An Elementary Expressively Complete Temporal Logic for Mazurkiewicz Traces*

Paul Gastin[1] and Madhavan Mukund[2]

[1] LIAFA, Université Paris 7, 2, place Jussieu, F-75251 Paris Cedex 05
Fax: 33 1 44 27 68 49 — Paul.Gastin@liafa.jussieu.fr
[2] Chennai Mathematical Institute, 92 G N Chetty Road, Chennai 600 017, India
Fax: 91 44 823 3671 — madhavan@cmi.ac.in

Abstract. In contrast to the classical setting of sequences, no temporal logic has yet been identified over Mazurkiewicz traces that is equivalent to the first-order theory of traces and yet admits an elementary decision procedure. In this paper, we describe a local temporal logic over traces that is expressively complete and whose satisfiability problem is in PSPACE. Contrary to the situation for sequences, past modalities are essential for such a logic. A somewhat unexpected corollary is that first-order logic with three variables is expressively complete for traces.
Keywords Temporal logics, Mazurkiewicz traces, concurrency

1 Introduction

Linear-time temporal logic (LTL) [20] has established itself as a useful formalism for specifying the interleaved behaviour of reactive systems. To combat the combinatorial blow-up involved in describing computations of concurrent systems in terms of interleavings, there has been a lot of interest in using temporal logic more directly on labelled partial orders.

Mazurkiewicz traces [15] are labelled partial orders generated by dependence alphabets of the form (Σ, D), where D is a *dependence* relation over Σ. If $(a, b) \notin D$, a and b are deemed to be independent actions that may occur concurrently. Traces are a natural formalism for describing the behaviour of static networks of communicating agents as modelled, for instance, by Petri nets or communicating finite-state machines [34].

LTL over Σ-labelled sequences is equivalent to $\mathrm{FO}_\Sigma(<)$, the first-order theory of Σ-labelled linear orders [12] and thus defines the class of aperiodic languages over Σ. Though $\mathrm{FO}_\Sigma(<)$ permits assertions about both the past and the future, future modalities suffice for establishing the expressive completeness of LTL with respect to $\mathrm{FO}_\Sigma(<)$ [10]. From a practical point of view, a finite-state program may be checked against an LTL specification relatively efficiently [3, 26, 27].

Though a number of LTL-like temporal logics have been proposed for reasoning directly about Mazurkiewicz traces, it has proved elusive to define a tractable

* Work done while the second author was visiting LIAFA, Université Paris 7. Partial support of CEFIPRA-IFCPAR Project 2102-1 (ACSMV) is gratefully acknowledged.

logic that is expressively complete with respect to $\mathrm{FO}_{(\Sigma,D)}(<)$, the first order theory of traces over (Σ, D). The first expressively complete temporal logic over traces was described in [23]. The result was refined in [4, 6] to show expressive completeness without past modalities, using an extension of the proof technique developed for LTL in [32, 33]. Formulae in both these logics are defined at *global states* (maximal antichains). Unfortunately, reasoning at the level of global states makes the complexity of deciding satisfiability non-elementary [29].

Computational tractability seems to require interpreting formulae at *local states*—effectively at individual events. This was the approach followed (sometimes indirectly) in early temporal logics over traces [1, 17, 21, 24]. However, though these logics admit relatively efficient model-checking algorithms, they are not known to be expressively complete. In fact, the logic TLC described in [1] is not even first-order definable.

Recently, a local temporal logic has been described that is expressively complete for the subclass of Mazurkiewicz traces where the dependence graph of the underlying alphabet has a series-parallel structure [5]. Unfortunately, this logic *cannot* be expressively complete over all traces—the logic uses only future modalities and there exist two first-order inequivalent traces that cannot be distinguished based only on future modalities [31]. This marks a departure from the theory of LTL over sequences.

In this paper, we define a new local temporal logic over traces that is expressively complete and whose satisfiability problem is in PSPACE. The observation in [31] about the shortcoming of future modalities necessitates the introduction of past modalities in our logic. As in [4, 5], we show expressive completeness using an extension to traces of the proof technique introduced in [32, 33] for LTL over sequences. However, the earlier proofs exploit the fact that the past cannot affect the truth of a formula. The introduction of past modalities in our logic requires the use of new techniques for "relativizing" formulae placed in new past contexts. The satisfiability problem for our logic is settled using two-way alternating automata, extending techniques from [1, 5].

A corollary of our main result is that $\mathrm{FO}^3_{(\Sigma,D)}(<)$, the subclass of first-order logic formulae with at most three variables, is expressively complete over traces, thus extending the same result for sequences. This is somewhat surprising because intuition suggests that the number of variables required should be proportional to the width of a trace. (This result has been independently established by Walukiewicz [30].)

The paper is organized as follows. We begin with some preliminaries about traces. In Section 3 we define our new temporal logic. The next section describes a syntactic partition of traces that we use in our proof. We describe how to show expressive completeness of our logic in Section 5. The decision procedure for satisfiability is sketched briefly in Section 6. We conclude with a discussion of open questions.

