Local Temporal Logic is Expressively Complete for Cograph Dependence Alphabets *

Volker Diekert[1] and Paul Gastin[2]

[1] Institut für Formale Methoden der Informatik (FMI)
Universität Stuttgart, Universitätsstr. 38, D-70569 Stuttgart
diekert@fmi.uni-stuttgart.de
[2] LIAFA, UMR 7089, Université Paris 7
2, place Jussieu, F-75251 Paris Cedex 05
Paul.Gastin@liafa.jussieu.fr

Abstract. Recently, local logics for Mazurkiewicz traces are of increasing interest. This is mainly due to the fact that the satisfiability problem has the same complexity as in the word case. If we focus on a purely local interpretation of formulae at vertices (or events) of a trace, then the satisfiability problem of linear temporal logics over traces turns out to be PSPACE–complete if the dependence alphabet is not part of the input. But now the difficult problem is to obtain expressive completeness results with respect to first order logic.
The main result of the paper shows such an expressive completeness result, if the underlying dependence alphabet is a cograph, i.e., if all traces are series parallel posets. Moreover, we show that this is the best we can expect: If the dependence alphabet is not a cograph, then we cannot express all first order properties in our setting.
Keywords Temporal logics, Mazurkiewicz traces, concurrency

1 Introduction

Trace theory, initiated in computer science by Keller [19] and Mazurkiewicz [21], is one of the most popular settings to study concurrency. The behavior of a concurrent process is not represented by a string, but more accurately by some labelled partial order.

A suitable way for a formal specification of concurrent systems is given by temporal logic formulae which in turn have a direct (either global or local) interpretation for Mazurkiewicz traces. It is therefore no surprise that temporal logics for traces have received quite an attention, see [2, 23–25, 27, 29]. In [30] it was shown that the basic (global) linear temporal logic with future tense operators and with past tense constants is expressively complete with respect to the first order theory of *real traces* (i.e. finite or infinite traces). In [6, 8] we have

* Partial support of EC-FET project IST-1999-29082 (ADVANCE), CEFIPRA-IFCPAR Project 2102-1 (ACSMV) and PROCOPE project 00269SD (MoVe) is gratefully acknowledged.

obtained the same result but without any past tense modalities by quite different proof techniques (which will be used here again). This positive result has solved a long standing open question [11, 30]. The price of this logic is an extremely difficult satisfiability problem, it has been shown to be non-elementary by Walukiewicz [32]. The main reason for this difficulty is the global interpretation of a formula which makes it necessary to speak about configurations, i.e., we give an interpretation of a formula for a trace with respect to some finite prefix – and the prefix structure of a trace is much more complicated than in the case of linear orders (words). If we give a local interpretation such that each formula can be evaluated at a single vertex (or event), then we obtain logics where the satisfiability problem is still in PSPACE. This is in particular the case for the logic TLC, which has been introduced by Alur et al. in [2]. The logic has been extended and studied in detail by Henriksen in [15, 16]. The logic TLC uses an existential version of the until–operator, which is not expressible in first order, in general (Section 5). On the other hand, it is not known whether one can express in TLC all first order properties for all dependence alphabets.

In our paper, we shall use a universal version of the until–operator, which, by its very definition, is first order definable. The main result of our paper shows that we obtain a local logic which is expressively complete, if the underlying dependence alphabet is a cograph, i.e., every trace is a series parallel poset (or N–free). This result is robust, the same holds for TLC or other variants how to define a semantics to the until–operator. Moreover, we show that cograph dependence alphabets are in some sense the limit where we can expect such a positive result. As long as we use no past tense modalities we cannot specify all first order properties by our logic, whether or not the until is existential or universal or whether we have both options. Our main theorems (Thms. 2, 3) are therefore *if-and-only-if* statements.

In the final section we will see that a universal until does not change complexity issues very much. The satisfiability problem of TLC augmented by this operator can still be solved in PSPACE.

An extended abstract of this paper appeared in [7].

2 Preliminaries

We briefly recall some notions concerning Mazurkiewicz traces. For the background we refer to [10]. A *dependence alphabet* is a pair (Σ, D) where the alphabet Σ is a finite set and the dependence relation $D \subseteq \Sigma \times \Sigma$ is reflexive and symmetric. The *independence relation* I is the complement of D. For $A \subseteq \Sigma$, we denote $I(A) = \{b \in \Sigma \mid (a, b) \in I \text{ for all } a \in A\}$ the set of letters independent from A and we let $D(A) = \Sigma \setminus I(A)$ be the set of letters depending on (some action in) A.

A *real trace* is (an isomorphism class of) a labelled partial order $t = [V, \leq, \lambda]$ where V is a set of vertices, $\lambda : V \to \Sigma$ is the labelling, $\leq$ is a partial order over V satisfying the following conditions: For all $x \in V$, the downward set $\{y \in V \mid y \leq x\}$ is finite, $(\lambda(x), \lambda(y)) \in D$ implies $x \leq y$ or $y \leq x$, and $x \lessdot y$

implies $(\lambda(x), \lambda(y)) \in D$, where $\lessdot = < \setminus <^2$ is the direct successor relation in the Hasse diagram of t.

The *alphabet* of the trace t is the set $\mathrm{alph}(t) = \lambda(V) \subseteq \Sigma$ and its *alphabet at infinity* $\mathrm{alphinf}(t)$ is the set of letters occurring infinitely often in t. The set of all traces is denoted by $\mathbb{R}(\Sigma, D)$ or simply by $\mathbb{R}$. A trace t is called *finite*, if V is finite. For $t = [V, \le, \lambda] \in \mathbb{R}$, we define $\min(t) \subseteq V$ as the set of all minimal vertices of t. We can read $\min(t) \subseteq \Sigma$ also as the set of labels of the minimal vertices of t. It will always be clear from the context what we actually mean. If t is finite, we define $\max(t) \subseteq V$ as the set of all maximal vertices of t and we also use $\max(t) \subseteq \Sigma$ for the set of labels of the maximal vertices of t. Note that $\max(t)$ is only defined when t is a finite trace, though the definition would make sense also for infinite traces.

We define the concatenation of two traces $t_1 = [V_1, \le_1, \lambda_1] \in \mathbb{R}$ and $t_2 = [V_2, \le_2, \lambda_2] \in \mathbb{R}$ satisfying $\mathrm{alphinf}(t_1) \times \mathrm{alph}(t_2) \subseteq I$ by $t_1 \cdot t_2 = [V, \le, \lambda]$ where $V = V_1 \cup V_2$ (assuming w.l.o.g. that $V_1 \cap V_2 = \emptyset$), $\lambda = \lambda_1 \cup \lambda_2$ and $\le$ is the transitive closure of the relation $\le_1 \cup \le_2 \cup (V_1 \times V_2 \cap \lambda^{-1}(D))$. The set of finite traces becomes a monoid which is denoted by $\mathbb{M}(\Sigma, D)$ or simply by $\mathbb{M}$. The empty trace $1 = (\emptyset, \emptyset, \emptyset)$ is the unit element.

A trace $r \in \mathbb{R}$ is a prefix of a trace $t = [V, \le, \lambda] \in \mathbb{R}$, denoted by $r \le t$, if $r = [U, \le, \lambda]$ for some lower set U of V (for all $x \in V$ and $y \in U$, $x \le y$ implies $x \in U$). Equivalently, r is a prefix of t if $t = r \cdot s$ for some trace $s \in \mathbb{R}$. If $r \le t$ then we denote by $r^{-1}t$ the unique trace s such that $t = r \cdot s$.

If $A \subseteq \Sigma$, we let $\mathbb{R}_A = \{x \in \mathbb{R} \mid \mathrm{alph}(x) \subseteq A\}$ and $\mathbb{M}_A = \mathbb{M} \cap \mathbb{R}_A$. We also define $\mathbb{M}^+ = \mathbb{M} \setminus \{1\}$ and $\mathbb{M}_A^+ = \mathbb{M}^+ \cap \mathbb{M}_A$. We use other intuitive notations to denote sets of traces that are defined by alphabetic conditions. For instance,

$$\begin{aligned}
(a \in \min) &= \{t \in \mathbb{R} \mid a \in \min(t)\},\\
(\max \subseteq A) &= \{t \in \mathbb{M} \mid \max(t) \subseteq A\},\\
(\mathrm{alphinf} \subseteq A) &= \{t \in \mathbb{R} \mid \mathrm{alphinf}(t) \subseteq A\}.
\end{aligned}$$

We say that a trace $t \in \mathbb{R}(\Sigma, D)$ is *connected*, if the restriction of the graph (Σ, D) to $\mathrm{alph}(t)$ is connected. A finite trace $t \in \mathbb{M}(\Sigma, D)$ is *primitive*, if $t = x^n$ implies $n = 1$. The following result transfers from words to traces. It will be used in the proof of Proposition 1 below.

Lemma 1. *Let $t \in \mathbb{M}(\Sigma, D)$ be connected and primitive. Then, $rts \in t^*$ implies $r \in t^*$.*

Proof. Assume that $rts = t^n$ for some finite traces $r, t, s \in \mathbb{M}$ with t connected and primitive. If $r = 1$ or $s = 1$, the result is clear. Let $a, b \in \mathrm{alph}(t)$ be such that $(a, b) \in D$. Let u, v, w be the projections of r, s, t over $\{a, b\}^*$. We have $uwv = w^n$. It is well-known [20, Sect. 1.3] that an equation $uwv = w^n$ admits only cyclic solutions over words, i.e., for some $x \neq 1$ we have $u, v, w \in x^*$. We have $\{a, b\} = \mathrm{alph}(w) = \mathrm{alph}(x)$. Hence, $a \in \mathrm{alph}(r)$ if and only if $b \in \mathrm{alph}(r)$ and the same holds for s. Since t is connected and $u \neq 1 \neq v$, we deduce that $\mathrm{alph}(u) = \mathrm{alph}(v) = \mathrm{alph}(t)$. Since we have this alphabetic condition and since the equation $rts = t^n$ has only cyclic solutions over words, the same holds over traces by [4, Prop. 3.2.5]. Thus, we have $r, t, s \in y^*$ for some nonempty trace y. But then $t = y$ since t is primitive. $\square$

The main result of the paper concerns dependence alphabets which are cographs. According to standard graph theoretical notions, a dependence alphabet is called a *cograph*, if it belongs to the smallest class of undirected graphs which contains singletons and which is closed under the operations of disjoint union and complementation. Clearly, (Σ, D) is a cograph if and only if the *independence alphabet* (Σ, I) is a cograph. Cographs form an important and well-studied class of undirected graphs. A connected non-singleton cograph has a canonical decomposition as complex product of smaller cographs. On the algebraic side this corresponds to monoids which are obtained by direct and free products starting with free monoids over a singleton alphabet. Thus, cographs (and cograph monoids) are built in a strictly modular way. It is also well-known that an undirected graph is a cograph if and only if it does not contain any P_4 (a line of 4 vertices) as an induced subgraph [3, Thm. 11.3.3]. It turns out that (Σ, D) is a cograph if and only if all $t \in \mathbb{M}(\Sigma, D)$ are *series parallel posets*, i.e., we can build up the trace starting with letters by taking serial and parallel products. In particular, every such trace is N–free, i.e., whenever there are four vertices a, b, c, d with $a < b$, $c < b$, and $c < d$, then there is at least one more ordering between them.

Consider the following two traces t_1 and t_2 (given by their Hasse diagrams). The first trace is not a series parallel poset, whereas the second one is due to the additional dependency between a and d.

$$t_1 = \left(\begin{tikzcd} a \arrow[r] & b \arrow[dr] & \\ c \arrow[r] \arrow[ur] & d \arrow[r] & c \end{tikzcd}\right) \qquad t_2 = \left(\begin{tikzcd} a \arrow[r] \arrow[dr] & b \arrow[dr] & \\ c \arrow[r] \arrow[ur] & d \arrow[r] & c \end{tikzcd}\right)$$

We shall use the algebraic notion for recognizability: Let $h : \mathbb{M} \to S$ be a morphism to some finite monoid S. For $x, y \in \mathbb{R}$, we say that x and y are h-similar, denoted by $x \sim_h y$ if either $x, y \in \mathbb{M}$ and $h(x) = h(y)$ or x and y have infinite factorizations in nonempty finite traces $x = x_1 x_2 \cdots$, $y = y_1 y_2 \cdots$ with $x_i, y_i \in \mathbb{M}^+$ and $h(x_i) = h(y_i)$ for all i. We denote by $\approx_h$ the transitive closure of $\sim_h$ which is therefore an equivalence relation. Since S is finite, this equivalence relation is of finite index with at most $|S|^2 + |S|$ equivalence classes [26]. A real trace language $L \subseteq \mathbb{R}$ is *recognized* by h if it is saturated by $\sim_h$, i.e., $x \in L$ implies $[x]_{\approx_h} \subseteq L$ for all $x \in \mathbb{R}$.

