LTL is expressively complete for Mazurkiewicz traces

Volker Diekert[1] and Paul Gastin[2]

[1] Inst. für Informatik, Universität Stuttgart, Breitwiesenstr. 20-22, D-70565 Stuttgart
diekert@informatik.uni-stuttgart.de
[2] LIAFA, Université Paris 7, 2, place Jussieu, F-75251 Paris Cedex 05
Paul.Gastin@liafa.jussieu.fr

Abstract. A long standing open problem in the theory of (Mazurkiewicz) traces has been the question whether LTL (Linear Time Logic) is expressively complete with respect to the first order theory. We solve this problem positively for finite and infinite traces and for the simplest temporal logic, which is based only on next and until modalities. Similar results were established previously, but they were all weaker, since they used additional past or future modalities. Another feature of our work is that our proof is direct and does not use any reduction to the word case.
Keywords Temporal logics, Mazurkiewicz traces, concurrency

1 Introduction

Nowadays, it is widely accepted that we need to develop methods to verify critical systems. For this, we need formal specifications for the expected behaviors of systems. Most often, these formal specifications are given by temporal logics formulae. The task is even harder when considering concurrent systems. Indeed, the usual approach is to reduce concurrent systems to sequential ones by considering linearizations of concurrent executions. This allows to use techniques and tools developed for sequential systems but introduces a combinatorial explosion which has then to be fought by all means. Instead, one could try to work directly on concurrent systems and this explains why a lot of research has been devoted recently to the study of temporal logics for concurrency. A major aim is to find a temporal logic which is expressive enough to ensure that all desired specification can be formalized.

For sequential systems, Kamp's Theorem [8] states that the linear time logic has the same expressive power as the first order theory of words. Originally Kamp used future and past modalities, but it was established later by Gabbay's separation theorem [6] that past modalities can be avoided.

Trace theory, initiated by Mazurkiewicz is one of the most popular settings to study concurrency. See [3] for the general background of trace theory, and in particular [13] for traces and logic and [7] for infinite traces. It is no surprise that various temporal logics for traces have been extensively studied [1, 10–12, 14–16]. A long standing open problem [4, 17] was to know whether LTL (the basic linear temporal logic) is expressively complete with respect to the first order theory of finite and infinite traces.

The first completeness result for traces is from Ebinger [4]. He proved that a linear temporal logic with both past and future modalities is expressively complete for finite traces but his approach did not cope with infinite traces. Then, Thiagarajan and Walukiewicz [17] proved the completeness both for finite and infinite traces of LTrL, a linear temporal logic with future modalities and past modalities in the restricted form of past constants. These two results were obtained using a reduction to the word case. In [9] it was claimed that LTL is expressively complete for finite traces, but the proof contained a serious mistake. The expressive completeness for a pure future temporal logic (LTL_f) was first established in [2]. The result holds for finite and infinite traces, but LTL_f contains new filter modalities in addition to the usual next and until modalities. Thus, the problem of the expressive completeness of the basic linear temporal logic was still open.

In this paper, we solve this problem positively. Previous expressive completeness results for words and for traces are now formal corollaries of our main theorem. It should be noted that, contrary to most previous works, our proof does not use any reduction to the word case. Instead, we extend the new proof introduced by Wilke for finite words [18] which is based on the well-known fact that first order languages are aperiodic. We follow here the same approach as in [2]. In the former paper filter modalities were used to express some special products of trace languages. Our new proof is a substantial revision of the previous one. We are now able to express the products mentioned above directly with basic formulae without using the filter modalities.

2 Preliminaries

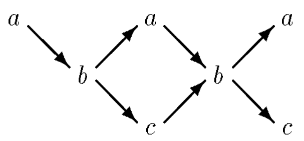
By (Σ, I) we mean a finite *independence alphabet* where Σ denotes a finite alphabet and $I \subseteq \Sigma \times \Sigma$ is an irreflexive and symmetric relation called the *independence relation*. The complementary relation $D = (\Sigma \times \Sigma) \setminus I$ is called the *dependence relation*. The monoid of *finite traces* $\mathbb{M}(\Sigma, I)$ is defined as a quotient monoid with respect to the congruence relation induced by I, i.e., $\mathbb{M}(\Sigma, I) = \Sigma^*/\{\, ab = ba \mid (a, b) \in I\,\}$. We also write $\mathbb{M}$ instead of $\mathbb{M}(\Sigma, I)$. For $A \subseteq \Sigma$ we denote by $\mathbb{M}_A$ the submonoid of $\mathbb{M}(\Sigma, I)$ generated by A:

$$\mathbb{M}_A = \mathbb{M}(A, I \cap A \times A) = \{x \in \mathbb{M}(\Sigma, I) \mid \mathrm{alph}(x) \subseteq A\}.$$

A trace $x \in \mathbb{M}$ is given by a congruence class of a word $a_1 \cdots a_n \in \Sigma^*$ where $a_i \in \Sigma$, $1 \leq n$. By abuse of language, but for simplicity we denote a trace x by one of its representing words $a_1 \cdots a_n$. The number n is called the length of x, it is denoted by $|x|$. For $n = 0$ we obtain the *empty* trace, it is denoted by 1. The *alphabet* $\mathrm{alph}(x)$ of a trace x is the set of letters occurring in x. A *trace language* is a subset $L \subseteq \mathbb{M}$. The concatenation is defined as usual:

$$KL = \{xy \in \mathbb{M} \mid x \in K, y \in L\}.$$

Every trace $a_1 \cdots a_n \in \mathbb{M}$ can be identified with its dependence graph. This is (an isomorphism class of) a node–labeled, acyclic, directed graph $[V, E, \lambda]$, where $V = \{1, \ldots, n\}$ is a set of vertices, each $i \in V$ is labeled by $\lambda(i) = a_i$, and there is an edge $(i, j) \in E$ if and only if both $i < j$ and $(\lambda(i), \lambda(j)) \in D$. In pictures it is common to draw the Hasse diagram only. Thus, all redundant edges are omitted. For instance, let $(\Sigma, D) = a — b — c$, i.e., $I = \{(a, c), (c, a)\}$. Then the trace $x = abcabca$ is given by