2 Preliminaries

We briefly recall some notions about Mazurkiewicz traces (see [7] for background). A *dependence alphabet* is a pair (Σ, D) where the alphabet Σ is a finite set of actions and the *dependence relation* $D \subseteq \Sigma \times \Sigma$ is reflexive and symmetric. The *independence relation* I is the complement of D. For $A \subseteq \Sigma$, the set of letters independent of A is denoted by $I(A) = \{b \in \Sigma \mid (a, b) \in I \text{ for all } a \in A\}$ and the set of letters depending on (some action in) A is denoted by $D(A) = \Sigma \setminus I(A)$.

A *Mazurkiewicz trace* is a labelled partial order $t = [V, \leq, \lambda]$ where V is a set of vertices labelled by $\lambda : V \to \Sigma$ and $\leq$ is a partial order over V satisfying the following conditions: For all $x \in V$, the downward set ${\downarrow}x = \{y \in V \mid y \leq x\}$ 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 immediate successor relation in t.

The *alphabet* of a 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 $t = [V, \leq, \lambda] \in \mathbb{R}$, we define $\min(t) \subseteq V$ as the set of all minimal vertices of t. We can also read $\min(t) \subseteq \Sigma$ as the set of labels of the minimal vertices of t. It will be clear from the context what we actually mean.

Let $t_1 = [V_1, \leq_1, \lambda_1]$ and $t_2 = [V_2, \leq_2, \lambda_2]$ be a pair of traces such that $\mathrm{alphinf}(t_1) \times \mathrm{alph}(t_2) \subseteq I$. We then define the concatenation of t_1 and t_2 to be $t_1 \cdot t_2 = [V, \leq, \lambda]$ where $V = V_1 \cup V_2$ (assuming wlog that $V_1 \cap V_2 = \emptyset$), $\lambda = \lambda_1 \cup \lambda_2$ and $\leq$ is the transitive closure of the relation $\leq_1 \cup \leq_2 \cup (V_1 \times V_2 \cap \lambda^{-1}(D))$. The set of finite traces is then a monoid, denoted $\mathbb{M}(\Sigma, D)$ or simply $\mathbb{M}$, with the empty trace $1 = (\emptyset, \emptyset, \emptyset)$ as unit.

Here is some useful notation for subclasses of traces. For $C \subseteq \Sigma$, let $\mathbb{R}_C = \{t \in \mathbb{R} \mid \mathrm{alph}(x) \subseteq C\}$ and $\mathbb{M}_C = \mathbb{M} \cap \mathbb{R}_C$. Also, $(\mathrm{alph} = C) = \{t \in \mathbb{R} \mid \mathrm{alph}(t) = C\}$, $(\mathrm{alphinf} = C) = \{t \in \mathbb{R} \mid \mathrm{alphinf}(t) = C\}$ and $(\min = C) = \{t \in \mathbb{R} \mid \min(t) = C\}$. For $A, C \subseteq \Sigma$, we set $\mathbb{R}_C^A = \mathbb{R}_C \cap (\mathrm{alphinf} = A)$. Observe that $\mathbb{M}_C = \mathbb{R}_C^{\emptyset}$.

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

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

where $a \in \Sigma$ and $x, y \in \mathrm{Var}$ are first order variables. Given a trace $t = [V, \leq, \lambda]$ and a valuation $\sigma : \mathrm{Var} \to V$, $t, \sigma \models \varphi$ denotes that t satisfies φ under σ. We interpret each predicate P_a by the set $\{x \in V \mid \lambda(x) = a\}$ and the relation $<$ as the strict partial order relation of t. The semantics then lifts to all formulas as usual. Since the meaning of a closed formula (sentence) φ is independent of the valuation σ, we can associate with each sentence φ 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 a sentence $\varphi \in \mathrm{FO}_\Sigma(<)$ such that $L = \mathcal{L}(\varphi)$. We denote by $\mathrm{FO}_{(\Sigma,D)}(<)$ the set of trace languages $L \subseteq \mathbb{R}(\Sigma, D)$ that are expressible in $\mathrm{FO}_\Sigma(<)$. For $n > 0$, $\mathrm{FO}^n_\Sigma(<)$ denotes the set of formulae with at most n distinct variables (note that each variable may be bound and reused several times).

We use the algebraic notion of recognizability. Let $h : \mathbb{M} \to S$ be a morphism to a finite monoid S. For $t, u \in \mathbb{R}$, we say that t and u are h-similar, denoted

$t \sim_h u$, if either $t, u \in \mathbb{M}$ and $h(t) = h(u)$ or t and u have infinite factorizations in non-empty finite traces $t = t_1 t_2 \cdots$, $u = u_1 u_2 \cdots$ with $h(t_i) = h(u_i)$ for all i. The transitive closure $\approx_h$ of $\sim_h$ is an equivalence relation. Since S is finite, this equivalence relation is of finite index with at most $|S|^2 + |S|$ equivalence classes. A trace language $L \subseteq \mathbb{R}$ is *recognized* by h if it is saturated by $\approx_h$ (or equivalently by $\sim_h$), i.e., $t \in L$ implies $[t]_{\approx_h} \subseteq L$ for all $t \in \mathbb{R}$.

Let $L \subseteq \mathbb{R}$ be recognized by a morphism $h : \mathbb{M} \to S$. For $B \subseteq \Sigma$, $L \cap \mathbb{M}_B$ and $L \cap \mathbb{R}_B$ are recognized by $h\restriction_{\mathbb{M}_B}$ the restriction of h to $\mathbb{M}_B$.