Let $L \subseteq \mathbb{R}$ be recognized by a morphism $h : \mathbb{M} \to S$ and $A \subseteq \Sigma$. Then, $L \cap \mathbb{M}_A$ and $L \cap \mathbb{R}_A$ are recognized by the restriction $h_{|\mathbb{M}_A}$.

A finite monoid S is *aperiodic*, if there is some $n \geq 0$ such that $s^n = s^{n+1}$ for all $s \in S$. A real trace language $L \subseteq \mathbb{R}$ is *aperiodic* if it is recognized by some morphism to some finite and aperiodic monoid. We denote by $\mathrm{AP}(\Sigma, D)$ or simply by AP the set of aperiodic languages $L \subseteq \mathbb{R}(\Sigma, D)$. If $A \subseteq \Sigma$, we use the notation AP_A for the aperiodic languages over $\mathbb{R}_A$.

The first order theory of traces is given by the syntax of $\mathrm{FO}_\Sigma(<)$:

$$\varphi ::= P_a(x) \mid x < y \mid \neg\varphi \mid \varphi \vee \varphi \mid (\exists x)\varphi,$$

where $a \in \Sigma$ and x, y are first order variables. Given a trace $t = [V, \leq, \lambda]$ and a valuation σ of the free variables to the vertices, the semantics is obtained by interpreting the predicate $P_a(x)$ by $\lambda(\sigma(x)) = a$ and the relation $<$ as the strict partial order relation of the trace t. Then we can say whether or not $t, \sigma \models \varphi$. If φ is a sentence, i.e., a closed formula, then we simply write $t \models \varphi$ (since there are no free variables in a closed formula) and we define the language $\mathcal{L}(\varphi) = \{t \in \mathbb{R} \mid t \models \varphi\}$. We say that a trace language $L \subseteq \mathbb{R}$ is expressible in $\mathrm{FO}_\Sigma(<)$ if there exists some sentence $\varphi \in \mathrm{FO}_\Sigma(<)$ such that $L = \mathcal{L}(\varphi)$. We denote by $\mathrm{FO}_{(\Sigma,D)}(<)$ the set of real trace languages $L \subseteq \mathbb{R}(\Sigma, D)$ such that for some sentence $\varphi \in \mathrm{FO}_\Sigma(<)$ we have $L = \mathcal{L}(\varphi)$

We say that a first order formula is in $\mathrm{FO}^n_\Sigma(<)$ if it uses at most n first order variables (it may use each variable several times).

The following result states the equivalence between first-order definability and aperiodic languages. So, we can switch between both notions.

Theorem 1 ([11, 12]). *A language $L \subseteq \mathbb{R}(\Sigma, D)$ is expressible in $\mathrm{FO}_\Sigma(<)$ if and only if it is aperiodic, i.e., $\mathrm{FO}_{(\Sigma,D)}(<) = \mathrm{AP}(\Sigma, D)$.*

In Section 5, in order to prove that an existential until modality is not first order definable in general, we use the following characterization:

Proposition 1. *Let $1 \neq t \in \mathbb{M}(\Sigma, D)$ be a nonempty finite trace. The language $t^* = \{t^n \mid n \geq 0\}$ is expressible in $\mathrm{FO}_\Sigma(<)$ (or aperiodic) if and only if the trace t is connected and primitive.*

Proof. It is well-known and easy to see that if t is not connected then t^* is not recognizable. If $t \neq 1$ is connected but non-primitive, then t^* is recognizable but not aperiodic. Indeed, assume that $t = x^n$ with $n > 1$ and consider the morphism $g : \{a\}^* \to \mathbb{M}$ defined by $g(a) = x$. We have $g^{-1}(t^*) = (a^n)^*$ which is not aperiodic. Since aperiodic languages are closed under inverse morphisms, we deduce that t^* is not aperiodic.

For the other direction let $w = [V, \leq, \lambda]$ be a trace and let $(x_1, \ldots, x_m) \in V^m$ be a sequence of m vertices. The sequence defines a factor of w if and only if $x_i \leq y \leq x_j$ implies both $i \leq j$ and $y \in \{x_1, \ldots, x_m\}$ for all $y \in V$ and $1 \leq i, j \leq m$. Clearly, a factor $(x_1, \ldots, x_m) \in V^m$ leads to a factorization $w = utv$, where t is given by $\lambda(x_1) \cdots \lambda(x_m)$. Thus, for each finite trace $t \in \mathbb{M}$ of length m we can construct a first-order formula $F_t(x_1, \ldots, x_m)$ which becomes true if and only if the interpretation of $(x_1, \ldots, x_m)$ in V^m defines the factor t. In particular, $\mathbb{M}t\mathbb{M}$ is the language defined by $\exists x_1 \cdots \exists x_m\, F_t(x_1, \ldots, x_m)$. We may also define with a first order formula that all minimal vertices of V appear in the set $\{x_1, \ldots, x_m\}$.

Now let t be connected and primitive. We can express the language $t^+ \subseteq \mathbb{M}$ by some first-order formula as follows: We say that there exists some factor containing all minimal elements and which defines t. Moreover for all $(x_1, \ldots, x_m)$

where $\{x_1, \ldots, x_m\}$ does not contain all maximal elements we demand the existence of $(y_1, \ldots, y_m)$ such that the following implication holds:

$$F_t(x_1, \ldots, x_m) \Longrightarrow F_{t^2}(x_1, \ldots, x_m, y_1, \ldots, y_m).$$

By Lemma 1, we can show that each trace in t^+ satisfies this first-order formula. The converse is clear. $\square$

3 Local temporal logics for traces

In this section, we introduce the *local temporal logic* over traces and its semantics. We introduce both future and past modalities. Our expressive completeness results already hold for pure future logics. On the other hand, we can include past modalities when we prove that a fragment is in $\mathrm{FO}_\Sigma(<)$ or for the decision procedure.

In the next sections we specialize the logics by considering various subsets of the modalities. We say that a temporal logic over traces is *local* if it is evaluated at the vertices of the trace as for first order formulae. This is in contrast with global temporal logic formulae that are evaluated at global configurations of the trace, i.e., at finite prefixes of the trace.

For global formulae, we say that a trace is a model of a formula if it satisfies the formula at the empty configuration. There is no such canonical way to interpret a formula at some trace without fixing some vertex since there is no canonical vertex in the trace where to start the evaluation of the formula. Natural vertices are the minimal ones but a trace may have several minimal vertices. We have chosen to introduce initial formulae to address this problem. There are other possibilities, like adding a unique minimal dummy, but then the logic becomes more expressive and we are mainly interested in expressive completeness (with respect to first-order) for a weak fragment of our logic.

We start with the definition of (*local*) formulae that are evaluated at vertices. The syntax of $\mathrm{LocTL}_\Sigma(\mathsf{EX}, \mathsf{U}, \mathsf{EU}, \mathsf{EY}, \mathsf{S}, \mathsf{ES})$ is given by

$$\varphi ::= \bot \mid a \in \Sigma \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{EX}\,\varphi \mid \varphi \,\mathsf{U}\, \varphi \mid \varphi \,\mathsf{EU}\, \varphi. \mid \mathsf{EY}\,\varphi \mid \varphi \,\mathsf{S}\, \varphi \mid \varphi \,\mathsf{ES}\, \varphi.$$

The symbol $\bot$ means *false*, $\mathsf{EX}\,\varphi$ claims that φ holds for some immediate successor of the current vertex; $\varphi \,\mathsf{U}\, \psi$ is a *universal* until claiming that ψ holds for some vertex above the current one and that φ holds for all vertices in between; on the contrary, $\varphi \,\mathsf{EU}\, \psi$ is an *existential* until claiming the existence of some path in the Hasse diagram of the trace starting at the current vertex and where φ holds until ψ does. The interpretation is similar for the past modalities. Above, we have included some operators for comparison with other logics used in the literature such as TLC [2]. The focus and our completeness result is upon the basic pure future operators EX and U. For words this is just the standard logic LTL.

Formally, the semantics is inductively given as follows. Let $t = [V, \leq, \lambda] \in \mathbb{R}$ be a real trace and let $x \in V$ be a vertex (we also write $x \in t$).

$$
\begin{array}{ll}
t, x \models a & \text{if } \lambda(x) = a \\
t, x \models \neg\varphi & \text{if } t, x \not\models \varphi \\
t, x \models \varphi \vee \psi & \text{if } t, x \models \varphi \text{ or } t, x \models \psi \\
t, x \models \mathsf{EX}\, \varphi & \text{if } \exists y,\ x \lessdot y \text{ and } t, y \models \varphi \\
t, x \models \varphi \,\mathsf{U}\, \psi & \text{if } \exists z,\ x \leq z \text{ and } t, z \models \psi \text{ and } t, y \models \varphi, \forall x \leq y < z \\
t, x \models \varphi \,\mathsf{EU}\, \psi & \text{if } \exists x = y_0 \lessdot \cdots \lessdot y_n,\ \text{with } t, y_n \models \psi \text{ and } t, y_i \models \varphi, \forall 0 \leq i < n \\
t, x \models \mathsf{EY}\, \varphi & \text{if } \exists y,\ y \lessdot x \text{ and } t, y \models \varphi \\
t, x \models \varphi \,\mathsf{S}\, \psi & \text{if } \exists z,\ z \leq x \text{ and } t, z \models \psi \text{ and } t, y \models \varphi, \forall z < y \leq x. \\
t, x \models \varphi \,\mathsf{ES}\, \psi & \text{if } \exists y_n \lessdot \cdots \lessdot y_0 = x,\ \text{with } t, y_n \models \psi \text{ and } t, y_i \models \varphi, \forall 0 \leq i < n
\end{array}
$$

We define $\top = \neg\bot$, hence $\top$ means *true*. We derive some more operators from the above ones. Eventually φ claims the existence of some vertex where φ holds above the current one: $\mathsf{F}\, \varphi = \top \,\mathsf{U}\, \varphi = \top \,\mathsf{EU}\, \varphi$. Its dual operator, always φ, means that φ holds at all positions above the current one: $\mathsf{G}\, \varphi = \neg\, \mathsf{F}\, \neg\varphi$.

The initial formulae $\mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\cdots)$ are defined by the syntax

$$\alpha ::= \bot \mid \mathsf{EM}\, \varphi \mid \neg\alpha \mid \alpha \vee \alpha$$

where $\varphi \in \mathrm{LocTL}_{\Sigma}(\cdots)$. Intuitively, $\mathsf{EM}\, \varphi$ means that φ holds at some minimal vertex. Formally, the semantics is given by

$$
\begin{array}{ll}
t \models \mathsf{EM}\, \varphi & \text{if } \exists x \in \min(t) \text{ with } t, x \models \varphi \\
t \models \neg\alpha & \text{if } t \not\models \alpha \\
t \models \alpha \vee \beta & \text{if } t \models \alpha \text{ or } t \models \beta
\end{array}
$$

The dual $\mathsf{AM}\, \varphi = \neg\, \mathsf{EM}\, \neg\varphi$ means that φ holds for all minimal vertices. The following identity holds for all $\varphi \in \mathrm{LocTL}_{\Sigma}(\cdots)$:

$$\neg\, \mathsf{AM}\, \varphi \quad = \quad \bigvee_{a \in \Sigma} (\mathsf{EM}\, a \wedge \mathsf{AM}(\neg a \vee \neg\varphi))$$

Therefore, each initial formula is equivalent to a positive boolean combination of formulae of the form $\mathsf{EM}\, a$ or $\mathsf{AM}\, \varphi$. Since $\mathsf{AM}\, \varphi \wedge \mathsf{AM}\, \psi = \mathsf{AM}(\varphi \wedge \psi)$, we deduce that each initial formula is equivalent to a finite disjunction of formulae of the form $\mathsf{AM}\, \varphi \wedge \bigwedge_{a \in A} \mathsf{EM}\, a$ with $\varphi \in \mathrm{LocTL}_{\Sigma}(\cdots)$ and $A \subseteq \Sigma$.