An *infinite trace* is an infinite dependence graph $[V, E, \lambda]$ such that for all $j \in V$ the set $\downarrow j = \{i \in V \mid i \leq j\}$ is finite. A *real trace* is a finite or infinite trace. The set of real traces is denoted by $\mathbb{R}(\Sigma, I)$ or simply by $\mathbb{R}$. For a real trace $x = [V, E, \lambda]$ the *alphabet* is $\mathrm{alph}(x) = \lambda^{-1}(V)$ and the *alphabet at infinity* is defined by the set of letters occurring infinitely many times in x, i.e., $\mathrm{alphinf}(x) = \{a \in \Sigma \mid |\lambda^{-1}(a)| = \infty\}$. See [7] for details about infinite traces. Usually we consider real trace languages. If we speak about *finitary* languages, then we refer to subsets of $\mathbb{M}$. As for finite traces, if $A \subseteq \Sigma$ is a subalphabet then we let $\mathbb{R}_A = \{x \in \mathbb{R} \mid \mathrm{alph}(x) \subseteq A\}$.

By $\min(x)$ and $\max(x)$ we refer to the *minimal* and *maximal letters* in the dependence graph. (We shall use the notation $\max(x)$ only for finite traces $x \in \mathbb{M}$.) In the example above $\min(x) = \{a\}$ and $\max(x) = \{a, c\}$. Formally:

$$\begin{aligned} \min(x) &= \{a \in \Sigma \mid x \in a\mathbb{R}\}, \\ \max(x) &= \{a \in \Sigma \mid x \in \mathbb{M}a\}. \end{aligned}$$

For $B \subseteq \Sigma$ and $\# \in \{\subseteq, =, \supseteq, \neq\}$ we define:

$$\begin{aligned} I(B) &= \{a \in \Sigma \mid \forall b \in B : (a, b) \in I\}, \\ D(B) &= \{a \in \Sigma \mid \exists b \in B : (a, b) \in D\}, \\ (\mathrm{Min} \ \# \ B) &= \{x \in \mathbb{R} \mid \min(x) \ \# \ B\}, \\ (\mathrm{Max} \ \# \ B) &= \{x \in \mathbb{M} \mid \max(x) \ \# \ B\}. \end{aligned}$$

Note that $(\mathrm{Max} \ \# \ B)$ denotes a finitary language, whereas $(\mathrm{Min} \ \# \ B)$ is a real trace language. If B happens to be a singleton, then we usually omit braces, e.g. we write $I(b)$ or $(\mathrm{Max} = b)$.

3 Temporal logic for traces

The syntax of the temporal logic $\mathrm{LTL}(\Sigma)$ is defined as follows. There are a constant symbol $\perp$ representing *false*, the logical connectives $\neg$ (not) and $\vee$ (or), for each $a \in \Sigma$ a unary operator $\langle a \rangle$, called *next-a*, and a binary operator $\mathbf{U}$, called *until*. Formally, the syntax is given by:

$$\varphi ::= \perp \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle a \rangle \varphi \mid \varphi \, \mathbf{U} \, \varphi,$$

where $a \in \Sigma$.

The semantics is usually defined by saying when some formula φ is satisfied by some real trace z at some configuration (i.e., finite prefix) x; hence by defining $(z, x) \models \varphi$. Since our temporal logic uses future modalities only, we have $(z, x) \models \varphi$ if and only if $(y, 1) \models \varphi$, where y is the unique trace satisfying $z = xy$. Therefore, we do not need to deal with configurations and it is enough to say when a trace satisfies a formula at the empty configuration, denoted simply by $z \models \varphi$. This is done inductively on the formula as follows:

$$
\begin{array}{ll}
z \not\models \bot, & \\
z \models \neg\varphi & \text{if } z \not\models \varphi, \\
z \models \varphi \vee \psi & \text{if } z \models \varphi \text{ or } z \models \psi, \\
z \models \langle a \rangle \varphi & \text{if } z = ay \text{ and } y \models \varphi, \\
z \models \varphi \,\mathbf{U}\, \psi & \text{if } z = xy,\ x \in \mathbb{M},\ y \models \psi, \text{ and } x = x'x'',\ x'' \neq 1 \text{ implies } x''y \models \varphi.
\end{array}
$$

As usual, we define $\mathrm{L}_{\mathbb{R}}(\varphi) = \{x \in \mathbb{R} \mid x \models \varphi\}$. We say that a trace language $L \subseteq \mathbb{R}$ is *expressible in* $\mathrm{LTL}(\Sigma)$, if there exists a formula $\varphi \in \mathrm{LTL}(\Sigma)$ such that $L = \mathrm{L}_{\mathbb{R}}(\varphi)$.

Equivalently, we can define inductively the language $\mathrm{L}_{\mathbb{R}}(\varphi)$ as follows:

$$
\begin{aligned}
\mathrm{L}_{\mathbb{R}}(\bot) &= \emptyset \\
\mathrm{L}_{\mathbb{R}}(\neg\varphi) &= \mathbb{R} \setminus \mathrm{L}_{\mathbb{R}}(\varphi) \\
\mathrm{L}_{\mathbb{R}}(\varphi \vee \psi) &= \mathrm{L}_{\mathbb{R}}(\varphi) \cup \mathrm{L}_{\mathbb{R}}(\psi) \\
\mathrm{L}_{\mathbb{R}}(\langle a \rangle \varphi) &= a\mathrm{L}_{\mathbb{R}}(\varphi) \\
\mathrm{L}_{\mathbb{R}}(\varphi \,\mathbf{U}\, \psi) &= \mathrm{L}_{\mathbb{R}}(\varphi) \,\mathbf{U}\, \mathrm{L}_{\mathbb{R}}(\psi)
\end{aligned}
$$

where the *until* operator $\mathbf{U}$ is defined on real trace languages by

$$L \,\mathbf{U}\, K = \{xy \mid x \in \mathbb{M},\ y \in K, \text{ and } x = x'x'',\ x'' \neq 1 \text{ implies } x''y \in L\}.$$

