (B)(b), we conclude $u[i, j]\rtimes \equiv_A v[i', j']\rtimes$. As a consequence, $\inf(\bar{r}) \cap \inf(\bar{s}) \neq \varnothing$. Since $\bar{r}(0) \neq \bar{s}(0)$, Lemma 3 applies: L is XF-expressible.

The equivalence between (B) and (C) follows directly from Lemma 4. □

6 Effectiveness and Computational Complexity

To conclude, we explain how Theorem 3 can be used effectively. In general, we have:

Theorem 5. *Each of the fragments listed in Table 1 is decidable.*

Observe that for the fragment with operator set $\{\mathsf{F}, \mathsf{U}\}$, this is a result from [15], and for the fragment with operator set $\{\mathsf{X}, \mathsf{F}\}$, this is a result from [22].

Proof (of Theorem 5). First, observe that $\boldsymbol{A}_\varphi$ can be constructed effectively. Also, it is easy to derive the left quotient of $\boldsymbol{A}_\varphi$ from $\boldsymbol{A}_\varphi$ itself and DFA's for the loop languages, even minimum-state DFA's for them.

Second, observe that the presence of the listed forbidden patterns can be checked effectively. The test for the existence of a path between two states can be restricted to paths of length at most the number of states; the test for the existence of two paths with the same label (see forbidden patterns for $\{\mathsf{X}\}$ and $\{\mathsf{X},\mathsf{F}\}$) can be restricted to paths of length at most the number of states squared.

Third, the conditions on the loop languages can be checked effectively. For 1-local testability, this is because a language $L \subseteq A^*$ is not 1-locally testable if, and only if, one of the following conditions holds:

1. There are words $u, v \in A^*$ and there is a letter $a \in A$ such that $uav \in L \leftrightarrow uaav \in L$.

2. There are words $u \in A^*, v \in A^*$ and letters $a, b \in A$ such that $uabv \in L \leftrightarrow ubav \in L$.

Again, u and v can be bounded in length by the number of states. For local testability, we refer to [11], where it was shown this can be decided in polynomial time. For stutter invariance, remember that a language $L \subseteq A^*$ is not stutter-invariant if, and only if, the above condition 1. holds. So this can be checked effectively, too. (One could also use the forbidden pattern listed.)   □

As to the computational complexity of the problems considered, we first note:

Proposition 2. *Each of the fragments listed in Table 1 is PSPACE-hard.*

Proof. The proof is an adaptation of a proof for a somewhat weaker result given in [15]. First, recall that LTL satisfiability is PSPACE-hard for some fixed alphabet [20], hence LTL unsatisfiability for this alphabet is PSPACE-hard, too. Let A denote such an alphabet in the following.

Second, note that (because all fragments considered are proper fragments of LTL) there exists some alphabet B such that for each operator set O in question there exists an LTL formula α_O such that α_O is not O-expressible.

We can now describe a reduction from LTL unsatisfiability to the O-fragment using formulas over the alphabet $A \times B$. Let α'_O be the formula which is obtained from α by replacing every occurrence of a letter b by the formula $\bigvee_{a \in A}(a, b)$. Given a formula φ over A, we first construct $\hat{\varphi}$, where $\hat{\varphi}$ is obtained from φ by replacing every occurrence of a letter a by the formula $\bigvee_{b \in B}(a, b)$. Then φ is satisfiable iff $\hat{\varphi}$ is satisfiable iff $\hat{\varphi} \wedge \alpha'_O$ is satisfiable. Moreover, $\hat{\varphi} \wedge \alpha'_O$ cannot be expressed in the fragment in question, provided $\hat{\varphi}$ is satisfiable. Therefore, $\hat{\varphi} \wedge \alpha'_O$ is equivalent to a formula in the fragment iff φ is unsatisfiable.   □

Our upper bounds are as follows:

Theorem 6. *The $\{\mathsf{X},\mathsf{F}\}$-fragment is in E, the other fragments listed in Table 1 are in PSPACE.*

Observe that the result for the $\{\mathsf{U}\}$-fragment is not new, but was already obtained in [15].

Proof (sketch). Observe that each property expressed as forbidden pattern can not only be checked in polynomial time (which is folklore), it can also be checked non-deterministically in logarithmic space, even if we are given a GCMA and need to check it on its left quotient. So if we compose the construction of $\boldsymbol{A}_\varphi$, which has an exponential number of states, with the non-deterministic logarithmic-space tests for the existence of forbidden patterns, we obtain a polynomial-space procedure for testing the conditions on $\mathrm{T}(\boldsymbol{A}_\varphi/{\equiv_{\boldsymbol{A}_\varphi}})$.

The situation is more complicated for the conditions on the loop languages. We first deal with 1-local testability and stutter invariance. Observe that from the automaton $\boldsymbol{A}_\varphi$ we can get reverse DFA's of polynomial size in the size of $\boldsymbol{A}_\varphi$ such that every loop language is the union of the languages recognized by these reverse DFA's. Moreover, 1. and 2. from the proof of Theorem 5 can be transferred to this context as follows. There are two states p and q in $\boldsymbol{A}_\varphi$ that are not equivalent with respect to $\equiv_{\boldsymbol{A}}$ and such that one of the following conditions is true:

1. There are words $u, v \in A^*$ and there is a letter $a \in A$ such that $uav \in \mathrm{LL}(p)$ and $uaav \in \mathrm{LL}(q)$.