An initial formula $\alpha \in \mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\cdots)$ defines the language $\mathcal{L}(\alpha) = \{t \in \mathbb{R} \mid t \models \alpha\}$ and we say that a trace language $L \subseteq \mathbb{R}$ is expressible in $\mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\cdots)$ if there exists an initial formula $\alpha \in \mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\cdots)$ such that $L = \mathcal{L}(\alpha)$. We denote by $\mathrm{LocTL}_{(\Sigma,D)}^{\mathrm{init}}(\cdots)$ the set of languages over $\mathbb{R}(\Sigma, D)$ that are expressible by some local temporal formula using the modalities $(\cdots)$.

With local temporal formulae, we can express various alphabetic properties.

$$
\begin{array}{lcl}
(a \in \min) & = & \mathcal{L}(\mathsf{EM}\, a), \\
(a \in \mathrm{alph}) & = & \mathcal{L}(\mathsf{EM}\, \mathsf{F}\, a), \\
(a \in \mathrm{alphinf}) & = & \mathcal{L}(\mathsf{EM}(\mathsf{F}\, a \wedge \mathsf{G}(a \Rightarrow \mathsf{EX}\, \mathsf{F}\, a))).
\end{array}
$$

To make formulae more intuitive, we use notations like ($\min \subseteq A$), ($\mathrm{alphinf} \subseteq A$) also for formulae defining these languages. For instance, ($\min \subseteq A$) $= \mathsf{AM}\,A$ where $A = \bigvee_{a \in A} a$. Also, ($\mathrm{alphinf} \subseteq A$) $= \bigwedge_{a \notin A} \neg\, \mathsf{EM}(\mathsf{F}\, a \wedge \mathsf{G}(a \Rightarrow \mathsf{EX}\,\mathsf{F}\, a))$.

It is clear from the semantics of $\mathsf{EX}, \mathsf{U}, \mathsf{EY}$ and S that all trace languages expressible in $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U}, \mathsf{EY}, \mathsf{S})$ are also expressible in $\mathrm{FO}_\Sigma(<)$, and even in $\mathrm{FO}^3_\Sigma(<)$. This is however not true in general for the existential versions of until and since (see Section 5).

We say that a formula $\varphi \in \mathrm{LocTL}_\Sigma(\cdots)$ is pure future if for all $t = t_1 t_2 \in \mathbb{R}$ and all $x \in t_2$, $t, x \models \varphi$ if and only if $t_2, x \models \varphi$. It is easy to see that all formulae in $\mathrm{LocTL}_\Sigma(\mathsf{EX}, \mathsf{U}, \mathsf{EU})$ are pure future.

Note that if two traces t_1 and t_2 have the same minimal letters and for all minimal letters the corresponding upper sets in t_1 and t_2 are the same, then the two traces cannot be distinguished by any formula from a future local temporal logic (e.g. $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U}, \mathsf{EU})$). Consider the following example which was first used by Walukiewicz [33] in relation with the μ-calculus. The dependence alphabet is $(\Sigma, D) = a — b — c — d$ and we let

$$t_1 = \left(\begin{tikzcd} a \arrow[r] & b \arrow[r] & c \arrow[r] & b \arrow[r] & c & \cdots \\ d \arrow[r] & c \arrow[u] \end{tikzcd}\right) \qquad t_2 = \left(\begin{tikzcd} a \arrow[r] & b \arrow[d] \\ d \arrow[r] & c \arrow[r] & b \arrow[r] & c \arrow[r] & b & \cdots \end{tikzcd}\right)$$

Since t_1 and t_2 are clearly distinguishable in $\mathrm{FO}_\Sigma(<)$, we deduce that a pure future local temporal logic cannot be expressively complete for $\mathrm{FO}_\Sigma(<)$ as soon as (Σ, D) is not a cograph. Note that it is easy to distinguish t_1 from t_2 if we allow some past tense modalities. For instance, the formula $\mathsf{EM}(a \,\mathsf{U}\, (b \wedge \mathsf{EY}\, c))$ is satisfied by t_1 but not by t_2.

Another possibility used in [5] is to introduce an additional minimal vertex (labeled by some special character) and to evaluate a formula at this new minimal vertex[3]. This amounts to consider a new letter $\# \notin \Sigma$ which depends on all letters in Σ. Then, we say that a trace $t \in \mathbb{R}(\Sigma, D)$ satisfies a local formula $\varphi \in \mathrm{LocTL}_\Sigma(\cdots)$ if φ holds at the initial vertex of the trace $\# \cdot t$. With this definition, the formula $(\neg c)\, \mathsf{U}\, b$ is satisfied by t_2 but not by t_1.

$$\# t_1 = \left(\begin{tikzcd} & a \arrow[r] & b \arrow[r] & c \arrow[r] & b & \cdots \\ \# \arrow[ur] \arrow[dr] \\ & d \arrow[r] & c \arrow[uu] \end{tikzcd}\right) \quad \# t_2 = \left(\begin{tikzcd} & a \arrow[r] & b \arrow[dd] \\ \# \arrow[ur] \arrow[dr] \\ & d \arrow[r] & c \arrow[r] & b \arrow[r] & c & \cdots \end{tikzcd}\right)$$

The same results hold for finite trace languages. For instance, the language $ad(bc)^+ \subseteq \mathbb{M}$ of finite traces is first order by Proposition 1. One can also see this directly with the star-free expression

$$(bc)^+ = (b\Sigma^* \cap \Sigma^* c) \setminus (\Sigma^* \{a, d, bb, cc\} \Sigma^*).$$

But it cannot be expressed in $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U}, \mathsf{EU})$. Indeed, assume $ad(bc)^+ = \mathcal{L}(\alpha)$ for some $\alpha \in \mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U}, \mathsf{EU})$. We have seen that α can be written as a finite union of formulae $\mathsf{AM}\,\varphi_A \wedge (\bigwedge_{a \in A} \mathsf{EM}\, a)$ for $A \subseteq \Sigma$ and $\varphi_A \in$

[3] This is also the approach we have taken later in [9]

$\mathrm{LocTL}_\Sigma(\mathsf{EX},\mathsf{U},\mathsf{EU})$. The languages $L_A = dc(bc)^* \cap \mathcal{L}(\mathsf{EM}\,\varphi_A)$ for $A \subseteq \Sigma$ are aperiodic (we restrict to $dc(bc)^*$ so that the modality EU is equivalent to the modality U and is therefore first-order). Hence, we find some integer N such that for all $u, v \in \mathbb{M}$ and $n > N$ we have for all $A \subseteq \Sigma$, $uv^{n-1} \in L_A$ if and only if $uv^n \in L_A$. Now, fix $n > N$ and consider the traces $t_1 = adc(bc)^n$ and $t_2 = ad(bc)^n$. We have $t_2 \models \alpha$, hence $t_2 \models \mathsf{AM}\,\varphi_A$ for some $A \subseteq \Sigma$. Since the logic is pure future, we deduce that $a(bc)^n \models \mathsf{EM}\,\varphi_A$ and $dc(bc)^{n-1} \models \mathsf{EM}\,\varphi_A$. Using the aperiodicity of L_A we get $dc(bc)^n \models \mathsf{EM}\,\varphi_A$ and we deduce that $t_1 \models \mathsf{AM}\,\varphi_A$. This is a contradiction since $t_1 \notin ad(bc)^+$.

In this paper, we are interested in the expressive completeness of pure future local temporal logics. The simple example above shows that we can restrict our study to traces defined by a dependence alphabet that are cographs.

The following lemma shows that we can *localize* initial formulae.

Lemma 2. *Let* $\alpha \in \mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX},\mathsf{U})$ *be an initial formula. There exists a local formula* $\mathrm{loc}(\alpha) \in \mathrm{LocTL}_\Sigma(\mathsf{EX},\mathsf{U})$ *such that for all* $t = t_1 t_2 \in \mathbb{R}$ *and* $x \in \max(t_1)$ *with* $\min(t_2) \subseteq D(\lambda(x))$, *we have* $t_2 \models \alpha$ *if and only if* $t, x \models \mathrm{loc}(\alpha)$. *The same holds for the fragment* $\mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX},\mathsf{U},\mathsf{EU})$.

Proof. Clearly, we have $\mathrm{loc}(\alpha \vee \beta) = \mathrm{loc}(\alpha) \vee \mathrm{loc}(\beta)$ and $\mathrm{loc}(\neg\alpha) = \neg\mathrm{loc}(\alpha)$. The interesting case is $\mathsf{EM}\,\varphi$ where $\varphi \in \mathrm{LocTL}$. We have,

$$\begin{array}{llll} t_2 \models \mathsf{EM}\,\varphi & \text{iff } \exists y \in \min(t_2),\ t_2, y \models \varphi & \\ & \text{iff } \exists y \in \min(t_2),\ t, y \models \varphi & \text{since the logic is pure future} \\ & \text{iff } \exists y,\ x \lessdot y \text{ and } t, y \models \varphi & \text{using the hypothesis on } x \\ & \text{iff } t, x \models \mathsf{EX}\,\varphi. & \end{array}$$

Therefore, $\mathrm{loc}(\mathsf{EM}\,\varphi) = \mathsf{EX}\,\varphi$. ◻

4 Universal until

In this section, we consider the fragment of the local temporal logic using next and the *universal* until, only. We give the following characterization of the expressive completeness of $\mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX},\mathsf{U})$ with respect to $\mathrm{FO}_\Sigma(<)$.

Theorem 2. *Let* (Σ, D) *be a dependence alphabet. Then we have the equality* $\mathrm{LocTL}^{\mathrm{init}}_{(\Sigma,D)}(\mathsf{EX},\mathsf{U}) = \mathrm{FO}_{(\Sigma,D)}(<)$ *if and only if* (Σ, D) *is a cograph.*

We have seen in the previous section that $\mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX},\mathsf{U})$ is not expressively complete, if (Σ, D) is not a cograph. Conversely, for all dependence alphabets, we already know that $\mathrm{LocTL}^{\mathrm{init}}_{(\Sigma,D)}(\mathsf{EX},\mathsf{U}) \subseteq \mathrm{FO}^3_{(\Sigma,D)}(<)$ and that first order languages coincide with aperiodic languages (Theorem 1). Hence, in order to get the converse inclusion, we will prove that $\mathrm{AP} \subseteq \mathrm{LocTL}^{\mathrm{init}}_{(\Sigma,D)}(\mathsf{EX},\mathsf{U})$, if (Σ, D) is a cograph.

For the proof of Theorem 2, we use an induction on $|\Sigma|$. If (Σ, D) is a cograph, then either Σ is a singleton, or Σ is the disjoint union of two nonempty sets $\Sigma = A \cup B$ with either $A \times B \subseteq I$ or $A \times B \subseteq D$. We consider these three cases in turn.

The base case is when $\Sigma = \{a\}$. Then, an aperiodic language L is either a finite set or the union of a finite set and a set of the form $a^n a^*$, $n \geq 0$. In both cases, L is expressible in $\mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. For instance, $\{a^\omega\}$ and $\{a^2\}$ correspond to the formulae $\mathsf{EM}\,\mathsf{G}\,\mathsf{EX}\top$ and $\mathsf{EM}\,\mathsf{EX}\neg\,\mathsf{EX}\top$. Also, $a^3a^* \cup \{a^\omega\}$ is expressed by $\mathsf{EM}\,\mathsf{EX}\,\mathsf{EX}\top$.

Next, we consider the case where the alphabet is the disjoint union of two independent subsets. This means $\mathbb{R}(\Sigma, D) = \mathbb{R}_A \times \mathbb{R}_B$ is a direct product.