Remark 1. For comparison let us mention that the syntax and semantics of the logics LTrL defined in [17] and LTL_f defined in [2] are very similar. For LTrL, the difference is only that there is in addition for each letter $a \in \Sigma$ a constant $\langle a^{-1} \rangle \top$. Since the constant $\langle a^{-1} \rangle \top$ refers to the past, we need to use configurations to define its semantics: It holds $(z, x) \models \langle a^{-1} \rangle \top$ if and only if $a \in \max(x)$. For LTL_f, the difference is that for each subalphabet $B \subseteq \Sigma$ there is in addition a modality $\langle B^* \rangle \varphi$ whose semantics is given by $z \models \langle B^* \rangle \varphi$ if $z = xy$, $x \in \mathbb{M}_B$, and $y \models \varphi$. It is not clear whether there is a direct translation of LTrL or LTL_f to LTL; or between LTrL and LTL_f.

The following operators are standard abbreviations. They serve as macros.

$$
\begin{array}{rll}
\top & := \neg\bot & \textit{true}, \\
a & := \langle a \rangle \top & \text{for } a \in \Sigma, \\
A & := \bigvee_{a \in A} \langle a \rangle \top & \text{for } A \subseteq \Sigma, \\
\mathbf{X}\,\varphi & := \bigvee_{a \in \Sigma} \langle a \rangle \varphi & \textit{neXt } \varphi, \\
\mathrm{stop} & := \neg\,\mathbf{X}\,\top & \textit{termination}, \\
\mathbf{F}\,\varphi & := \top\,\mathbf{U}\,\varphi & \textit{future} \text{ or } \textit{eventually}\varphi, \\
\mathbf{G}\,\varphi & := \neg\,\mathbf{F}\,\neg\varphi & \textit{globally} \text{ or } \textit{always}\varphi.
\end{array}
$$

Example 1. $\mathrm{L}_{\mathbb{R}}(\mathrm{stop}) = \{1\}$, $\mathrm{L}_{\mathbb{R}}(\mathbf{F}\,\mathrm{stop}) = \mathbb{M}$ and for $A \subseteq \Sigma$ we have:

$$\begin{aligned}
\mathbb{R}_A &= \mathrm{L}_{\mathbb{R}}(\neg\,\mathbf{F}(\Sigma \setminus A)),\\
\mathbb{M}_A &= \mathrm{L}_{\mathbb{R}}(\mathbf{F}\,\mathrm{stop} \wedge \neg\,\mathbf{F}(\Sigma \setminus A)),\\
(\mathrm{Min} = A) &= \mathrm{L}_{\mathbb{R}}(\neg(\Sigma \setminus A) \wedge \textstyle\bigwedge_{a \in A} a),\\
(\mathrm{Max} \supseteq A) &= \mathrm{L}_{\mathbb{R}}(\textstyle\bigwedge_{a \in A} \mathbf{F}\langle a\rangle \mathrm{stop}),\\
(\mathrm{alphinf} = A) &= \mathrm{L}_{\mathbb{R}}(\mathbf{F}\,\mathbf{G}\,\neg(\Sigma \setminus A) \wedge \textstyle\bigwedge_{a \in A} \mathbf{G}\,\mathbf{F}\,a).
\end{aligned}$$

Remark 2. Later we shall perform an induction on the size of Σ leading to formulae $\varphi \in \mathrm{LTL}(A)$ for $A \subseteq \Sigma$. Such a formula may be interpreted over $\mathbb{R}_A$ or over $\mathbb{R}$ and we have $\mathrm{L}_{\mathbb{R}_A}(\varphi) = \mathrm{L}_{\mathbb{R}}(\varphi) \cap \mathbb{R}_A$. Hence $\mathrm{L}_{\mathbb{R}_A}(\varphi)$ is expressible in $\mathrm{LTL}(\Sigma)$. Another trivial observation is that if $\psi \in \mathrm{LTL}(\Sigma)$, then we can construct a formula $\psi_A \in \mathrm{LTL}(A)$ such that $\mathrm{L}_{\mathbb{R}}(\psi) \cap \mathbb{R}_A = \mathrm{L}_{\mathbb{R}_A}(\psi_A)$.

We will also use an induction on $|\Sigma|$ when the graph (Σ, D) is not connected. Then we find a partition of the alphabet $\Sigma = \Sigma_1 \cup \Sigma_2$ with $\Sigma_1 \times \Sigma_2 \subseteq I$ and each trace $x \in \mathbb{R}$ can be split into two independent traces x_1 and x_2 over Σ_1 and Σ_2 with $x = x_1 x_2 = x_2 x_1$. The following lemma can be shown by induction.

Lemma 1. *Assume that $\Sigma = \Sigma_1 \cup \Sigma_2$ with $\Sigma_1 \times \Sigma_2 \subseteq I$. Let $\mathbb{R}_i = \mathbb{R}_{\Sigma_i}$ and $\varphi_i \in \mathrm{LTL}(\Sigma_i)$ for $i = 1, 2$. Then $\mathrm{L}_{\mathbb{R}}(\varphi_1 \wedge \varphi_2) = \mathrm{L}_{\mathbb{R}_1}(\varphi_1) \cdot \mathrm{L}_{\mathbb{R}_2}(\varphi_2)$.*

4 First-order logic

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 $x, y \in \mathrm{Var}$ are first order variables. Given a trace $t = [V, E, \lambda]$ and a valuation of the free variables into the vertices $\sigma : \mathrm{Var} \to V$, the semantics is obtained by interpreting the relation $<$ as the transitive closure of E and the predicate $P_a(x)$ by $\lambda(\sigma(x)) = a$. Then we can say when $(t, \sigma) \models \varphi$. If φ is a closed formula (a sentence), then the valuation σ has an empty domain and we define the language $\mathrm{L}_{\mathbb{R}}(\varphi) = \{t \in \mathbb{R} \mid t \models \varphi\}$. 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 $L = \mathrm{L}_{\mathbb{R}}(\varphi)$.