A finite monoid S is *aperiodic* if there is an $n \geq 0$ such that $s^n = s^{n+1}$ for all $s \in S$. A trace language $L \subseteq \mathbb{R}$ is *aperiodic* if it is recognized by some morphism to a finite and aperiodic monoid.

First-order definability coincides with aperiodicity for traces.

Theorem 1 ([9, 8]). *A language $L \subseteq \mathbb{R}(\Sigma, D)$ is expressible in $\mathrm{FO}_\Sigma(<)$ if and only if it is aperiodic.*

3 Local temporal logic

Our local temporal logic over traces consists of two types of formulae: *internal formulae*, evaluated at arbitrary events within a trace, and *initial formulae*, asserting properties of minimal events of the trace. Internal formulae only talk about the structure within a connected portion of a trace. Initial formulae permit us to combine local assertions across disjoint components and form "global" assertions about the structure of the trace.

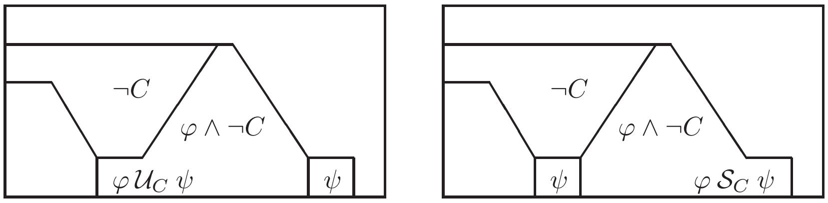
We denote by $\mathrm{LocTL}^{\mathrm{i}}_\Sigma$ the set of internal formulae over the alphabet Σ. They are given by the following syntax:

$$\varphi ::= a \in \Sigma \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{EX}\,\varphi \mid \varphi\,\mathcal{U}_C\,\varphi, C \subseteq \Sigma \mid \mathsf{EY}\,\varphi \mid \varphi\,\mathcal{S}_C\,\varphi, C \subseteq \Sigma$$

Let $t = [V, \leq, \lambda] \in \mathbb{R}$ be a finite or infinite trace and let $x \in V$ be some vertex of t. We write $t, x \models \varphi$ to denote that trace t at node x satisfies the formula $\varphi \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma$. This is defined inductively as follows:

$$\begin{array}{lll}
t, x \models a & \text{if} & \lambda(x) = a \\
t, x \models \neg\varphi & \text{if} & t, x \not\models \varphi \\
t, x \models \varphi \vee \psi & \text{if} & t, x \models \varphi \text{ or } t, x \models \psi \\
t, x \models \mathsf{EX}\,\varphi & \text{if} & \exists y.\ x \lessdot y \text{ and } t, y \models \varphi \\
t, x \models \mathsf{EY}\,\varphi & \text{if} & \exists y.\ y \lessdot x \text{ and } t, y \models \varphi \\
t, x \models \varphi\,\mathcal{U}_C\,\psi & \text{if} & \exists z \geq x.\ [t, z \models \psi \text{ and } \forall y.\ (x \leq y < z) \Rightarrow t, y \models \varphi \text{ and} \\
 & & \qquad\qquad \forall y.\ (\lambda(y) \in C \text{ and } y \leq z) \Rightarrow y \leq x] \\
t, x \models \varphi\,\mathcal{S}_C\,\psi & \text{if} & \exists z \leq x.\ [t, z \models \psi \text{ and } \forall y.\ (z < y \leq x) \Rightarrow t, y \models \varphi \text{ and} \\
 & & \qquad\qquad \forall y.\ (\lambda(y) \in C \text{ and } y \leq x) \Rightarrow y \leq z]
\end{array}$$

The modalities EX and EY are existential extensions of the next state and previous state operators X and Y of LTL. The semantics of $\mathcal{U}_C$ and $\mathcal{S}_C$ are depicted below.



The modality $\mathcal{U}_C$ extends the "universal" until operator $\mathcal{U}$ defined in [5] with an alphabetic *filter* that restricts the actions performed while fulfilling an until requirement. If $t, x \models \varphi \mathcal{U}_C \psi$ then no C-actions are performed when going from x to the node satisfying ψ. Since the filter also applies to events that are concurrent to x, the modality $\mathcal{U}_C$ no longer looks purely at the future of an event. Similarly, the modality $\mathcal{S}_C$ is an extenstion of the LTL since operator with a filter.

Past modalities are essential, as indicated by the following example from [31], where the dependence relation is $a - b - c - d$. These two traces are not first-order equivalent but are bisimilar at the level of events and thus cannot be distinguished by purely future modalities.

$$\begin{array}{lll} a \to b \to c \to b \to c \cdots & \qquad & d \to c \to b \to c \to b \cdots \\ \quad\;\; \uparrow & & \quad\;\; \uparrow \\ d \to c & & a \to b \end{array}$$