Proposition 2. *Assume that* $\Sigma = A \cup B$ *with* $A \times B \subseteq I$ *and that we have both* $\mathrm{AP}_A \subseteq \mathrm{LocTL}_{(A, D\cap A\times A)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$ *and* $\mathrm{AP}_B \subseteq \mathrm{LocTL}_{(B, D\cap B\times B)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. *Then it also holds* $\mathrm{AP}_\Sigma \subseteq \mathrm{LocTL}_{(\Sigma, D)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$.

Proof. Let $L \subseteq \mathbb{R}$ be recognized by some morphism h from $\mathbb{M}$ to some finite aperiodic monoid S. We claim that L is a finite union of languages of the form $(L_1 \cap \mathbb{R}_A) \cdot (L_2 \cap \mathbb{R}_B)$ where the languages $L_i \subseteq \mathbb{R}$ are recognized by h.

Indeed, let $x \in \mathbb{R}_A$, $y \in \mathbb{R}_B$ with $xy \in L$. We prove that $([x]_{\approx_h} \cap \mathbb{R}_A) \cdot ([y]_{\approx_h} \cap \mathbb{R}_B) \subseteq L$. Note that the claim follows from this fact since there are only finitely many $\approx_h$-classes.

let $x' \in \mathbb{R}_A$ and $y' \in \mathbb{R}_B$ with $x \sim_h x'$ and $y \sim_h y'$. The cases x or y finite are simpler, hence we assume that both x and y are infinite. We have $x = x_1x_2\cdots$, $x' = x'_1x'_2\cdots$, $y = y_1y_2\cdots$ and $y' = y'_1y'_2\cdots$ with $h(x_i) = h(x'_i)$ and $h(y_i) = h(y'_i)$ for all $i \geq 1$. Then, $x'y' = (x'_1y'_1)(x'_2y'_2)\cdots \sim_h (x_1y_1)(x_2y_2)\cdots = xy \in L$. Since L is recognized by h, we deduce that $x'y' \in L$ which concludes the proof of the claim.

Now, $(L_1 \cap \mathbb{R}_A) \cdot (L_2 \cap \mathbb{R}_B) = \big((L_1 \cap \mathbb{R}_A) \cdot \mathbb{R}_B\big) \cap \big(\mathbb{R}_A \cdot (L_2 \cap \mathbb{R}_B)\big)$. Therefore, it remains to show that $(L_1 \cap \mathbb{R}_A) \cdot \mathbb{R}_B$ is expressible in $\mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. Since L_1 is recognized by h, the language $L_1 \cap \mathbb{R}_A$ is recognized by the restriction $h_{|\mathbb{M}_A}$. Hence, $L_1 \cap \mathbb{R}_A \in \mathrm{AP}_A \subseteq \mathrm{LocTL}_{(A, D\cap A\times A)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$.

One can check by structural induction on the formula $\varphi \in \mathrm{LocTL}_A(\mathsf{EX}, \mathsf{U})$ that for all $t_1 \in \mathbb{R}_A$, $x \in t_1$ and $t_2 \in \mathbb{R}_B$, we have $t_1, x \models \varphi$ if and only if $t_1t_2, x \models \varphi$. Moreover, $t_1 \models \mathsf{EM}\,\varphi$ if and only if $t_1t_2 \models \mathsf{EM}(A \wedge \varphi)$ (recall that A stands for the formula $\bigvee_{a \in A} a$). Therefore, $(L_1 \cap \mathbb{R}_A) \cdot \mathbb{R}_B$ is expressible in $\mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. $\square$

The last case and the most interesting one is when the dependence alphabet is the disjoint union of two fully dependent subsets: $\Sigma = A \cup B$ with $A \cap B = \emptyset$ and $A \times B \subseteq D$. In this case, the trace monoid $\mathbb{M}$ is the free product of the monoids $\mathbb{M}_A$ and $\mathbb{M}_B$.

Proposition 3. *Assume that* $\Sigma = A \cup B$ *with* $A \cap B = \emptyset$ *and* $A \times B \subseteq D$. *If we have* $\mathrm{AP}_A \subseteq \mathrm{LocTL}_{(A, D\cap A\times A)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$ *and* $\mathrm{AP}_B \subseteq \mathrm{LocTL}_{(B, D\cap B\times B)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$, *then* $\mathrm{AP}_\Sigma \subseteq \mathrm{LocTL}_{(\Sigma, D)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$.

Our proof is inspired by a technique introduced by Wilke [34] in order to show that aperiodic languages over words are expressible in LTL. Wilke used both an induction on the size of the alphabet and on the size of the recognizing semigroup. Here, we have chosen to skip the additional induction on the size of the semigroup needed to get a direct proof. Instead, we appeal to Kamp's Theorem in Lemma 6. This allows to keep our proof as simple as possible.

We use several splittings of languages in products. We shall use the following composition lemmas to get the expressibility of the products from the expressibility of their components. Note that, since aperiodic languages are closed under product, a consequence of our theorem is that the product of two languages expressible in local temporal logic is again expressible. But we are not able to prove this result directly and the following lemmas correspond to three very special cases where an easy proof can be given. We assume until the end of this section that $\Sigma = A \cup B$ with $A \cap B = \emptyset$ and $A \times B \subseteq D$.

Lemma 3. *Let* $L \subseteq \mathbb{R}_A$ *be a language expressible in* $\mathrm{LocTL}_A^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$*. Then, the language* $(L \cap \mathbb{M}_A^+) \cdot (\min \subseteq B)$ *is expressible in* $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$*.*

Proof. Note that $\mathbb{M}_A^+ \cdot (\min \subseteq B) = \mathcal{L}(\mathsf{EM}\, A)$ is expressible in $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. Hence, Lemma 3 follows from the

Claim: For all $\alpha \in \mathrm{LocTL}_A^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$ and $\varphi \in \mathrm{LocTL}_A(\mathsf{EX}, \mathsf{U})$, there exist $\widetilde{\alpha} \in \mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$ and $\widetilde{\varphi} \in \mathrm{LocTL}_\Sigma(\mathsf{EX}, \mathsf{U})$ such that for all $t = t_1 t_2$ with $t_1 \in \mathbb{M}_A$ and $\min(t_2) \subseteq B$ and all $x \in t_1$, we have $t_1 \models \alpha$ if and only if $t \models \widetilde{\alpha}$ and $t_1, x \models \varphi$ if and only if $t, x \models \widetilde{\varphi}$.

We proceed by structural induction on the formulae. As usual, the cases for disjunction and negation are trivial. Finally, it is easy to verify that $\widetilde{\mathsf{EM}\,\varphi} = \mathsf{EM}(A \wedge \varphi)$, $\widetilde{a} = a$ for $a \in A$, $\widetilde{\mathsf{EX}\,\varphi} = \mathsf{EX}(A \wedge \widetilde{\varphi})$ and $\widetilde{\varphi \,\mathsf{U}\, \psi} = (A \wedge \widetilde{\varphi}) \,\mathsf{U}\, (A \wedge \widetilde{\psi})$. We use A in the formulae above to insure that we stay inside t_1, even when evaluating the formulae in t. □

Lemma 4. *Let* $\alpha \in \mathrm{LocTL}_A^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$*. The language* $(\max \subseteq B) \cdot \mathcal{L}_A(\alpha)$ *is definable in* $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$ *by the formula*

$$\big(\mathbb{R}_A \wedge \alpha\big) \vee \mathsf{EM}\,\mathsf{F}\big(B \wedge \neg\,\mathsf{EX}\,\mathsf{F}\,B \wedge \mathrm{loc}(\alpha)\big).$$

Proof. Note that $(\max \subseteq B) \cdot \mathbb{R}_A = (\mathrm{alphinf} \subseteq A)$. Let $t = t_1 t_2$ with $\max(t_1) \subseteq B$ and $t_2 \in \mathcal{L}_A(\alpha)$. If $t_1 = 1$ then $t = t_2 \in \mathbb{R}_A$ and $t = t_2 \in \mathcal{L}_\Sigma(\alpha)$. Now, assume that $t_1 \neq 1$ and let $y \in \max(t_1)$. Using Lemma 2 we deduce that $t, y \models B \wedge \neg\,\mathsf{EX}\,\mathsf{F}\,B \wedge \mathrm{loc}(\alpha)$ and then, $t \models \mathsf{EM}\,\mathsf{F}(B \wedge \neg\,\mathsf{EX}\,\mathsf{F}\,B \wedge \mathrm{loc}(\alpha))$.

Conversely, the case $t \in \mathbb{R}_A$ and $t \in \mathcal{L}_\Sigma(\alpha)$ is simple. Next, assume that $t, x \models \mathsf{F}(B \wedge \neg\,\mathsf{EX}\,\mathsf{F}\,B \wedge \mathrm{loc}(\alpha))$ for some $x \in \min(t)$. Then, $t = t_1 t_2$ with $\max(t_1) \subseteq B$ and $t_2 \in \mathbb{R}_A$. Let $y \geq x$ be such that $t, y \models B \wedge \neg\,\mathsf{EX}\,\mathsf{F}\,B \wedge \mathrm{loc}(\alpha)$, we deduce that $y \in \max(t_1)$ and from Lemma 2 we obtain $t_2 \models \alpha$. □

Lemma 5. *Let* $L \subseteq \mathbb{R}$ *be a language expressible in* $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$*. Then, the language* $(L \cap (\max \subseteq B)) \cdot \mathbb{R}_A$ *is expressible in* $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$*.*

Proof. Note first that $(\max \subseteq B) \cdot \mathbb{R}_A = (\mathrm{alphinf} \subseteq A)$ is expressible in $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. Hence, Lemma 5 follows from the

Claim: For all $\alpha \in \mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX},\mathsf{U})$ and all $\varphi \in \mathrm{LocTL}_\Sigma(\mathsf{EX},\mathsf{U})$, there exist $\widetilde{\alpha} \in \mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX},\mathsf{U})$ and $\widetilde{\varphi} \in \mathrm{LocTL}_\Sigma(\mathsf{EX},\mathsf{U})$ such that for all $t = t_1 t_2$ with $\max(t_1) \subseteq B$ and $t_2 \in \mathbb{R}_A$ and all $x \in t_1$, we have $t_1 \models \alpha$ if and only if $t \models \widetilde{\alpha}$ and $t_1, x \models \varphi$ if and only if $t, x \models \widetilde{\varphi}$. We proceed by structural induction on the formulae. As usual, the cases for disjunction and negation are trivial. We have $\widetilde{a} = a$ for $a \in \Sigma$. Finally, it is easy to verify that $\widetilde{\mathsf{EM}\,\varphi} = \mathsf{EM}(\widetilde{\varphi} \wedge \mathsf{F}\,B)$, $\widetilde{\mathsf{EX}\,\varphi} = \mathsf{EX}(\widetilde{\varphi} \wedge \mathsf{F}\,B)$ and $\widetilde{\varphi\,\mathsf{U}\,\psi} = \widetilde{\varphi}\,\mathsf{U}\,(\widetilde{\psi} \wedge \mathsf{F}\,B)$. We use $\mathsf{F}\,B$ in the formulae above to insure that we stay inside t_1, even when evaluating the formulae in t. □

We consider the set $\Delta = \mathbb{M} \cup ((\mathrm{alphinf} \not\subseteq A) \cap (\mathrm{alphinf} \not\subseteq B))$. Note that Δ is expressible in $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX},\mathsf{U})$ since it is characterized by alphabetic information. Each trace $x \in \Delta$ has a unique (finite or infinite) factorization $x = x_1 x_2 \cdots$ in nonempty finite traces alternating $\mathbb{M}_A^+$ and $\mathbb{M}_B^+$.

Let $h : \mathbb{M} \to S$ be a morphism to some finite aperiodic monoid S. Let $T = h(\mathbb{M}^+)$ and let $e : T^* \to S$ be the evaluation morphism. Using the unique factorization of elements $x \in \Delta$, we can define a mapping $\sigma : \Delta \to T^\infty$ by $\sigma(x) = h(x_1)h(x_2)\cdots$. This mapping σ allows to reduce our problem to words over the alphabet T.