Passing from a temporal logic formula to a first order one is not very difficult. It is well-known or belongs to folklore. The transformation relies on the fact that a prefix (configuration) p of a trace t can be defined by its maximal vertices. Such a set of maximal vertices is bounded by the maximal number of pairwise independent letters in Σ. Therefore, a prefix inside a trace can be defined using a bounded number of first order variables.

Proposition 1. *If a trace language is expressible in $\mathrm{LTL}(\Sigma)$, then it is expressible in $\mathrm{FO}(\Sigma, <)$.*

5 Aperiodic languages

Recall that 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 finitary trace language $L \subseteq \mathbb{M}$ is *aperiodic*, if there exists a morphism to some finite aperiodic monoid $h : \mathbb{M} \to S$ such that $L = h^{-1}(h(L))$. Since our considerations include infinite traces, we have to extend the notion of an aperiodic language to real trace languages such that it becomes equivalent for finitary languages.

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 non-empty finite traces $x = x_1 x_2 \cdots$, $y = y_1 y_2 \cdots$ with $x_i, y_i \in \mathbb{M} \setminus \{1\}$ and $h(x_i) = h(y_i)$ for all i. According to the definition of h-similarity, we never have $x \sim_h y$ when x is finite and y is infinite. We denote by $\approx_h$ the transitive closure of $\sim_h$ which is therefore an equivalence relation. An equivalence class is denoted by $[x]_{\approx_h} = \{y \in \mathbb{R} \mid y \approx_h x\}$. For a finite trace $x \in \mathbb{M}$ we have $[x]_{\approx_h} = h^{-1}(h(x))$ and the monoid $\mathbb{M}$ is covered by at most $|S|$ classes. Using a Ramsey-type argument we can show that $\mathbb{R} \setminus \mathbb{M}$ is covered by at most $|S|^2$ classes. Therefore, $\{[x]_{\approx_h} \mid x \in \mathbb{R}\}$ defines a finite partition of $\mathbb{R}$ of cardinality at most $|S|^2 + |S|$. 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 $x \in \mathbb{R}$. A real trace language $L \subseteq \mathbb{R}$ is *aperiodic* if it is recognized by some morphism to some finite and aperiodic monoid.

In order to prove the main theorem we shall use the equivalence between $\mathrm{FO}(\Sigma, <)$–definability and aperiodic languages.

Theorem 1 ([4, 5]). *A finitary trace language (real trace language resp.) over an independence alphabet (Σ, I) is expressible in $\mathrm{FO}(\Sigma, <)$ if and only if it is aperiodic.*

We conclude this section by some easy results which will be useful later. The first proposition will allow us to use an induction on the size of the alphabet Σ.

Proposition 2. *Let $L \subseteq \mathbb{R}$ be a language recognized by the morphism $h : \mathbb{M} \to S$ into a finite monoid S and let $A \subseteq \Sigma$. Then, $L \cap \mathbb{R}_A$ and $L \cap \mathbb{M}_A$ are recognized by the restriction $h{\restriction}_{\mathbb{M}_A}$ of h to $\mathbb{M}_A$.*

Then, we consider the case where the dependence alphabet (Σ, D) is non-connected. A kind of Mezei's Theorem holds for real trace languages too:

Proposition 3. *Let $L \subseteq \mathbb{R}$ be a language recognized by the morphism $h : \mathbb{M} \to S$ into a finite monoid S. Assume that $\Sigma = \Sigma_1 \cup \Sigma_2$ with $\Sigma_1 \times \Sigma_2 \subseteq I$ and let $\mathbb{M}_i = \mathbb{M}_{\Sigma_i}$, $\mathbb{R}_i = \mathbb{R}_{\Sigma_i}$ and $h_i = h{\restriction}_{M_i}$ for $i = 1, 2$. Then, L is a finite union of products $L_1 \cdot L_2$ where $L_i \subseteq \mathbb{R}_i$ is recognized by h_i for $i = 1, 2$.*

Let $h : \mathbb{M} \to S$ be a morphism to some finite monoid S and let $L \subseteq \mathbb{R}$. For $s \in S$ we define $L(s) = \{x \in \mathbb{R} \mid h^{-1}(s)x \cap L \neq \emptyset\}$.

Proposition 4. *If L is recognized by h then $L = \bigcup_{s \in S} h^{-1}(s) \cdot L(s)$ and for each $s \in S$, the language $L(s)$ is recognized by h.*

6 Composition of LTL languages

In the next section we will prove that each aperiodic trace language is expressible in $\mathrm{LTL}(\Sigma)$ using some induction. For this, we will write an aperiodic language as a finite union of products of simpler languages. By induction, the simpler languages will be shown to be expressible in $\mathrm{LTL}(\Sigma)$. Then we have to prove that their products and unions are still expressible in $\mathrm{LTL}(\Sigma)$. Since union corresponds with disjunction, the only problem is with concatenation. Though it is true in general that the product of languages expressible in $\mathrm{LTL}(\Sigma)$ is still expressible in $\mathrm{LTL}(\Sigma)$, we do not have a direct proof for this general result. Actually, this becomes a consequence of our main theorem since aperiodic trace languages are closed under product. In this section we show that some special products of expressible languages are still expressible in $\mathrm{LTL}(\Sigma)$. We state now two crucial composition lemmas. The proof techniques are similar, for lack of space we show the proof of Lemma 3 only.

Lemma 2. *Let $b \in \Sigma$, $B = \Sigma \setminus \{b\}$ and $\Pi = \mathbb{M}_B \cap (\mathrm{Max} \subseteq D(b))$. Let $L_1, L_2 \subseteq \mathbb{R}$ and $L_3 \subseteq (\mathrm{Min} = b)$ be trace languages expressible in* $\mathrm{LTL}(\Sigma)$*. Then the language $(L_1 \cap \Pi)(L_2 \cap \mathbb{M}_{I(b)})L_3$ is expressible in* $\mathrm{LTL}(\Sigma)$.