2. There are words $u, v \in A^*$ and letters $a, b \in A$ such that $uabv \in \mathrm{LL}(p)$ and $ubav \in \mathrm{LL}(q)$.

From this, it follows that we can bound the length of u and v polynomially in the size of $\boldsymbol{A}_\varphi$, which again yields polynomial-space procedures for both, 1-local testability and stutter invariance.

For (general) local testability, we apply [11] to the product of the reverse DFA's mentioned above, which yields an exponential-time algorithm all together.   □

7   Conclusion

We would like to state some questions:

1. Our lower and upper bounds for the complexity of the $\{\mathsf{X}, \mathsf{F}\}$-fragment don't match. What is the exact complexity of this fragment?

2. Clearly, from our proofs it can be deduced that if a formula φ is equivalent to a formula in a fragment, an equivalent formula can be constructed effectively. What is the complexity of this construction task?

3. It is not difficult to come up with examples where every equivalent formula has exponential size (even exponential circuit size). What is the worst-case blow-up?— Observe that, in terms of circuit size, there is a polynomial upper bound for the $\{U\}$-fragment, see [12].

References

1. Brzozowski, J.A., Simon, I.: Characterizations of locally testable events. Discrete Math. 4(3), 243–271 (1973)
2. Carton, O., Michel, M.: Unambiguous Büchi Automata. In: Gonnet, G.H., Panario, D., Viola, A. (eds.) LATIN 2000. LNCS, vol. 1776, pp. 407–416. Springer, Heidelberg (2000)

3. Carton, O., Michel, M.: Unambiguous Büchi automata. Theor. Comput. Sci. 297, 37–81 (2003)
4. Cohen, J., Perrin, D., Pin, J.-É.: On the expressive power of temporal logic. J. Comput. System Sci. 46(3), 271–294 (1993)
5. Diekert, V., Kufleitner, M.: Fragments of first-order logic over infinite words. Theory Comput. Syst. 48(3), 486–516 (2011)
6. Allen Emerson, E.: Temporal and modal logic. In: Handbook of Theoretical Computer Science, vol. B, pp. 995–1072. Elsevier, Amsterdam (1990)
7. Etessami, K., Wilke, T.: An until hierarchy and other applications of an Ehrenfeucht-Fraïssé game for temporal logic. Inf. Comput. 160(1-2), 88–108 (2000)
8. Gabbay, D.M., Hodkinson, I., Reynolds, M.: Temporal logic: Mathematical Foundations and Computational Aspects, vol. 1. Clarendon Press, New York (1994)
9. Gerth, R., Peled, D., Vardi, M.Y., Wolper, P.: Simple on-the-fly automatic verification of linear temporal logic. In: Dembinski, P., Sredniawa, M. (eds.) Protocol Specification, Testing and Verification. IFIP Conference Proceedings, vol. 38, pp. 3–18. Chapman & Hall (1995)
10. Kamp, H.: Tense logic and the theory of linear order. PhD thesis, University of California, Los Angeles (1968)
11. Kim, S.M., McNaughton, R., McCloskey, R.: A polynomial time algorithm for the local testability problem of deterministic finite automata. IEEE Trans. Comput. 40, 1087–1093 (1991)
12. Etessami, K.: A note on a question of Peled and Wilke regarding stutter-invariant LTL. Inform. Process. Lett. 75(6), 261–263 (2000)
13. McNaughton, R., Papert, S.A.: Counter-free automata. MIT Press, Boston (1971)
14. Peled, D., Wilke, T.: Stutter-invariant temporal properties are expressible without the next-time operator. Inform. Process. Lett. 63(5), 243–246 (1997)
15. Peled, D., Wilke, T., Wolper, P.: An algorithmic approach for checking closure properties of temporal logic specifications and ω-regular languages. Theor. Comput. Sci. 195(2), 183–203 (1998)
16. Perrin, D.: Recent Results on Automata and Infinite Words. In: Chytil, M., Koubek, V. (eds.) MFCS 1984. LNCS, vol. 176, pp. 134–148. Springer, Heidelberg (1984)
17. Perrin, D., Pin, J.-É.: Infinite Words: Automata, Semigroups, Logic and Games. Pure and Applied Mathematics, vol. 141. Elsevier, Amsterdam (2004)
18. Pnueli, A.: The temporal logic of programs. In: FOCS, pp. 46–57. IEEE (1977)
19. Schützenberger, M.P.: On finite monoids having only trivial subgroups. Inform. and Control 8(2), 190–194 (1965)
20. Prasad Sistla, A., Clarke, E.M.: The complexity of propositional linear temporal logics. J. ACM 32, 733–749 (1985)
21. Vardi, M.Y., Wolper, P.: Reasoning about infinite computations. Inform. and Comput. 115(1), 1–37 (1994)
22. Wilke, T.: Classifying discrete temporal properties. Post-doctoral thesis, Christian-Albrechts-Universität zu Kiel (1998)
23. Wolfgang, T.: Star-free regular sets of ω-sequences. Inform. and Control 42(2), 148–156 (1979)
24. Wolper, P., Vardi, M.Y., Prasad Sistla, A.: Reasoning about infinite computation paths (extended abstract). In: FOCS, pp. 185–194. IEEE (1983)