Recall that the syntax of the linear temporal logics $\mathrm{LTL}_T(\mathsf{XU})$ on words over the alphabet T is given by

$$f ::= \bot \mid s \in T \mid \neg f \mid f \vee f \mid f\,\mathsf{XU}\,f.$$

The semantics is defined as follows. Let $t = t_0 t_1 \cdots \in T^\infty$ and let $0 \le i < |t|$. Then,

$$\begin{array}{ll} t,i \models s & \text{if } \ t_i = s \\ t,i \models \neg f & \text{if } \ t,i \not\models f \\ t,i \models f \vee g & \text{if } \ t,x \models f \text{ or } t,x \models g \\ t,i \models f\,\mathsf{XU}\,g & \text{if } \ \exists k,\ i < k < |t| \text{ and } t,k \models g \text{ and } t,j \models f, \forall i < j < k. \end{array}$$

A formula $f \in \mathrm{LTL}_T(\mathsf{XU})$ defines a language $\mathcal{L}(f) = \{t \in T^\infty \mid t,0 \models f\}$. In the next lemma, we use Kamp's Theorem on words. More precisely, we use the following equality $\mathrm{AP}_T = \mathrm{FO}_T(<) = \mathrm{LTL}_T(\mathsf{XU})$ ([18, 13, 22, 28]).

Lemma 6. *Let $L \subseteq \mathbb{R}$ be recognized by h. Then, $L \cap \Delta = \sigma^{-1}(K)$ for some language $K \in T^\infty$ expressible in $\mathrm{LTL}_T(\mathsf{XU})$.*

Proof. Let $K = [\sigma(L \cap \Delta)]_{\approx_e}$. By definition, K is recognized by the evaluation morphism e to the aperiodic monoid S. Hence K is an aperiodic word language over T. Since $\mathrm{AP}_T = \mathrm{FO}_T(<) = \mathrm{LTL}_T(\mathsf{XU})$, we deduce that K is expressible in $\mathrm{LTL}_T(\mathsf{XU})$. Therefore, it remains to show that $L \cap \Delta = \sigma^{-1}(K)$. The inclusion $L \cap \Delta \subseteq \sigma^{-1}(K)$ is clear.

For the converse inclusion, let $y \in \sigma^{-1}(K)$. There exists $x \in L \cap \Delta$ such that $\sigma(x) \approx_e \sigma(y)$. Note that x is finite if and only if $\sigma(x)$ is finite if and only if $\sigma(y)$ is finite if and only if y is finite and in this case $h(x) = e(\sigma(x)) = e(\sigma(y)) = h(y)$.

Therefore, $y \approx_h x \in L$ and we deduce $y \in L$. Assume now that x and y are both infinite. Using the following claim we also have $y \approx_h x$ and we deduce as above that $y \in L$.

Claim: Let $x = x_1 x_2 \cdots \in \mathbb{R}$ and $y = y_1 y_2 \cdots \in \mathbb{R}$ with $x_i, y_i \in \mathbb{M}^+$. If $h(x_1)h(x_2)\cdots \approx_e h(y_1)h(y_2)\cdots$, then $x \approx_h y$.

We have $\approx_e = \bigcup_{n>0} \sim_e^n$. We prove the claim by induction on n. For the base case $n = 1$ we have $h(x_1)h(x_2)\cdots \sim_e h(y_1)h(y_2)\cdots$. Then, there exist two sequences $0 = i_0 < i_1 < \cdots$ and $0 = j_0 < j_1 < \cdots$ such that $e(h(x_{1+i_k})\cdots h(x_{i_{k+1}})) = e(h(y_{1+j_k})\cdots h(y_{j_{k+1}}))$ for all $k \geq 0$. We deduce that $h(x_{1+i_k}\cdots x_{i_{k+1}}) = h(y_{1+j_k}\cdots y_{j_{k+1}})$ for all $k \geq 0$. Therefore, $x \sim_h y$.

For the induction step, assume that $h(x_1)h(x_2)\cdots \sim_e^n t \sim_e h(y_1)h(y_2)\cdots$ for some $t = t_1 t_2 \cdots \in T^\omega$. Since $T = h(\mathbb{M}^+)$, we find $z = z_1 z_2 \cdots \in \mathbb{R}$ with $z_i \in \mathbb{M}^+$ and $h(z_i) = t_i$ for all $i > 0$. By induction, we get $x \approx_h z$ and using the case $n = 1$ above we also have $z \sim_h y$. It follows $x \approx_h y$ which proves the claim.

Note that, even if $x, y \in \Delta$, the intermediary trace z may not be in Δ. This is why our claim has a stronger statement that the one that is actually used and which is simply $\sigma(x) \approx_e \sigma(y) \Rightarrow x \approx_h y$ for all $x, y \in \Delta$. $\square$

In order to make use of the previous lemma, we need to lift through σ^{-1} an LTL formula over T to some local temporal formula. This is the purpose of the next lemma.

Lemma 7. *Let $f \in \mathrm{LTL}_T(\mathsf{XU})$. Then, the trace language $\sigma^{-1}(\mathcal{L}(f))$ is expressible in $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$.*

Proof. We show by structural induction on the formula f that there exists a formula $\widetilde{f} \in \mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$ such that $\mathcal{L}(\widetilde{f}) \cap \Delta = \sigma^{-1}(\mathcal{L}(f))$. Since Δ is expressible in $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$, the lemma follows.

First, $\widetilde{\bot} = \bot$, $\widetilde{f \vee g} = \widetilde{f} \vee \widetilde{g}$, and $\widetilde{\neg f} = \neg \widetilde{f}$. Now, we prove that

$$\widetilde{f \mathbin{\mathsf{XU}} g} = \mathsf{EM}(\Phi \mathbin{\mathsf{U}} \Psi)$$

where

$$\begin{aligned}
\Phi &= (A \wedge \mathsf{EX}\, A) \vee (B \wedge \mathsf{EX}\, B) \vee \mathrm{loc}(\widetilde{f}) \\
\Psi &= ((A \wedge \mathsf{EX}\, B) \vee (B \wedge \mathsf{EX}\, A)) \wedge \mathrm{loc}(\widetilde{g}).
\end{aligned}$$

Let $t \in \sigma^{-1}(\mathcal{L}(f \mathsf{XU} g))$. then, $t \in \Delta$. Let $t = t_1 t_2 \cdots$ be its canonical factorization. There exists $j > 1$ such that $\sigma(t_j t_{j+1} \cdots) \models g$ and $\sigma(t_i t_{i+1} \cdots) \models f$ for all $1 < i < j$. By induction, we deduce that $t_j t_{j+1} \cdots \models \widetilde{g}$ and $t_i t_{i+1} \cdots \models \widetilde{f}$ for all $1 < i < j$.

Let $x \in \min(t_1)$, $z \in \max(t_{j-1})$ and $x \leq y < z$. Clearly, $t, z \models (A \wedge \mathsf{EX}\, B) \vee (B \wedge \mathsf{EX}\, A)$. Moreover from Lemma 2 we deduce that $t, z \models \mathrm{loc}(\widetilde{g})$. Also, either y is not maximal in any t_k and we have $t, y \models (A \wedge \mathsf{EX}\, A) \vee (B \wedge \mathsf{EX}\, B)$ or $y \in \max(t_k)$ for some $1 \leq k < j - 1$ and by Lemma 2 we deduce that $t, y \models \mathrm{loc}(\widetilde{f})$. Therefore, $t, x \models \Phi \mathbin{\mathsf{U}} \Psi$ and $t \models \mathsf{EM}(\Phi \mathbin{\mathsf{U}} \Psi)$.

Conversely, let $t = t_1 t_2 \cdots \in \Delta$ and $x \in \min(t)$ be such that $t, x \models \Phi \mathsf{U} \Psi$. Let $z \geq x$ be such that $t, z \models \Psi$. Then, there exists $j > 1$ such that $z \in \max(t_{j-1})$ and from Lemma 2, we deduce that $t_j t_{j+1} \cdots \models \widetilde{g}$ and by induction $\sigma(t_j t_{j+1} \cdots) \models g$. Now, for all $1 < i < j$, let $y \in \max(t_{i-1})$ with $x \leq y$. We also have $y < z$ and from $t, y \models \Phi$, we deduce that $t, y \models \mathrm{loc}(\widetilde{f})$. Using Lemma 2, we obtain $t_i t_{i+1} \cdots \models \widetilde{f}$ and by induction $\sigma(t_i t_{i+1} \cdots) \models f$. Therefore, $\sigma(t) \models f \; \mathsf{XU} \; g$.

It remains to deal with the case $f = s$ with $s \in T$. We have $\mathcal{L}(s) = sT^\infty$ and

$$\sigma^{-1}(sT^\infty) = \Delta \cap \Big((h^{-1}(s) \cap \mathbb{M}_A^+)(\min \subseteq B) \cup (h^{-1}(s) \cap \mathbb{M}_B^+)(\min \subseteq A) \Big).$$

The language $h^{-1}(s) \cap \mathbb{M}_A^+$ is recognized by $h_{|\mathbb{M}_A}$ and is therefore an aperiodic language over $\mathbb{R}_A$. Since we have assumed $\mathrm{AP}_A \subseteq \mathrm{LocTL}^{\mathrm{init}}_{(A, D \cap A \times A)}(\mathsf{EX}, \mathsf{U})$, the language $h^{-1}(s) \cap \mathbb{M}_A^+$ is expressible in $\mathrm{LocTL}^{\mathrm{init}}_A(\mathsf{EX}, \mathsf{U})$. Using Lemma 3, we deduce that $(h^{-1}(s) \cap \mathbb{M}_A^+)(\min \subseteq B)$ is expressible in $\mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX}, \mathsf{U})$. Similarly, $(h^{-1}(s) \cap \mathbb{M}_B^+)(\min \subseteq A)$ is expressible in $\mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX}, \mathsf{U})$ and we deduce that $\sigma^{-1}(sT^\infty)$ is expressible in $\mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX}, \mathsf{U})$. □

Now, we have all we need in hand to prove the main result of this section.

Proof of Proposition 3 Let $L \subseteq \mathbb{R}$ be recognized by the morphism h. Since $\mathbb{R} = \Delta \cup (\mathrm{alphinf} \subseteq A) \cup (\mathrm{alphinf} \subseteq B)$, we have

$$L = (L \cap \Delta) \cup (L \cap (\mathrm{alphinf} \subseteq A)) \cup (L \cap (\mathrm{alphinf} \subseteq B)).$$

From Lemmas 6 and 7, we deduce that $L \cap \Delta$ is expressible in $\mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX}, \mathsf{U})$.

We claim that $L \cap (\mathrm{alphinf} \subseteq A)$ is a finite union of products of the form $(L_1 \cap (\max \subseteq B)) \cdot (L_2 \cap \mathbb{R}_A)$ where the languages L_1, L_2 are recognized by h.

Let $x \in (\max \subseteq B)$ and $y \in \mathbb{R}_A$ with $xy \in L$. We prove that

$$([x]_{\approx_h} \cap (\max \subseteq B)) \cdot ([y]_{\approx_h} \cap \mathbb{R}_A) \subseteq L.$$

Note that the claim follows from this fact since there are only finitely many classes and $L \cap (\mathrm{alphinf} \subseteq A)$ is therefore the finite union of those products.

let $x' \in (\max \subseteq B)$ and $y' \in \mathbb{R}_A$ with $x \sim_h x'$ and $y \sim_h y'$. The case y finite is simpler, hence we assume y infinite. We have $y = y_1 y_2 \cdots$, $y' = y_1' y_2' \cdots$ with $h(y_i) = h(y_i')$ for all $i \geq 1$. Then, $x'y' = (x'y_1')y_2' \cdots \sim_h (xy_1)y_2 \cdots = xy \in L$. Since L is recognized by h, we deduce that $x'y' \in L$ which concludes the proof of the claim.

Since $(\max \subseteq B) \subseteq \mathbb{M} \subseteq \Delta$, we have $L_1 \cap (\max \subseteq B) = (L_1 \cap \Delta) \cap (\max \subseteq B)$. Now, L_1 is recognized by h and using Lemmas 6 and 7, we deduce that $L_1 \cap \Delta$ is expressible in $\mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX}, \mathsf{U})$. Finally, from Lemma 5 we deduce that $(L_1 \cap (\max \subseteq B)) \cdot \mathbb{R}_A$ is expressible in $\mathrm{LocTL}^{\mathrm{init}}_\Sigma(\mathsf{EX}, \mathsf{U})$.