Lemma 3. *Let $b \in \Sigma$ and $B = \Sigma \setminus \{b\}$. Let $L_1 \subseteq \mathbb{R}$ and $L_2 \subseteq \mathbb{R}_B$ be expressible in* $\mathrm{LTL}(\Sigma)$*. Then the language $(L_1 \cap (\mathrm{Max} \subseteq \{b\}))L_2$ is expressible in* $\mathrm{LTL}(\Sigma)$.

Proof. The product $(\mathrm{Max} \subseteq \{b\})\mathbb{R}_B$ is unambiguous and we have

$$(L_1 \cap (\mathrm{Max} \subseteq \{b\}))L_2 = (L_1 \cap (\mathrm{Max} \subseteq \{b\}))\mathbb{R}_B \cap (\mathrm{Max} \subseteq \{b\})L_2.$$

1) We first show that $(L_1 \cap (\mathrm{Max} \subseteq \{b\}))\mathbb{R}_B$ is expressible in $\mathrm{LTL}(\Sigma)$. This is done by induction on the formula which defines L_1. The cases $\perp$ and $\varphi \vee \psi$ are trivial.

• $\neg\varphi$: Since the product $(\mathrm{Max} \subseteq \{b\})\mathbb{R}_B$ is unambiguous, we have

$$(\mathrm{L}_{\mathbb{R}}(\neg\varphi) \cap (\mathrm{Max} \subseteq \{b\}))\mathbb{R}_B = (\mathrm{Max} \subseteq \{b\})\mathbb{R}_B \setminus (\mathrm{L}_{\mathbb{R}}(\varphi) \cap (\mathrm{Max} \subseteq \{b\}))\mathbb{R}_B.$$

Moreover $(\mathrm{Max} \subseteq \{b\})\mathbb{R}_B$ is the set $(\mathrm{Alphinf} \subseteq B)$, which is expressible in $\mathrm{LTL}(\Sigma)$ by the formula $\mathbf{F}\,\mathbf{G}\,\neg b$.

• $\langle a \rangle\varphi$: The language $(\mathrm{L}_{\mathbb{R}}(\langle a \rangle\varphi) \cap (\mathrm{Max} \subseteq \{b\}))\mathbb{R}_B$ is equal to the language $a\Big((\mathrm{L}_{\mathbb{R}}(\varphi) \cap (\mathrm{Max} \subseteq \{b\}))\mathbb{R}_B \cap (a^{-1}(\mathrm{Max} = b))\mathbb{R}_B\Big)$. We conclude by induction since we have $(a^{-1}(\mathrm{Max} = b))\mathbb{R}_B = \mathrm{L}_{\mathbb{R}}(\psi)$ with

$$\psi = \bigvee_{a = b_0 - b_1 - \cdots - b_k = b} \mathbf{F}\langle b_1 \rangle \cdots \mathbf{F}\langle b_k \rangle(\neg\,\mathbf{F}\,b)$$

where the disjunction ranges over all simple paths from a to b in the graph of the dependence alphabet (Σ, D).

• $\varphi\,\mathbf{U}\,\psi$: This case follows by induction from the formula

$$\begin{aligned}(\mathrm{L}_{\mathbb{R}}(\varphi\,\mathbf{U}\,\psi) \cap (\mathrm{Max} \subseteq \{b\}))\mathbb{R}_B = \\ \big(\mathrm{L}_{\mathbb{R}}(\varphi) \cap (\mathrm{Max} \subseteq \{b\})\big)\mathbb{R}_B\,\mathbf{U}\,\big(\mathrm{L}_{\mathbb{R}}(\psi) \cap (\mathrm{Max} \subseteq \{b\})\big)\mathbb{R}_B.\end{aligned}$$

2) Next, if $\varphi \in \mathrm{LTL}(\Sigma)$ is a formula such that $\mathrm{L}_{\mathbb{R}}(\varphi) \subseteq \mathbb{R}_B$ then we have $(\mathrm{Max} \subseteq \{b\}) \cdot \mathrm{L}_{\mathbb{R}}(\varphi) = \mathrm{L}_{\mathbb{R}}((\mathbf{F}\,b)\,\mathbf{U}\,\varphi)$.

7 Kamp's theorem for real traces

Theorem 2. *A finitary trace language (real trace language resp.) is expressible in* $\mathrm{FO}(\Sigma, <)$ *if and only if it is expressible in* $\mathrm{LTL}(\Sigma)$*.*

By Proposition 1 and Theorem 1 it is enough to show that all aperiodic languages in $\mathbb{R}(\Sigma, I)$ are expressible in $\mathrm{LTL}(\Sigma)$. The overall strategy is as in [2], but we must not use any filter operator. The filter operators allowed us to express directly a language of type $\mathbb{M}_A\, L(\varphi)$ for $A \subseteq \Sigma$ and $\varphi \in \mathrm{LTL}$. This simplified the proof considerably, but cannot be used here. We can circumvent these difficulties thanks to the results of the previous section.

Let Q be a finite set of states. We denote by $\mathrm{Trans}(Q)$ the monoid of mappings from Q to Q. The multiplication is the composition (in reverse order) of mappings: $(fg)(x) = g(f(x))$; and the unit element is the identity id_Q. We will use the fact that every finite monoid S can be realized as a submonoid of some $\mathrm{Trans}(Q)$ where $|Q| \leq |S|$. Indeed, it suffices to consider the right action of S over itself. More precisely, if we define $\chi(s) \in \mathrm{Trans}(S)$ by $\chi(s)(t) = ts$ then it is easy to see that $\chi : S \to \mathrm{Trans}(S)$ is an injective morphism.