As usual, we can derive useful operators. Using unfiltered versions of until and since, $\varphi \mathcal{U} \psi = \varphi \mathcal{U}_\emptyset \psi$ and $\varphi \mathcal{S} \psi = \varphi \mathcal{S}_\emptyset \psi$, we recover the classical modalities *eventually in the future* $\mathsf{F}\,\varphi = \top\,\mathcal{U}\,\varphi$, *eventually in the past* $\mathsf{F}^{-1}\,\varphi = \top\,\mathcal{S}\,\varphi$, *always in the future* $\mathsf{G}\,\varphi = \neg\,\mathsf{F}\,\neg\varphi$, and *always in the past* $\mathsf{G}^{-1}\,\varphi = \neg\,\mathsf{F}^{-1}\,\neg\varphi$. We also use some filtered versions of these modalities: $\mathsf{F}_b\,\varphi = \top\,\mathcal{U}_b\,\varphi$ (eventually in the future but above the same b's), $\mathsf{G}_b\,\varphi = \neg\,\mathsf{F}_b\,\neg\varphi$ (for all future vertices that are above the same b's). The existence of a successor (resp. a predecessor) above the same b's can be written as follows:

$$\begin{aligned} \mathsf{EX}_b\,\varphi &= \bigvee_{d\neq b} \mathsf{EX}(d \wedge \varphi) \wedge \top\,\mathcal{U}_b\,d = \bigvee_{c} c \wedge \mathsf{EX}(\neg b \wedge \varphi \wedge \top\,\mathcal{S}_b\,c), \\ \mathsf{EY}_b\,\varphi &= \bigvee_{c} \mathsf{EY}(c \wedge \varphi) \wedge \top\,\mathcal{S}_b\,c \;\; = \bigvee_{d\neq b} d \wedge \mathsf{EY}(\varphi \wedge \top\,\mathcal{U}_b\,d). \end{aligned}$$

Finally, $\mathsf{F}^\infty\,a = \mathsf{F}\,a \wedge \mathsf{G}(a \Rightarrow \mathsf{EX}\,\mathsf{F}\,a)$ expresses the existence of infinitely many vertices labelled with a above the current vertex. The filtered version of this formula, $\mathsf{F}^\infty_b\,a = \mathsf{F}_b\,a \wedge \mathsf{G}_b(a \Rightarrow \mathsf{EX}_b\,\mathsf{F}_b\,a)$, requires in addition that the vertices labelled with a are above the same b's.

For $A \subseteq \Sigma$, let A denote the internal formula $\bigvee_{a\in A} a$. An *alphabetic since* is a formula of the form $A\,\mathcal{S}_C\,a$ for $A, C \subseteq \Sigma$ and $a \in \Sigma$.

We turn now to initial formulae. These are boolean combinations of formulae $\mathsf{EM}\,\varphi$, asserting the existence of a minimal vertex in a trace that satisfies the internal formula φ. More precisely, we define LocTL_Σ, the set of initial formulae over the alphabet Σ, as follows:

$$\alpha ::= \bot \mid \mathsf{EM}\,\varphi, \varphi \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma \mid \neg\alpha \mid \alpha \vee \alpha$$

The semantics is given by:

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

An initial formula $\alpha \in \mathrm{LocTL}_\Sigma$ defines the trace language $\mathcal{L}(\alpha) = \{t \in \mathbb{R} \mid t \models \alpha\}$. We can then express various alphabetic properties using initial formulae: $\mathcal{L}(\mathsf{EM}\,a) = \{t \in \mathbb{R} \mid a \in \min(t)\}$, $\mathcal{L}(\mathsf{EM}\,\mathsf{F}\,a) = \{t \in \mathbb{R} \mid a \in \mathrm{alph}(t)\}$, and $\mathcal{L}(\mathsf{EM}\,\mathsf{F}^\infty\,a) = \{t \in \mathbb{R} \mid a \in \mathrm{alphinf}(t)\}$. Therefore, for $C \subseteq \Sigma$, trace languages such as $(\mathrm{alph} = C)$, $(\mathrm{alphinf} = C)$ and $(\min = C)$ are expressible in LocTL_Σ.

The following result is immediate from the definition of LocTL_Σ.

Proposition 2. *If a trace language is expressible in* LocTL_Σ*, then it is expressible in* $\mathrm{FO}^3_\Sigma(<)$.

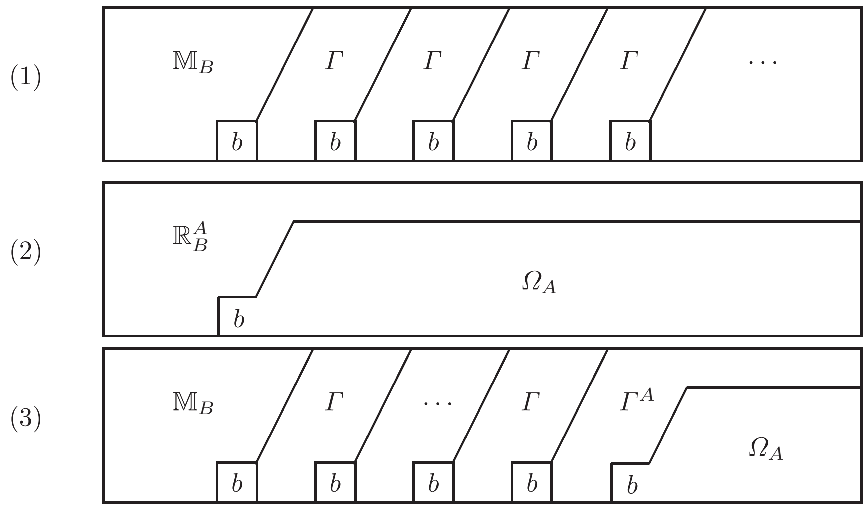
4 Decomposition of traces

The proof of our main result is a case analysis based on partitioning the set of traces according to the structure of the trace. Fix a letter $b \in \Sigma$ and set $B = \Sigma \setminus \{b\}$. Using the notation introduced at the end of Section 2, let $\Gamma^A = \{t \in \mathbb{R}^A_B \mid \min(t) \subseteq D(b)\}$, $\Gamma = \Gamma^\emptyset$, and $\Omega_A = \{t \in \mathbb{R}_{I(A)} \mid \min(t) \subseteq \{b\}\}$.