Now, L_2 is recognized by h. Hence, $L_2 \cap \mathbb{R}_A$ is recognized by the restriction $h_{|\mathbb{M}_A}$ and is therefore an aperiodic language over $\mathbb{R}_A$. Since we have assumed $\mathrm{AP}_A \subseteq \mathrm{LocTL}^{\mathrm{init}}_{(A, D \cap A \times A)}(\mathsf{EX}, \mathsf{U})$, $L_2 \cap \mathbb{R}_A$ is also expressible in $\mathrm{LocTL}^{\mathrm{init}}_A(\mathsf{EX}, \mathsf{U})$.

Using Lemma 4, we deduce that $(\max \subseteq B) \cdot (L_2 \cap \mathbb{R}_A)$ is expressible in $\mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$.

Finally, the product $(\max \subseteq B) \cdot \mathbb{R}_A$ is unambiguous, hence we have

$$(L_1 \cap (\max \subseteq B)) \cdot (L_2 \cap \mathbb{R}_A) = (L_1 \cap (\max \subseteq B)) \cdot \mathbb{R}_A \cap (\max \subseteq B) \cdot (L_2 \cap \mathbb{R}_A).$$

We deduce that $(L_1 \cap (\max \subseteq B)) \cdot (L_2 \cap \mathbb{R}_A)$ is expressible in $\mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. Therefore, $L \cap (\mathrm{alphinf} \subseteq A)$ is expressible in $\mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. □

5 Existential until

In this section, we consider the fragment of the local temporal logic using next and the *existential* until only. We prove the following characterization.

Theorem 3. *Let (Σ, D) be a dependence alphabet. Then we have the equality $\mathrm{LocTL}_{(\Sigma,D)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{EU}) = \mathrm{FO}_{(\Sigma,D)}(<)$ if and only if (Σ, D) is a cograph.*

We have seen in Section 3 that $\mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\mathsf{EX}, \mathsf{EU})$ is not expressively complete, if (Σ, D) is not a cograph. Conversely, The proofs of Propositions 2 and 3 can be carried out with slight modifications using the existential until instead of the universal until. Therefore, we get $\mathrm{FO}_{(\Sigma,D)}(<) = \mathrm{AP} \subseteq \mathrm{LocTL}_{(\Sigma,D)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{EU})$ if (Σ, D) is a cograph. The difficulty with the existential until is that we do not get the converse inclusion for free as with the universal until. Indeed, the semantics of existential until is given by a monadic second order formula since it claims the existence of some path in the trace. This MSO formula can be expressed in first order, if (Σ, D) is a cograph. This is certainly not true for arbitrary dependence alphabet:

Proposition 4. *In general,* $\mathrm{LocTL}_{(\Sigma,D)}^{\mathrm{init}}(\mathsf{EU}) \not\subseteq \mathrm{FO}_{(\Sigma,D)}(<)$.

Proof. Let (Σ, D) be the dependence alphabet where $\Sigma = \{a, b, c, d, e, f, g, h\}$ and the dependence relation D is depicted in Figure 1. Let $t = bdegachf \in \mathbb{M}(\Sigma, D)$ as in Figure 1 and let $\varphi = (a \vee b \vee c \vee d) \;\mathsf{EU}\; \mathsf{G}\, b$ (Note that $\mathsf{G}\,\varphi = \neg(\top \;\mathsf{EU}\; \neg\varphi)$). Following Figure 2 is easy to verify that $aht^n b^\omega \models \mathsf{EM}\,\varphi$ if and only if n is even. Hence, we have $aht^* b^\omega \cap \mathcal{L}(\mathsf{EM}\,\varphi) = ah(t^2)^* b^\omega$. Since t is connected and primitive, by Proposition 1 the trace language t^* is aperiodic. Therefore, the language $aht^* b^\omega$ is aperiodic as well. Now, aperiodic languages are closed under arbitrary quotients, hence using again Proposition 1 we deduce that the language $ah(t^2)^* b^\omega$ is not aperiodic. Since aperiodic languages are closed under intersection, it follows that $\mathcal{L}(\mathsf{EM}\,\varphi)$ is not aperiodic. Therefore, $\mathrm{LocTL}_{(\Sigma,D)}^{\mathrm{init}}(\mathsf{EU}) \not\subseteq \mathrm{FO}_{(\Sigma,D)}(<)$ for the dependence alphabet of Figure 1.

Note that, by considering traces of the form $aht^* b$, we get the same result for finite traces. □

The key result for showing that $\mathrm{LocTL}_{(\Sigma,D)}^{\mathrm{init}}(\mathsf{EX}, \mathsf{EU}) \subseteq \mathrm{FO}_{(\Sigma,D)}(<)$, if the dependence alphabet is a cograph, is to express in $\mathrm{FO}_{\Sigma}(<)$ the existence of a

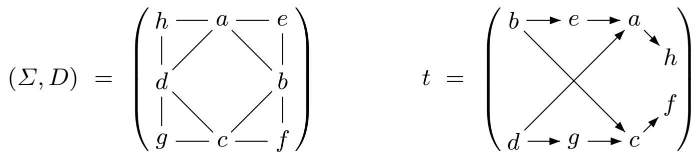


Fig. 1. The dependence alphabet (Σ, D) and the trace $t = bdegachf \in \mathbb{M}(\Sigma, D)$.

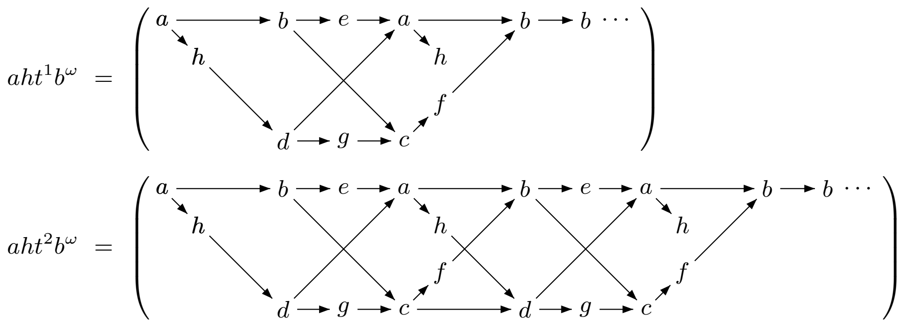


Fig. 2. The traces $aht^i b^\omega \in \mathbb{M}(\Sigma, D)$ for $i = 1, 2$.

path satisfying some first order formula. Let $\varphi(z)$ be a first order formula with z as only free variable and let $C \subseteq \Sigma$. We define the second order formula $\mathrm{Path}_C^\varphi(x, y)$ which mainly claims the existence of a path from x to y satisfying φ by

$$
\begin{aligned}
&x \le y \wedge (\forall z, x \le z \le y \Rightarrow \lambda(z) \in C) \wedge \exists X, \\
&\qquad x \in X \wedge (\forall z \in X, \varphi(z)) \wedge (\forall z \in X, z < y \Rightarrow \exists z' \in X, z \lessdot z' \le y).
\end{aligned}
$$

Lemma 8. *Let (Σ, D) be a dependence alphabet and let $C \subseteq \Sigma$. If (C, D) is a cograph, then there exists a first order formula semantically equivalent to $\mathrm{Path}_C^\varphi(x, y)$ on $\mathbb{R}(\Sigma, D)$.*

Proof. If (C, D) is a cograph, then either C is a singleton, or C is the disjoint union of two nonempty sets $C = A \cup B$ with either $A \times B \subseteq I$ or $A \times B \subseteq D$. We consider these three cases in turn. Note that for all subset $A \subseteq C$, (A, D) is a cograph too.

First, assume that $C \times C \subseteq D$, which is in particular the case when C is a singleton. Then, $\mathrm{Path}_C^\varphi(x, y)$ is semantically equivalent to the first order formula

$$x \le y \wedge (\forall z, x \le z \le y \Rightarrow \lambda(z) \in C) \wedge (\forall z, x \le z \le y \Rightarrow \varphi(z)).$$

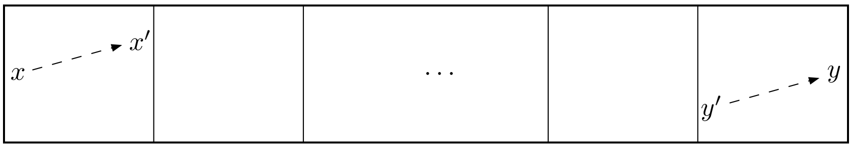
Next, Assume that $C = A \cup B$ with $A \times B \subseteq I$. Then, we have

$$\mathrm{Path}_C^\varphi(x, y) = \mathrm{Path}_A^\varphi(x, y) \vee \mathrm{Path}_B^\varphi(x, y).$$

Finally, we consider the more interesting case $C = A \cup B$ with $A \cap B = \emptyset$ and $A \times B \subseteq D$. Then, we have

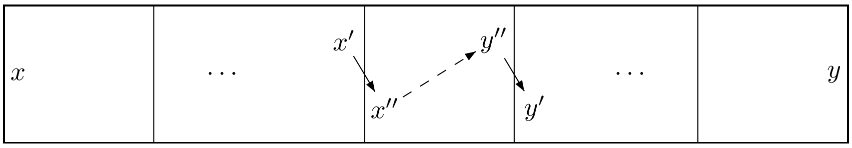
$$\begin{aligned}\mathrm{Path}^{\varphi}_{C}(x,y) = {} & \mathrm{Path}^{\varphi}_{A}(x,y) \vee \mathrm{Path}^{\varphi}_{B}(x,y) \vee \\ & (x \le y \wedge (\forall z, x \le z \le y \Rightarrow \lambda(z) \in C) \wedge \Phi_1(x,y) \wedge \Phi_2(x,y) \wedge \Phi_3(x,y))\end{aligned}$$

where the formulae $\Phi_i(x,y)$ are defined below. We give simultaneously pictures that should help understanding the formulae. The vertical lines in the pictures indicate a separation between factors from $\mathbb{M}^{+}_{A}$ and factors from $\mathbb{M}^{+}_{B}$.



$$\Phi_1(x,y) = \exists x', \begin{pmatrix} x \le x' < y, \\ \wedge\ (\mathrm{Path}^{\varphi}_{A}(x,x') \vee \mathrm{Path}^{\varphi}_{B}(x,x')) \\ \wedge\ \exists u, x' \lessdot u \le y \wedge (\lambda(u) \in A \Leftrightarrow \lambda(x') \notin A) \end{pmatrix}$$

$$\Phi_2(x,y) = \exists y', \begin{pmatrix} x < y' \le y, \\ \wedge\ (\mathrm{Path}^{\varphi}_{A}(y',y) \vee \mathrm{Path}^{\varphi}_{B}(y',y)) \\ \wedge\ \exists u, x \le u \lessdot y' \wedge (\lambda(u) \in A \Leftrightarrow \lambda(y') \notin A) \end{pmatrix}$$



$$\begin{aligned}\Phi_3(x,y) = \forall x'\forall y', & \begin{pmatrix} (x \le x' < y' \le y) \wedge \neg(x' \lessdot y') \\ \wedge\ (\lambda(x') \in A \Leftrightarrow \lambda(y') \in A) \\ \wedge\ (\forall u, x' < u < y' \Rightarrow (\lambda(u) \in A \Leftrightarrow \lambda(x') \notin A)) \end{pmatrix} \\ \Rightarrow & \left(\exists x''\exists y'', x' \lessdot x'' \le y'' \lessdot y' \wedge \left(\mathrm{Path}^{\varphi}_{A}(x'',y'') \vee \mathrm{Path}^{\varphi}_{B}(x'',y'')\right)\right) \quad \square\end{aligned}$$

From this lemma, we immediately deduce the desired result.

Proposition 5. *If the dependence alphabet is a cograph, then the existential modalities* EU *and* ES *can be expressed in* $\mathrm{FO}_{\Sigma}(<)$.