We deduce that every aperiodic trace language can be recognized by some morphism $h : \mathbb{M}(\Sigma, I) \to S \subseteq \mathrm{Trans}(Q)$ where S is aperiodic. We show by induction on $(|Q|, |\Sigma|)$ that all languages recognized by h are expressible in $\mathrm{LTL}(\Sigma)$. We use the following well-founded lexicographic order on $\mathbb{N}^2$ for this induction: $(m, n) < (m', n')$ if and only if $m < m'$ or $m = m'$ and $n < n'$.

First, assume that $h(a) = \mathrm{id}_Q$ for all $a \in \Sigma$, which is in particular the case when $|Q| = 1$. If $L \subseteq \mathbb{R}$ is recognized by h then L is one of the sets $\emptyset$, $\mathbb{R}$, $\mathbb{M}$ or $\mathbb{R} \setminus \mathbb{M}$ which are respectively defined by the formulas $\bot$, $\top$, $\mathbf{F}\,\mathrm{stop}$, and $\mathbf{G}\,\mathbf{X}\,\top$. This shows the basis case of the induction.

Second, assume that $h(b) \neq \mathrm{id}_Q$ for some $b \in \Sigma$. The crucial observation here is that $h(b)$ is no permutation of Q. Indeed, since S is aperiodic, there exists some n such that $h(b)^n = h(b)^{n+1}$. If $h(b)$ were a permutation then it would be invertible and the last equality would imply $h(b) = \mathrm{id}_Q$, a contradiction. Hence, $h(b)(Q) = Q'$ for some $Q' \subset Q$ with $|Q'| < |Q|$.

We let $B = \Sigma \setminus \{b\}$ and we define two subsets of $\mathbb{M}$:

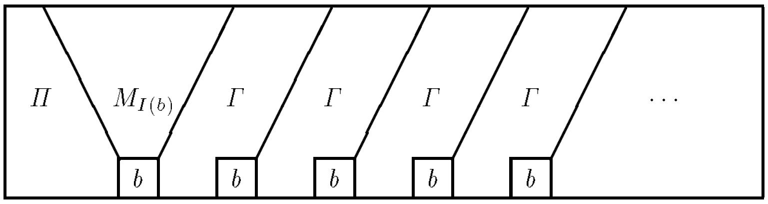
$$\begin{aligned} \Pi &= \{x \in \mathbb{M}_B \mid \max(x) \subseteq D(b)\}, \\ \Gamma &= \{x \in \mathbb{M}_B \mid \min(x) \subseteq D(b)\}. \end{aligned}$$

The notation Π is chosen since Πb are exactly the *pyramids* of $\mathbb{M}$ where the unique maximal element is b. It should be noted that $(\Pi b)^*$ and $(\Gamma b)^*$ are free submonoids of $\mathbb{M}$, both being infinitely generated if $D(b) \neq \{b\}$.

By Δ we denote the subset of real traces x which are either in $\mathbb{M}b$ or which can be factorized into an infinite product of *finite* traces such that all factors except the first belong to Γb, that is, $\Delta = \mathbb{M}b(\Gamma b)^\infty$. The set Δ, which plays a key-role, admits the following unambiguous decomposition:

$$\Delta = \Pi \mathbb{M}_{I(b)} b (\Gamma b)^\infty.$$

This decomposition is best visualized by the following picture; it is in some sense the guide for the modular construction of a formula defining the language $L \cap \Delta$.



The core of the proof is now the following proposition.

Proposition 5. *Let $L \subseteq \mathbb{R}$ be recognized by h. Then, $L \cap \Delta$ is expressible in* LTL(Σ).

The proof of this proposition will be given later in Section 7.1. Here we show how it is used in the proof of Theorem 2. We start with two corollaries.

Corollary 1. *Assume that (Σ, D) is connected and let $L \subseteq \mathbb{R}$ be recognized by h. Then the language $L \cap (\mathrm{Alphinf} = \Sigma)$ is expressible in* LTL(Σ).

Proof. Since (Σ, D) is connected, we have $(\mathrm{Alphinf} = \Sigma) \subseteq \Delta$. Therefore,

$$L \cap (\mathrm{Alphinf} = \Sigma) = (L \cap \Delta) \cap (\mathrm{Alphinf} = \Sigma).$$

By Proposition 5 we know that $L \cap \Delta$ is expressible in LTL(Σ) and we conclude easily since $(\mathrm{Alphinf} = \Sigma) = \mathrm{L}_{\mathbb{R}}(\wedge_{a \in \Sigma} \mathbf{G}\,\mathbf{F}\, a)$.

Corollary 2. *Let $c \in \Sigma$, $C = \Sigma \setminus \{c\}$ and let $L \subseteq \mathbb{R}$ be recognized by h. Then the language $L \cap (\mathrm{Alphinf} \subseteq C)$ is expressible in* LTL(Σ).

Proof. We claim that $L \cap (\mathrm{Alphinf} \subseteq C)$ is the finite union

$$\bigcup_{u,v \in S} \Big(\big((h^{-1}(u) \cap (\mathrm{Max} \subseteq \{b\})) \cdot (h^{-1}(v) \cap \mathbb{M}_B) \big) \cap (\mathrm{Max} \subseteq \{c\}) \Big) \cdot (L(uv) \cap \mathbb{R}_C)$$

We will now apply Lemma 3 twice in order to conclude the proof of Corollary 2. We fix some $u, v \in S$. First, by Propositions 2 and 4 we know that $K_2 = L(uv) \cap \mathbb{R}_C$ is recognized by the morphism $h\lceil_{\mathbb{M}_C}$ and $L_2 = h^{-1}(v) \cap \mathbb{M}_B$ is recognized by the morphism $h\lceil_{\mathbb{M}_B}$. By induction on the size of the alphabet, we deduce that L_2 and K_2 are expressible in LTL(B) and LTL(C) respectively. By Remark 2 they are also expressible in LTL(Σ).