Each trace $t \in \mathbb{R}$ has a unique factorization $t = t_0 b t_1 b t_2 \cdots$ with $t_0 \in \mathbb{R}_B$ and $t_i \in \mathbb{R}_B \cap (\min \subseteq D(b))$ for all $i > 0$. We obtain the following partition of $\mathbb{R}$.

$$\mathbb{R} = \mathbb{M}_B(b\Gamma)^\infty \uplus \left(\bigcup_{A \neq \emptyset} \mathbb{R}^A_B \Omega_A\right) \uplus \left(\bigcup_{A \neq \emptyset} \mathbb{M}_B(b\Gamma)^* b\Gamma^A \Omega_A\right)$$

If all t_i are finite, then we are in the first block. If $t_0 \notin \mathbb{M}$, then we are in the second block, Otherwise, we are in the last block. Note that $\mathbb{M}_B(b\Gamma)^* = \mathbb{M}$. The three possibilities are depicted below.



The following two results will allow us to use this decomposition effectively in proving the expressive completeness of our logic.

Lemma 3. *All these sets are expressible in* LocTL_Σ*. More precisely,*

1. *The sets* $\mathbb{M}_B(b\Gamma)^*$ *and* $\mathbb{M}_B(b\Gamma)^\omega$ *are expressible in* LocTL_Σ*.*
2. *The set* $\mathbb{R}_B^A \cdot \Omega_A = \mathcal{L}(\alpha)$ *where*

$$\alpha = \left(\bigwedge_{a\in A} \mathsf{EM}(\neg b \wedge \mathsf{F}_b^\infty\, a)\right) \wedge \left(\bigwedge_{a\notin A} \neg\,\mathsf{EM}(\neg b \wedge \mathsf{F}_b^\infty\, a)\right).$$

3. *The set* $\mathbb{M}_B(b\Gamma)^* b\Gamma^A\Omega_A = \mathbb{M}b\Gamma^A\Omega_A = \mathcal{L}(\beta)$ *where*

$$\beta = \bigvee_{C\subseteq\Sigma}\left((\mathrm{alphinf} = C) \wedge \mathsf{EM}\,\mathsf{F}\left(b \wedge \bigwedge_{c\in C} \mathsf{F}^\infty\, c \wedge \bigwedge_{a\in A} \mathsf{F}_b^\infty\, a \wedge \bigwedge_{a\notin A} \neg\,\mathsf{F}_b^\infty\, a\right)\right).$$

   Note that "the" b *in* $\mathbb{M}b\Gamma^A\Omega_A$ *is characterized by the formula* $b \wedge \mathsf{F}_b^\infty\, a$*, where* a *is any letter in* A*.*

Proof. (1) Indeed, $\mathbb{M}_B(b\Gamma)^* = (\mathrm{alph} = \emptyset)$ and $\mathbb{M}_B(b\Gamma)^\omega = \bigcup_C(\mathrm{alphinf} = C)$ where the union ranges over all $C \subseteq \Sigma$ such that $b \in C$ and C is connected.

(2) For all $t \in \mathbb{R}$, there is a unique decomposition $t = t_1t_2$ with $t_1 \in \mathbb{R}_B$ and $\min(t_2) \subseteq \{b\}$. We have $\mathrm{alphinf}(t_1) \times \mathrm{alph}(t_2) \subseteq I$. Moreover, $a \in \mathrm{alphinf}(t_1)$ iff $t \models \mathsf{EM}(\neg b \wedge \mathsf{F}_b^\infty\, a)$. Therefore, $t \models \beta$ iff $t_1 \in \mathbb{R}_B^A$ and $t_2 \in \Omega_A$.

(3) let $t = t_1bt_2t_3$ with $t_1 \in \mathbb{M}$, $t_2 \in \Gamma^A$ and $t_3 \in \Omega_A$. Let $C = \mathrm{alphinf}(t)$ and $x = \min(bt_2t_3) \in t$. Since $t_1 \in \mathbb{M}$, we have $C = \mathrm{alphinf}(bt_2t_3)$ and we get $t, x \models \bigwedge_{c\in C} \mathsf{F}^\infty\, c$. We have, $a \in \mathrm{alphinf}(t_2)$ iff $t, x \models \mathsf{F}_b^\infty\, a$. We deduce that $t_x \models \bigwedge_{a\in A} \mathsf{F}_b^\infty\, a \wedge \bigwedge_{a\notin A} \neg\,\mathsf{F}_b^\infty\, a$. Therefore, $t \models \alpha$.