Proof. Follows directly from the Lemma 8 since we have

$$t,x \models \varphi\,\mathsf{EU}\,\psi \quad \text{iff} \quad t,x \models \psi \text{ or } \exists y, \mathrm{Path}^{\varphi}_{\Sigma}(x,y) \text{ and } \exists z,\ y \lessdot z \text{ and } t,z \models \psi$$

$$t,x \models \varphi\,\mathsf{ES}\,\psi \quad \text{iff} \quad t,x \models \psi \text{ or } \exists y, \mathrm{Path}^{\varphi}_{\Sigma}(y,x) \text{ and } \exists z,\ z \lessdot y \text{ and } t,z \models \psi \quad \square$$

The logic TLC introduced in [2] uses EX, EY, the *existential* versions EU and ES of until and since and two additional modalities $\mathsf{Eco}\,\varphi$ claiming that φ holds for some vertex that is concurrent with the current one; and $\mathsf{EG}\,\varphi$ claiming the existence of some maximal path starting from the current vertex such that φ holds everywhere along this path. EX, EY and Eco are clearly first order modalities, while EU, ES and EG are in general only (monadic) second order. But, using the technique of Lemma 8 and Proposition 5 one can show that EG is expressible in first order, if the dependence alphabet is a cograph.

Proposition 6. *Let (Σ, D) be a dependence alphabet which is a cograph. Then,* $\mathrm{TLC}(\Sigma, D) \subseteq \mathrm{FO}_{(\Sigma,D)}(<)$.

6 Complexity

In this section, we show that the satisfiability problem for local temporal logics is PSPACE-complete. The PSPACE-hardness is a consequence of the PSPACE-hardness for words. For TLC the inclusion in PSPACE has been shown in [2]. In order to prove that our problem is still in PSPACE, we have to deal with the universal until–operator which we have introduced here. For this, we associate with each initial formula α an alternating automaton that accepts all linearizations of finite and infinite traces that model α. Here, we describe the construction for the pure future local temporal logic $\mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. Basically we follow the ideas for the usual translation from LTL formulae over words to alternating automata [31].

Let $\alpha \in \mathrm{LocTL}_\Sigma^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$. Without loss of generality, we assume that the negations in α are only over formulae of the form $\mathsf{EM}\,\varphi$ or b or $\mathsf{EX}\,\varphi$ or $\varphi \,\mathsf{U}\, \psi$.

We define $Q = Q_1 \cup Q_2 \cup Q_3 \cup Q_4$ according to Table 1 and we let $\overline{Q} = \{\neg p \mid p \in Q\}$. We construct an alternating automaton $\mathcal{A}_\alpha$ where the state set is $Q' \cup \overline{Q'}$ with $Q' = Q_2 \cup Q_3 \cup Q_4$. However, the larger set Q is used to define the transition relation in a convenient way.

$$
\begin{aligned}
Q_1 &= \{\xi \mid \xi \text{ is a subformula of } \alpha \text{ of the form } \mathsf{EM}\,\varphi \text{ or } b \text{ or } \mathsf{EX}\,\varphi \text{ or } \varphi \,\mathsf{U}\, \psi\}\\
Q_2 &= \{(D(b), \mathsf{EM}\,\varphi) \mid \mathsf{EM}\,\varphi \in Q_1 \text{ and } b \in \Sigma\}\\
Q_3 &= \{(D(b), D(B), \mathsf{EX}\,\varphi) \mid \mathsf{EX}\,\varphi \in Q_1 \text{ and } b \in \Sigma,\ B \subseteq \Sigma,\ \{b\} \cup B \text{ connected}\}\\
Q_4 &= \{(D(A), D(B), \varphi \,\mathsf{U}\, \psi) \mid \varphi \,\mathsf{U}\, \psi \in Q_1 \text{ and } A, B \subseteq \Sigma,\ A \cup B \text{ connected}\}
\end{aligned}
$$

Table 1. States of the alternating automaton $\mathcal{A}_\alpha$.

For the states in Q_2, Q_3, and Q_4 we give a global semantics by defining $t \models q$ for $t = [V, \leq, \lambda] \in \mathbb{R}$ and $q \in Q_2 \cup Q_3 \cup Q_4$ as follows.

$$
\begin{array}{ll}
t \models (D(b), \mathsf{EM}\,\varphi) & \text{if } \exists x \text{ minimal},\ \lambda(x) = a,\ D(a) = D(b),\ t, x \models \varphi\\
t \models (D(b), D(B), \mathsf{EX}\,\varphi) & \text{if } \exists x \text{ minimal},\ \lambda(x) = a,\ a \in D(b) \setminus D(B),\ t, x \models \varphi\\
t \models (D(A), D(B), \varphi \,\mathsf{U}\, \psi) & \text{if } \exists z,\ t, z \models \psi,\ \forall y < z : t, y \models \varphi,\\
& \quad \exists y \leq z : \lambda(y) \in D(A),\ \forall y \leq z : \lambda(y) \notin D(B)
\end{array}
$$

It is clear that some states in Q_3 and Q_4 can never been satisfied. For example, this happens if $D(b) \setminus D(B) = \emptyset$ in Q_3, and similarly, if $D(A) \setminus D(B) = \emptyset$ in Q_4. Thus, all these states can be replaced by the symbol $\perp$. Moreover, some states can also be identified. We have not done it explicitly since this would lead to more case distinctions in the formulae below and hence to a more complicated reading. We shall come to this point in Remark 1.

The set of positive boolean combinations of elements in a set Z is denoted $\mathbb{B}^+(Z)$. Recall that α is a boolean combination of EM-formulae and, by the semantics above, we have $\mathsf{EM}\,\varphi = \bigvee_{b\in\Sigma}(D(b), \mathsf{EM}\,\varphi)$. Hence, α is equivalent to some $\alpha' \in \mathbb{B}^+(Q' \cup \overline{Q'})$, which becomes the initial condition of $\mathcal{A}_\alpha$.

The extended transition function is a mapping $\delta : \mathbb{B}^+(Q \cup \overline{Q}) \times \Sigma \to \mathbb{B}^+(Q \cup \overline{Q})$. The actual transition function of $\mathcal{A}_\alpha$ is the restriction of δ to $(Q' \cup \overline{Q'}) \times \Sigma$. As usual, we define δ inductively. For $f, g \in \mathbb{B}^+(Q \cup \overline{Q})$, $p \in Q$ and $a \in \Sigma$ we define

$$\begin{aligned} \delta(f \vee g, a) &= \delta(f, a) \vee \delta(g, a) \\ \delta(f \wedge g, a) &= \delta(f, a) \wedge \delta(g, a) \\ \delta(\neg p, a) &= \widetilde{\delta(p, a)} \end{aligned}$$

where $\widetilde{f \vee g} = \widetilde{f} \wedge \widetilde{g}$ and $\widetilde{f \wedge g} = \widetilde{f} \vee \widetilde{g}$, for $f, g \in \mathbb{B}^+(Q \cup \overline{Q})$; and $\widetilde{p} = \neg p$ and $\widetilde{\neg p} = p$ for $p \in Q$.

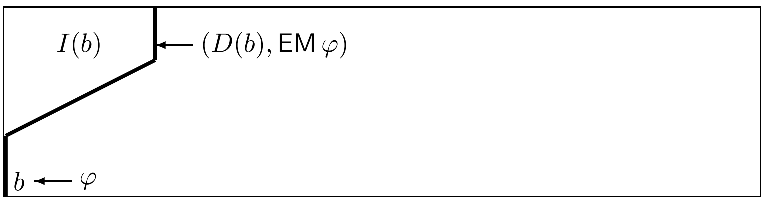
The simplest case for the transition function is when the state is a subformula of the form $b \in \Sigma$ and we read a letter $a \in \Sigma$:

$$\delta(b, a) = \begin{cases} \top & \text{if } a = b, \\ \bot & \text{if } a \neq b. \end{cases}$$

Transitions for $\mathsf{EM}\,\varphi$ and $a \in \Sigma$:

$$\delta(\mathsf{EM}\,\varphi, a) = \textstyle\bigvee_{b\in\Sigma} \delta((D(b), \mathsf{EM}\,\varphi), a)$$

$$\delta((D(b), \mathsf{EM}\,\varphi), a) = \begin{cases} \delta(\varphi, a) & \text{if } D(a) = D(b) \\ (D(b), \mathsf{EM}\,\varphi) & \text{if } a \notin D(b) \\ \bot & \text{otherwise.} \end{cases}$$

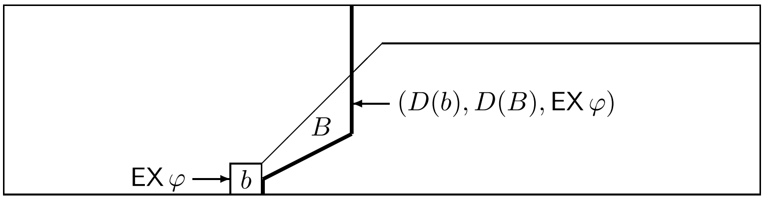


A transition from state $\mathsf{EM}\,\varphi$ only occurs on the first position of the word and looks for a minimal vertex x in the associated trace satisfying φ. Since this minimal vertex needs not be the first letter of the word, we may have to skip some letters independent from x before starting the verification of φ. For this, we guess the set $D(b)$ of letters that depend on x and we skip letters that are not in $D(b)$. When we start the verification of φ, we have to check that our guess was correct ($D(a) = D(b)$).

Transitions for $\mathsf{EX}\,\varphi$ and $a \in \Sigma$:

$$\delta(\mathsf{EX}\,\varphi, a) = (D(a), \emptyset, \mathsf{EX}\,\varphi).$$

$$\delta((D(b), D(B), \mathsf{EX}\,\varphi), a) = \begin{cases} (D(b), D(B), \mathsf{EX}\,\varphi) & \text{if } a \notin D(b) \cup D(B), \\ \delta(\varphi, a) \vee (D(b),\ D(B) \cup D(a), \mathsf{EX}\,\varphi) & \text{if } a \in D(b) \setminus D(B), \\ (D(b),\ D(B) \cup D(a), \mathsf{EX}\,\varphi) & \text{otherwise.} \end{cases}$$

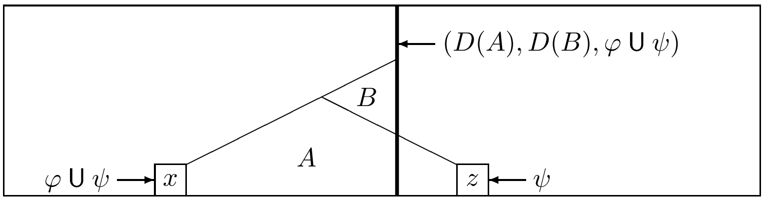


A transition from state $\mathsf{EX}\,\varphi$ looks in the associated trace for an immediate successor of the current vertex satisfying φ. This immediate successor has to be labeled with a letter dependent on the label (say b) of the current vertex. In a state $(D(b), D(B), \mathsf{EX}\,\varphi)$, the alphabetic components allow to check this fact. Since this immediate successor needs not be the next letter in the word that is dependent on b, we may have to skip some letters before starting the verification of φ. Once some letters have been skipped, all dependent letters have to be skipped as well. The second component of $(D(b), D(B), \mathsf{EX}\,\varphi)$ remembers the letters that have to be skipped.

Transitions for $\varphi \,\mathsf{U}\, \psi$ and $a \in \Sigma$:

$$\delta(\varphi \,\mathsf{U}\, \psi, a) = \delta(\psi, a) \vee \delta(\varphi, a) \wedge (D(a), \emptyset, \varphi \,\mathsf{U}\, \psi)$$

$$\delta((D(A), D(B), \varphi \,\mathsf{U}\, \psi), a) = \begin{cases} (D(A), D(B), \varphi \,\mathsf{U}\, \psi) & \text{if } a \notin D(A \cup B), \\ (D(A),\ D(B) \cup D(a), \varphi \,\mathsf{U}\, \psi) & \text{if } a \in D(B), \\ \delta(\psi, a) & \\ \vee\ \delta(\varphi, a) \wedge (D(A) \cup D(a), D(B), \varphi \,\mathsf{U}\, \psi) & \\ \vee\ (D(A),\ D(B) \cup D(a), \varphi \,\mathsf{U}\, \psi) & \text{otherwise.} \end{cases}$$



In order to check that $\varphi \,\mathsf{U}\, \psi$ holds at the current vertex (say x), we have to move forward in the word until we find some vertex (say z) causally in the future of x that satisfies ψ. The first two components of $(D(A), D(B), \varphi \,\mathsf{U}\, \psi)$ gives the letters that must be causally in the future of x. Hence, if we read a letter that is not in $D(A \cup B)$, we just skip it and continue to move forward. Otherwise, we guess whether the letter will be causally in the past of z or not. If yes, then we have to check that φ holds. If no, then we have to remember the letter to make sure that z will not be in its causal future. This is the purpose of the second component in a state $(D(A), D(B), \varphi \mathsf{U} \psi)$. At some point, we have to check that ψ holds when reading a letter in $D(A) \setminus D(B)$ which ensures that z is causally in the future of x and that we have checked φ for all vertices causally between x and z.