Second, note that $(\mathrm{Max} \subseteq \{b\}) \subseteq \Delta \cup \{1\}$. Hence,

$$h^{-1}(u) \cap (\mathrm{Max} \subseteq \{b\}) = (h^{-1}(u) \cap (\Delta \cup \{1\})) \cap (\mathrm{Max} \subseteq \{b\}).$$

By Proposition 5, the language $L_1 = h^{-1}(u) \cap (\Delta \cup \{1\})$ is expressible in LTL(Σ). Applying once Lemma 3 we deduce first that

$$K_1 = (L_1 \cap (\mathrm{Max} \subseteq \{b\})) \cdot L_2 = (h^{-1}(u) \cap (\mathrm{Max} \subseteq \{b\})) \cdot (h^{-1}(v) \cap \mathbb{M}_B)$$

is expressible in $\mathrm{LTL}(\Sigma)$. Next, applying a second time Lemma 3 we deduce that $(K_1 \cap (\mathrm{Max} \subseteq \{c\})) \cdot K_2$ is expressible in $\mathrm{LTL}(\Sigma)$. Therefore, $L \cap (\mathrm{Alphinf} \subseteq C)$ is expressible in $\mathrm{LTL}(\Sigma)$.

In order to conclude the proof of Theorem 2, we use the following proposition.

Proposition 6. *Assume that (Σ, D) is non-connected and let $L \subseteq \mathbb{R}$ be recognized by h. Then L is expressible in* $\mathrm{LTL}(\Sigma)$.

Proof. Assume that $\Sigma = \Sigma_1 \cup \Sigma_2$ with $\Sigma_1 \times \Sigma_2 \subseteq I$ and let $\mathbb{M}_i = \mathbb{M}_{\Sigma_i}$, $\mathbb{R}_i = \mathbb{R}_{\Sigma_i}$ and $h_i = h{\restriction}_{M_i}$ for $i = 1, 2$. By Proposition 3, L is a finite union of products $L_1 \cdot L_2$ where $L_i \subseteq \mathbb{R}_i$ is recognized by h_i for $i = 1, 2$. By induction on the size of the alphabet, we deduce that L_1 and L_2 are expressible in $\mathrm{LTL}(\Sigma_1)$ and $\mathrm{LTL}(\Sigma_2)$ respectively. Using Lemma 1, we deduce that $L_1 \cdot L_2$ is expressible in $\mathrm{LTL}(\Sigma)$.

7.1 Proof of Proposition 5

Let us first show that the set $(\Gamma b)^\infty$ is expressible in $\mathrm{LTL}(\Sigma)$.

Lemma 4. *The real trace language $(\Gamma b)^\infty$ is expressed by the formula*

$$(\mathrm{Min} \subseteq D(b)) \wedge \Big(\mathrm{stop} \vee \mathbf{F}\langle b\rangle\mathrm{stop} \vee \mathbf{G}\,\mathbf{F}(\mathrm{Min} = b)\Big)$$

where we write simply $(\mathrm{Min} \subseteq D(b))$ *and* $(\mathrm{Min} = b)$ *instead of the corresponding* $\mathrm{LTL}(\Sigma)$ *formula.*

Now, recall that $h(b)(Q) = Q'$. Each $s \in h((\Gamma b)^*)$ maps the subset Q' to Q'. Hence we may define two subsets $T, T' \subseteq \mathrm{Trans}(Q')$ by $T = \{s{\restriction}_{Q'} \mid s \in h(\Gamma b)\}$ and $T' = \{s{\restriction}_{Q'} \mid s \in h((\Gamma b)^*)\}$. Since $h((\Gamma b)^*)$ is a submonoid of S, the set T' is a monoid. Moreover, the monoid T' is generated by T and is aperiodic since S is aperiodic.

By T^* we denote the free monoid generated by the finite set T (here T is viewed as an alphabet). Accordingly, T^∞ means the set of finite or infinite words over the alphabet T. The inclusion $T \subseteq T'$ induces a canonical morphism $e : T^* \to T'$ which is called the *evaluation*.

The mapping $\sigma : \Gamma b \to T$, defined by the restriction $\sigma(x) = h(x){\restriction}_{Q'}$, induces a morphism $\sigma : (\Gamma b)^* \to T^*$ between free monoids. The mapping σ is also extended to infinite sequences $\sigma : (\Gamma b)^\omega \to T^\omega$, so finally we have $\sigma : (\Gamma b)^\infty \to T^\infty$.

Since T' is a submonoid of $\mathrm{Trans}(Q')$ and $|Q'| < |Q|$, we may use induction (Although we might have $|T| > |\Sigma|$). More precisely, we may assume that every language $K \subseteq T^\infty$ which is recognized by the morphism e is expressible in $\mathrm{LTL}(T)$. The following lemma allows us to make use of this induction step.

Lemma 5. *Let $K \subseteq T^\infty$ be a word language expressible in* $\mathrm{LTL}(T)$. *Then the real trace language $\sigma^{-1}(K) \subseteq (\Gamma b)^\infty$ is expressible in* $\mathrm{LTL}(\Sigma)$.

Proof. The statement of the lemma corresponds to Lemmas 5 and 9 in [2]. The proof of Lemma 5 given there can be translated to our situation without any difficulty. It is therefore omitted.

Lemma 6. *Let $L \subseteq \mathbb{R}$ be recognized by h. Then $L \cap b(\Gamma b)^\infty$ is expressible in* LTL(Σ).

Proof. We define the language $K \subseteq T^\infty$ with respect to the language L by

$$K = \{\sigma(x) \in T^\infty \mid bx \in L \cap b(\Gamma b)^\infty\}.$$

It was shown in [2, Lem. 6][1] that $L \cap b(\Gamma b)^\infty = b\sigma^{-1}(K)$ and that K is recognized by the morphism $e : T^* \to T'$.

Now, it is easy to conclude the proof. Since $|Q'| < |Q|$ and K is recognized by $e : T^* \to T' \subseteq \mathrm{Trans}(Q')$, we know by induction that K is expressible in LTL(T). Using Lemma 5, we deduce that $\sigma^{-1}(K)$ is expressible in LTL(Σ). Therefore, $L \cap b(\Gamma b)^\infty = b\sigma^{-1}(K)$ is expressible in LTL(Σ).