Conversely, assume that $t \models \alpha$. Let $C = \mathrm{alphinf}(t)$ and $x \in t$ with $t, x \models \left(b \wedge \bigwedge_{c\in C} \mathsf{F}^\infty\, c \wedge \bigwedge_{a\in A} \mathsf{F}_b^\infty\, a \wedge \bigwedge_{a\notin A} \neg\,\mathsf{F}_b^\infty\, a\right)$.
Let t_1 be the prefix of t defined by $\overline{\uparrow x}$. Since $\mathrm{alphinf}(t) = C$ and $t, x \models \bigwedge_{c\in C} \mathsf{F}^\infty\, c$ we deduce that $t_1 \in \mathbb{M}$. Write $t = t_1bt_2t_3$ with $t_2 \in \mathbb{R}_B$ and $\min(t_3) \subseteq \{b\}$. Then, $a \in \mathrm{alphinf}(t_2)$ iff $t, x \models \mathsf{F}_b^\infty\, a$. Therefore, $\mathrm{alphinf}(t_2) = A$ and $t_2 \in \Gamma_A$. Finally, $\mathrm{alphinf}(t_2) \times \mathrm{alph}(t_3) \subseteq I$, hence $t_3 \in \Omega_A$. $\square$

Lemma 4. *Let* $A \subseteq \Sigma$ *and let* $L \subseteq \mathbb{R}$ *be a trace language recognized by a morphism* h *from* $\mathbb{M}$ *into a finite monoid* S*. Then,*

$$\begin{array}{lll}
(1) \quad L \cap \mathbb{M}_B(b\Gamma)^\infty & = & \bigcup_{\textit{finite}} (L_1 \cap \mathbb{M}_B) \cdot (L_2 \cap (b\Gamma)^\infty) \\
(2) \quad L \cap \mathbb{R}_B^A\Omega_A & = & \bigcup_{\textit{finite}} (L_1 \cap \mathbb{R}_B^A) \cdot (L_2 \cap \Omega_A) \\
(3) \quad L \cap \mathbb{M}b\Gamma^A\Omega_A & = & \bigcup_{\textit{finite}} (L_1 \cap \mathbb{M}_B)(L_2 \cap (b\Gamma)^*)b(L_3 \cap \Gamma^A)(L_4 \cap \Omega_A)
\end{array}$$

where the trace languages $L_i \subseteq \mathbb{R}$ *are recognized by* h*.*

Proof. (1) follows from the formula:

$$L \cap \mathbb{M}_B (b\Gamma)^\infty = \bigcup ([x]_{\approx_h} \cap \mathbb{M}_B) \cdot ([y]_{\approx_h} \cap (b\Gamma)^\infty)$$

where the union ranges over all $xy \in L$ with $x \in \mathbb{M}_B$ and $y \in (b\Gamma)^\infty$.

The inclusion $\subseteq$ is clear. Conversely, let $x, x' \in \mathbb{M}_B$ and $y, y' \in (b\Gamma)^\infty$ with $xy \in L$, $x \sim_h x'$ and $y \sim_h y'$. The case y finite is simpler, hence we assume y infinite: $y = y_1 y_2 \cdots$, $y' = y'_1 y'_2 \cdots$ with $h(y_i) = h(y'_i)$ for all $i \geq 1$. Then, $xy = (xy_1)y_2 \cdots$, $x'y' = (x'y'_1)y'_2 \cdots$ and $h(xy_1) = h(x'y'_1)$. Therefore, $x'y' \sim_h xy \in L$ and $x'y' \in L$. We deduce by induction that if $x \approx_h x'$ and $y \approx_h y'$ then $x'y' \approx_h xy$ and $x'y' \in L$.

(2) follows from the formula:

$$L \cap \mathbb{R}^A_B \Omega_A = \bigcup ([x]_{\approx_h} \cap \mathbb{R}^A_B) \cdot ([y]_{\approx_h} \cap \Omega_A)$$

where the union ranges over all $xy \in L$ with $x \in \mathbb{R}^A_B$ and $y \in \Omega_A$.

The inclusion $\subseteq$ is clear. Conversely, let $x, x' \in \mathbb{R}^A_B$ and $y, y' \in \Omega_A$ with $xy \in L$, $x \sim_h x'$ and $y \sim_h y'$. The cases x or y finite are simpler, hence we assume x, y infinite: $x = x_1 x_2 \cdots$, $x' = x'_1 x'_2 \cdots$ with $h(x_i) = h(x'_i)$ for all $i \geq 1$, $y = y_1 y_2 \cdots$, $y' = y'_1 y'_2 \cdots$ with $h(y_i) = h(y'_i)$ for all $i \geq 1$, and wlog $\mathrm{alph}(x_i) \subseteq A$ and $\mathrm{alph}(x'_i) \subseteq A$ for all $i > 1$. Then, $xy = (x_1 y_1)(x_2 y_2) \cdots$, $x'y' = (x'_1 y'_1)(x'_2 y'_2) \cdots$ and $h(x_i y_i) = h(x'_i y'_i)$ for all $i \geq 1$. Therefore, $x'y' \sim_h xy \in L$ and $x'y' \in L$. We deduce by induction that if $x \approx_h x'$ and $y \approx_h y'$ then $x'y' \approx_h xy$ and $x'y' \in L$.