Acceptance condition. The automaton that we construct is *very weak* meaning that there is a partial order on states such that the transition function is non-increasing. The order is defined first by the subformulae ordering and second by the reverse containment ordering on the subalphabets. The non increasing property is clear from the definition of the transition function.

Since the automaton is very weak, on each infinite branch of a run, the state is ultimately constant, called its *ultimate* state. For a finite branch its ultimate state is the last state. The branch is accepting if its ultimate state is not in Q' since these states represent obligations that have to be fulfilled eventually. Therefore, the co-Büchi acceptance condition is given by the set Q' and the equivalent Büchi acceptance condition is given by $\overline{Q'}$. Note that $\overline{Q'}$ is also the set of final states used to accept finite traces.

Proposition 7. *The automaton $\mathcal{A}_\alpha$ accepts the word language*

$$\mathcal{L}(\mathcal{A}_\alpha) = \{w \in \Sigma^\infty \mid [w] \models \alpha\},$$

where $[w]$ denotes the trace associated with the word $w \in \Sigma^\infty$.

Remark 1. For the complexity the number of states of $\mathcal{A}_\alpha$ is important. Obviously, it is $|\alpha|$ times a quadratic polynomial in the number of subsets $D(A)$, where $A \subseteq \Sigma$ and $|\alpha|$ denotes the size of the initial formula. To have a better bound let us count the number of states more accurately after some minimization procedure. First we construct reachable states only. Thus we are inside the set $Q' \cup \overline{Q'}$. A state $(D(b), D(B), \mathsf{EX}\,\varphi) \in Q_3$ is never satisfiable if $D(b) \setminus D(B) = \emptyset$. Thus, all these states can be removed and replaced by the symbol $\perp$ when they appear in the right-hand side of a transition. Moreover, $\{b\} \cup B$ is always connected and we may identify $(D(b), D(B), \mathsf{EX}\,\varphi)$ with $(D(b'), D(B), \mathsf{EX}\,\varphi)$ as soon as $D(\{b\} \cup B) = D(\{b'\} \cup B)$. Analogously, a state $(D(A), D(B), \varphi \mathsf{U} \psi) \in Q_4$ is never satisfiable if $D(A) \setminus D(B) = \emptyset$. Again, all these states can be removed and replaced by the symbol $\perp$. Again, $A \cup B$ is connected and we may identify $(D(A), D(B), \varphi \mathsf{U} \psi)$ with $(D(A'), D(B), \varphi \mathsf{U} \psi)$ as soon as $D(A \cup B) = D(A' \cup B)$. Altogether, we can bound the number of states by $2N(\Sigma, D) \cdot |\alpha|$, where

$$N(\Sigma, D) = |\{(D(A \cup B), D(B)) \mid A \cup B \text{ connected and } D(A) \setminus D(B) \neq \emptyset\}|.$$

If the dependence relation is full, that is, when traces are actually words, then $N(\Sigma, D) = 1$ and the size of our automaton does not depend on the size of the alphabet. In this case, we essentially get the usual construction for LTL formulae over words.

Finally, we can decompose (Σ, D) into its connected components such that (Σ, D) is a disjoint union of connected graphs (Σ_i, D_i) for $1 \leq i \leq k$. Then we can construct the automaton for each connected component independently and the number $M(\Sigma, D) = \max_{1 \leq i \leq k} N(\Sigma_i, D_i)$ becomes important rather than $N(\Sigma, D)$. The reason is that $t \models (D(b), \mathsf{EM}\, \varphi)$ if and only $\pi_B(t) \models (D(b), \mathsf{EM}\, \varphi)$, where $\pi_B(t)$ denotes the projection of t over the connected component B of b, that is, $\pi_B(t)$ is the restriction of t to the vertices with label in B.

Theorem 4. *Let $c > 0$ be a constant. The following satisfiability problem is PSPACE-complete:*

Input: *A dependence alphabet (Σ, D) such that $M(\Sigma, D) \leq c$ and a formula $\alpha \in \mathrm{LocTL}_{\Sigma}^{\mathrm{init}}(\mathsf{EX}, \mathsf{U})$.*
Question: *Is there a real trace $t \in \mathbb{R}$ such that $t \models \alpha$?*

Proof. The PSPACE-hardness follows from the word case. The PSPACE algorithm reduces the satisfiability problem in a first phase to a conjunction of satisfiability problems, one for each connected component of (Σ, D). Then for each connected component one after another we can check emptiness for the alternating automata according to the construction in Proposition 7. Checking emptiness for an alternating automaton can be done in PSPACE with respect to the size the automaton. ☐

The construction of an alternating automaton associated with a local temporal logic formula can be carried out for the *existential* until EU and the operator EG from TLC. Also, we can extend the construction for the past modalities EY, S and ES and for the operator Eco from TLC is we use two-way alternating automata. Since the emptiness problem for two-way alternating automata is also PSPACE-complete [17], we get a similar result for the local temporal logic using all operators. This extends the result in [2] concerning TLC.

7 Conclusion

We have defined a basic and natural local logic for Mazurkiewicz traces which is expressively complete with respect to first order if and only if the dependence alphabet is a cograph, i.e., all traces are series parallel posets. The main open problem remains to define a (natural) local logic which yields expressive completeness for more general (best for all) dependence alphabets, and such that the satisfiability problem is in PSPACE or at least elementary.

There were two recent proposals to solve this problem. The first one [14] introduce a local temporal logic which is in the same spirit that the one used in the present paper but uses filtered until and past tense modalities (EY and S). The second one [1] uses a past oriented logic with a rather special form of previous modality with two arguments and a non-standard semantics (the evaluation of one argument is restricted to a factor of the original trace). In both cases, the logic is proved to be expressively complete for arbitrary dependence alphabets and decidable in PSPACE. Still, the problem of finding more natural logics that are expressively complete and decidable in PSPACE remains open.

There is also a proposal by Walukiewicz [33] for a local logic for traces, but his focus is on monadic second order logic and based on a μ–calculus, so it is of quite different spirit.

References

1. B. Adsul and M. Sohoni. Complete and tractable local linear time temporal logics over traces. In *Proc. of ICALP'02*, number 2380 in LNCS, pages 926–937. Springer Verlag, 2002.
2. R. Alur, D. Peled, and W. Penczek. Model-checking of causality properties. In *Proc. of LICS'95*, pages 90–100, 1995.
3. A. Brandstädt, Van Bang Le, and J.P. Spinrad. *Graph Classes: A Survey.* SIAM Monographs on Discrete Mathematics and Applications. 1999.
4. Ch. Choffrut. Combinatorics in trace monoids I. In V. Diekert and G. Rozenberg, editors, *The Book of Traces*, chapter 3, pages 71–82. World Scientific, Singapore, 1995.
5. V. Diekert. A pure future local temporal logic beyond cograph-monoids. In M. Ito, editor, *Proc. of the RIMS Symposium on Algebraic Systems, Formal Languages and Conventional and Unconventional Computation Theory, Kyoto, Japan*, 2002.
6. V. Diekert and P. Gastin. LTL is expressively complete for Mazurkiewicz traces. In *Proc. of ICALP'00*, number 1853 in LNCS, pages 211–222. Springer Verlag, 2000.
7. V. Diekert and P. Gastin. Local temporal logic is expressively complete for cograph dependence alphabets. In *Proc. of LPAR'01*, number 2250 in LNAI, pages 55–69. Springer Verlag, 2001.
8. V. Diekert and P. Gastin. LTL is expressively complete for Mazurkiewicz traces. *Journal of Computer and System Sciences*, 64:396–418, 2002.
9. V. Diekert and P. Gastin. Pure future local temporal logics are expressively complete for Mazurkiewicz traces. In M. Farach-Colton, editor, *Proc. of the 6th Latin American theoretical INformatics Symposium (LATIN'04)*, number 2976 in LNCS, pages 232–241. Springer Verlag, 2004.
10. V. Diekert and G. Rozenberg, editors. *The Book of Traces.* World Scientific, Singapore, 1995.
11. W. Ebinger. *Charakterisierung von Sprachklassen unendlicher Spuren durch Logiken.* Dissertation, Institut für Informatik, Universität Stuttgart, 1994.
12. W. Ebinger and A. Muscholl. Logical definability on infinite traces. *Theoretical Computer Science*, 154:67–84, 1996.
13. D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Proc. of PoPL'80*, pages 163–173, Las Vegas, Nev., 1980.
14. P. Gastin and M. Mukund. An elementary expressively complete temporal logic for Mazurkiewicz traces. In *Proc. of ICALP'02*, number 2380 in LNCS, pages 938–949. Springer Verlag, 2002.
15. J. G. Henriksen. An expressive extension of TLC. In *Proc. of ASIAN'99*, number 1742 in LNCS, pages 126–138. Springer Verlag, 1999.
16. J. G. Henriksen. *Logic and Automata for Verification: Expressiveness and Decidability Issues.* PhD thesis, Dept. of Comp. Sci., University of Aarhus, 2000.

17. T. Jiang and B. Ravikumar. A note on the space complexity of some decision problems for finite automata. *Information Processing Letters*, 40:25–31, 1991.
18. J.A.W. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, California, 1968.
19. R.M. Keller. Parallel program schemata and maximal parallelism I. Fundamental results. *Journal of the Association of Computing Machinery*, 20(3):514–537, 1973.
20. M. Lothaire. *Combinatorics on Words*, volume 17 of *Encyclopedia of Mathematics and its Applications*. Addison-Wesley, Reading, MA, 1983. Reprinted by *Cambridge University Press*, 1997.
21. A. Mazurkiewicz. Concurrent program schemes and their interpretations. DAIMI Rep. PB 78, Aarhus University, Aarhus, 1977.
22. R. McNaughton and S. Papert. *Counter-Free Automata*. The MIT Press, Cambridge, Mass., 1971.
23. M. Mukund and P.S. Thiagarajan. Linear time temporal logics over Mazurkiewicz traces. In *Proc. of MFCS'96*, number 1113 in LNCS, pages 62–92. Springer Verlag, 1996.
24. P. Niebert. A ν-calculus with local views for sequential agents. In *Proc. of MFCS'95*, number 969 in LNCS, pages 563–573. Springer Verlag, 1995.
25. W. Penczek. Temporal logics for trace systems: On automated verification. *International Journal of Foundations of Computer Science*, 4:31–67, 1993.
26. D. Perrin and J.-E. Pin. *Infinite words*, volume 141 of *Pure and Applied Mathematics*. Elsevier, 2004.
27. R. Ramanujam. Locally linear time temporal logic. In *Proc. of LICS'96*, pages 118–128, 1996.
28. M.-P. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8:190–194, 1965.
29. P.S. Thiagarajan. A trace based extension of linear time temporal logic. In *Proc. of LICS'94*, pages 438–447. IEEE Computer Society Press, 1994.
30. P.S. Thiagarajan and I. Walukiewicz. An expressively complete linear time temporal logic for Mazurkiewicz traces. In *Proc. of LICS'97*, pages 183–194, 1997.
31. M.Y. Vardi. An automata-theoretic approach to linear temporal logic. In *Logics for Concurrency: Structure versus Automata*, number 1043 in LNCS, pages 238–266. Springer Verlag, 1996.
32. I. Walukiewicz. Difficult configurations – on the complexity of LTrL. In *Proc. of ICALP'98*, number 1443 in LNCS, pages 140–151. Springer Verlag, 1998.
33. I. Walukiewicz. Local logics for traces. *Journal of Automata, Languages and Combinatorics*, 7(2):259–290, 2002.
34. Th. Wilke. Classifying discrete temporal properties. In *Proc. of STACS'99*, number 1563 in LNCS, pages 32–46. Springer Verlag, 1999.