We are now ready to complete the proof of Proposition 5. Let $L \subseteq \mathbb{R}$ be a real trace language recognized by the morphism h. We claim that

$$L \cap \Delta = \bigcup_{u,v \in S} (h^{-1}(u) \cap \Pi)(h^{-1}(v) \cap \mathbb{M}_{I(b)})(L(uv) \cap b(\Gamma b)^\infty).$$

Indeed, let $t = xyz$ with $x \in h^{-1}(u)$, $y \in h^{-1}(v)$ and $z \in L(uv)$ for some $u, v \in S$. Then $h(xy) = uv$ and we deduce from Proposition 4 that $t = xyz \in L$. Conversely, let $t \in (L \cap \Delta)$. Using the unambiguous factorization $\Delta = \Pi \mathbb{M}_{I(b)} b(\Gamma b)^\infty$, we can write $t = xyz$ with $x \in \Pi$, $y \in \mathbb{M}_{I(b)}$ and $z \in b(\Gamma b)^\infty$. If we let $u = h(x)$ and $v = h(y)$ then we get $z \in L(uv)$ which concludes the proof of the claim.

By Proposition 4, we know that $L(uv)$ is recognized by h, hence by Lemma 6 the language $L_3 = L(uv) \cap b(\Gamma b)^\infty \subseteq (\mathrm{Min} = b)$ is expressible in LTL(Σ). By Proposition 2, the languages $L_1 = h^{-1}(u) \cap \mathbb{M}_B$ and $L_2 = h^{-1}(v) \cap \mathbb{M}_B$ are recognized by the restriction of h to $\mathbb{M}_B$. Hence, by induction on the size of the alphabet, they are expressible in LTL(B), and also in LTL(Σ) by Remark 2. Since $\Pi \cup \mathbb{M}_{I(b)} \subseteq \mathbb{M}_B$, we have $h^{-1}(u) \cap \Pi = L_1 \cap \Pi$ and $h^{-1}(v) \cap \mathbb{M}_{I(b)} = L_2 \cap \mathbb{M}_{I(b)}$. Using Lemma 2, we deduce that

$$(L_1 \cap \Pi)(L_2 \cap \mathbb{M}_{I(b)})L_3 = (h^{-1}(u) \cap \Pi)(h^{-1}(v) \cap \mathbb{M}_{I(b)})(L(uv) \cap b(\Gamma b)^\infty)$$

is expressible in LTL(Σ).

Acknowledgement: We thank Alexander Rabinovich for stimulating discussions during the Computer Science Logic conference CSL'99 at Madrid.

[1] In [2] we used a slightly different notion of h-similarity, but the proof given there applies in fact to the definition as it is used here.

References

1. R. Alur, D. Peled, and W. Penczek. Model-checking of causality properties. In *Proceedings of LICS'95*, pages 90–100, 1995.
2. V. Diekert and P. Gastin. An expressively complete temporal logic without past tense operators for mazurkiewicz traces. In *Proceedings of CSL'99*, number 1683 in Lecture Notes in Computer Science, pages 188–203. Springer, 1999.
3. V. Diekert and G. Rozenberg, editors. *The Book of Traces.* World Scientific, Singapore, 1995.
4. W. Ebinger. *Charakterisierung von Sprachklassen unendlicher Spuren durch Logiken.* Dissertation, Institut für Informatik, Universität Stuttgart, 1994.
5. W. Ebinger and A. Muscholl. Logical definability on infinite traces. *Theoretical Computer Science*, 154:67–84, 1996.
6. D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Conference Record of the 12th ACM Symposium on Principles of Programming Languages*, pages 163–173, Las Vegas, Nev., 1980.
7. P. Gastin and A. Petit. Infinite traces. In V. Diekert and G. Rozenberg, editors, *The Book of Traces*, chapter 11, pages 393–486. World Scientific, Singapore, 1995.
8. J. A. W. Kamp. *Tense Logic and the Theory of Linear Order.* PhD thesis, University of California, Los Angeles, Calif., 1968.
9. R. Meyer and A. Petit. Expressive completeness of LTrL on finite traces: an algebraic proof. In *Proceedings of STACS'98*, number 1373 in Lecture Notes in Computer Science, pages 533–543, 1998.
10. M. Mukund and P. S. Thiagarajan. Linear time temporal logics over Mazurkiewicz traces. In *Proceedings of the 21th MFCS, 1996*, number 1113 in Lecture Notes in Computer Science, pages 62–92. Springer, 1996.
11. P. Niebert. A ν-calculus with local views for sequential agents. In *Proceedings of the 20th MFCS, 1995*, number 969 in Lecture Notes in Computer Science, pages 563–573. Springer, 1995.
12. W. Penczek. Temporal logics for trace systems: On automated verification. *International Journal of Foundations of Computer Science*, 4:31–67, 1993.
13. W. Penczek and R. Kuiper. Traces and logic. In V. Diekert and G. Rozenberg, editors, *The Book of Traces*, chapter 10, pages 307–390. World Scientific, Singapore, 1995.
14. R. Ramanujam. Locally linear time temporal logic. In *Proceedings of LICS'96*, Lecture Notes in Computer Science, pages 118–128, 1996.
15. P. S. Thiagarajan. A trace based extension of linear time temporal logic. In *Proceedings of LICS'94*, pages 438–447, 1994.
16. P. S. Thiagarajan. A trace consistent subset of PTL. In *Proceedings of CONCUR'95*, number 962 in Lecture Notes in Computer Science, pages 438–452, 1995.
17. P. S. Thiagarajan and I. Walukiewicz. An expressively complete linear time temporal logic for Mazurkiewicz traces. In *Proceedings of LICS'97*, 1997.
18. Th. Wilke. Classifying discrete temporal properties. In *Proceedings of STACS'99*, number 1563 in Lecture Notes in Computer Science, pages 32–46. Springer, 1999.