(3) follows from the formula:

$$L \cap \mathbb{M} b \Gamma^A \Omega_A = \bigcup ([w]_{\approx_h} \cap \mathbb{M}_B)([x]_{\approx_h} \cap (b\Gamma)^*) b([y]_{\approx_h} \cap \Gamma^A)([z]_{\approx_h} \cap \Omega_A)$$

where the union ranges over all $wxbyz \in L$ with $w \in \mathbb{M}_B$, $x \in (b\Gamma)^*$, $y \in \Gamma^A$ and $y \in \Omega_A$.

The inclusion $\subseteq$ is clear. Conversely, let $w, w' \in \mathbb{M}_B$, $x, x' \in (b\Gamma)^*$, $y, y' \in \Gamma^A$ and $z, z' \in \Omega_A$ with $wxbyz \in L$, $w \sim_h w'$, $x \sim_h x'$, $y \sim_h y'$ and $z \sim_h z'$. The cases y or z finite are simpler, hence we assume y, z infinite: $y = y_1 y_2 \cdots$, $y' = y'_1 y'_2 \cdots$ with $h(y_i) = h(y'_i)$ for all $i \geq 1$, $z = z_1 z_2 \cdots$, $z' = z'_1 z'_2 \cdots$ with $h(z_i) = h(z'_i)$ for all $i \geq 1$, and wlog $\mathrm{alph}(y_i) \subseteq A$ and $\mathrm{alph}(y'_i) \subseteq A$ for all $i > 1$.

We have $wxbyz = (wxby_1 z_1)(y_2 z_2) \cdots$, $w'x'by'z' = (w'x'by'_1 z'_1)(y'_2 z'_2) \cdots$, $h(wxby_1 z_1) = h(w'x'by'_1 z'_1)$, and $h(y_i z_i) = h(y'_i z'_i)$ for all $i > 1$. Therefore, $w'x'by'z' \sim_h wxbyz \in L$ and $w'x'by'z' \in L$. We deduce by induction that if $w \approx_h w'$, $x \approx_h x'$, $y \approx_h y'$ and $z \approx_h z'$ then $w'x'by'z' \approx_h wxbyz$ and $w'x'by'z' \in L$. ☐

5 Expressive completeness

We now establish that our local temporal logic for traces is expressively complete with respect to $\mathrm{FO}_\Sigma(<)$. In light of Theorem 1, it suffices for us to establish the following.

Theorem 5. *Every aperiodic trace language $L \subseteq \mathbb{R}$ is expressible in* LocTL_Σ.

From Proposition 2 and Theorems 1 and 5, we deduce immediately

Corollary 6. *Let $L \subseteq \mathbb{R}$ be a trace language. The following are equivalent*

1. *L is aperiodic,*
2. *L is expressible in* LocTL_Σ,
3. *L is expressible in* $\mathrm{FO}^3_\Sigma(<)$,
4. *L is expressible in* $\mathrm{FO}_\Sigma(<)$.

The rest of this section is devoted to the proof of Theorem 5. We start with a trace language $L \subseteq \mathbb{R}$ recognized by a morphism h from $\mathbb{M}$ to a finite aperiodic monoid S. We use induction on $|\Sigma|$.

The base case is when $\Sigma = \{a\}$ is a singleton. Then L is either a finite set or the union of a finite set and a set of the form $a^n a^*$ for some $n \geq 0$. In both cases, we see that L is expressible in LocTL_Σ. For instance, $a^3 a^*$ is expressed by the formula $\mathsf{EM}\,\mathsf{EX}\,\mathsf{EX}\,\top$. Moreover, $\{a^3\} = a^3 a^* \setminus a^4 a^*$.

For the induction step, with $|\Sigma| > 1$, we fix a letter $b \in \Sigma$ and use the decomposition introduced in Section 4. Therefore, we have

$$L = \Big(L \cap \mathbb{M}_B (b\Gamma)^\infty\Big) \uplus \Big(L \cap \bigcup_{A \neq \emptyset} \mathbb{R}_B^A \Omega_A\Big) \uplus \Big(L \cap \bigcup_{A \neq \emptyset} \mathbb{M}_B (b\Gamma)^* b \Gamma^A \Omega_A\Big).$$

We show separately that the three languages above are expressible in LocTL_Σ. The 3 cases are handled in Propositions 9, 15 and 17 below.

We start by a simple result that is used in all three cases. Recall that by $(\min \subseteq \{b\})$ we denote the trace language $\{t \in \mathbb{R} \mid \min(t) \subseteq \{b\}\}$. Note that $\mathbb{R} = \mathbb{R}_B \cdot (\min \subseteq \{b\})$.

Proposition 7. *Let $L \subseteq \mathbb{R}$ be a trace language recognized by h. Then, $(L \cap \mathbb{R}_B) \cdot (\min \subseteq \{b\})$ is expressible in* LocTL_Σ.

Proof. The language $L \cap \mathbb{R}_B$ is recognized by $h\restriction_{\mathbb{M}_B}$ (Section 2) and we can apply the induction hypothesis to infer that it is expressible in LocTL_B: $L \cap \mathbb{R}_B = \mathcal{L}_B(\alpha) = \{t \in \mathbb{R}_B \mid t \models \alpha\}$ for some $\alpha \in \mathrm{LocTL}_B$. Let $\widetilde{\alpha} \in \mathrm{LocTL}_\Sigma$ be the formula given by Lemma 8 (see below). We have $(L \cap \mathbb{R}_B) \cdot (\min \subseteq \{b\}) = \mathcal{L}_\Sigma(\widetilde{\alpha})$. $\square$

Lemma 8. *Let $\alpha \in \mathrm{LocTL}_B$. There exists a formula $\widetilde{\alpha} \in \mathrm{LocTL}_\Sigma$ such that for all $t = t_1 t_2$ with $t_1 \in \mathbb{R}_B$ and $\min(t_2) \subseteq \{b\}$,*

$$t_1 \models \alpha \textit{ iff } t \models \widetilde{\alpha}.$$

Proof. We use structural induction on α. We have $\widetilde{\alpha \vee \beta} = \widetilde{\alpha} \vee \widetilde{\beta}$ and $\widetilde{\neg\alpha} = \neg\widetilde{\alpha}$. To deal with formulae of the form $\mathsf{EM}\,\varphi$, we need a similar result for internal formulae: for all $\varphi \in \mathrm{LocTL}^{\mathrm{i}}_B$, there exists a formula $\widetilde{\varphi} \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma$ such that for all $t = t_1 t_2$ with $t_1 \in \mathbb{R}_B$ and $\min(t_2) \subseteq \{b\}$, and for all $x \in t_1$,

$$t_1, x \models \varphi \text{ iff } t, x \models \widetilde{\varphi}.$$

This is also proved by structural induction on φ. Again, we have $\widetilde{\varphi \vee \psi} = \widetilde{\varphi} \vee \widetilde{\psi}$ and $\widetilde{\neg\varphi} = \neg\widetilde{\varphi}$. Now, $\widetilde{a} = a$, $\widetilde{\mathsf{EX}\,\varphi} = \mathsf{EX}_b\,\widetilde{\varphi}$, $\widetilde{\mathsf{EY}\,\varphi} = \mathsf{EY}\,\widetilde{\varphi}$, $\widetilde{\varphi\,\mathcal{U}_C\,\psi} = \widetilde{\varphi}\,\mathcal{U}_{C\cup\{b\}}\,\widetilde{\psi}$, and $\widetilde{\varphi\,\mathcal{S}_C\,\psi} = \widetilde{\varphi}\,\mathcal{S}_C\,\widetilde{\psi}$. Finally, we conclude the proof with $\widetilde{\mathsf{EM}\,\varphi} = \mathsf{EM}(\neg b \wedge \widetilde{\varphi})$. $\square$

5.1 First case: $L \cap \mathbb{M}_B(b\Gamma)^\infty$

Proposition 9. *Let $L \subseteq \mathbb{R}$ be a trace language recognized by h.*

1. *$\mathbb{M}_B \cdot (L \cap (b\Gamma)^\infty)$ is expressible in* LocTL_Σ.
2. *$L \cap \mathbb{M}_B(b\Gamma)^\infty$ is expressible in* LocTL_Σ.

In order to prove this proposition, we use a reduction to the word case. If T is a finite alphabet, we define the linear temporal logic $\mathrm{LTL}_T(\mathsf{XU})$ by the syntax:

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

The semantics is given for a word $t = t_1 t_2 \cdots \in T^\infty$ by

$$\begin{array}{lll} t \models u & \text{if} & t_1 = u \\ t \models \neg f & \text{if} & t \not\models f \\ t \models f \vee g & \text{if} & t \models f \text{ or } t \models g \\ t \models f\,\mathsf{XU}\,g & \text{if} & \exists j > 1, t_j t_{j+1} \cdots \models g \text{ and } t_k t_{k+1} \cdots \models f, \forall 1 < k < j. \end{array}$$

A formula $f \in \mathrm{LTL}_T(\mathsf{XU})$ defines the word language $\mathcal{L}(f) = \{t \in T^\infty \mid t \models f\}$. We use the following proposition which is a consequence of several results on the equivalence between aperiodic word languages, star-free word languages, first order word languages and word languages expressible in $\mathrm{LTL}_T(\mathsf{XU})$ [22, 12, 16, 25, 10, 19, 18, 2].

Proposition 10. *Every aperiodic word language $K \in T^\infty$ is expressible in* $\mathrm{LTL}_T(\mathsf{XU})$.

We fix $T = h(b\Gamma)$ and we define the mapping $\sigma : (b\Gamma)^\infty \to T^\infty$ by $\sigma(x) = h(bx_1)h(bx_2)\cdots$ if $x = bx_1bx_2\cdots$ with $x_i \in \Gamma$ for $i \geq 1$. Note that the mapping σ is well-defined since each trace $x \in (b\Gamma)^\infty$ has a unique factorization $x = bx_1bx_2\cdots$ with $x_i \in \Gamma$ for $i \geq 1$.

Lemma 11. *Let $L \subseteq \mathbb{R}$ be recognized by h. Then,*

1. *$L \cap (b\Gamma)^\infty = \sigma^{-1}(K)$ for some K expressible in* $\mathrm{LTL}_T(\mathsf{XU})$.
2. *$L \cap (b\Gamma)^* = \sigma^{-1}(K)$ for some K expressible in* $\mathrm{LTL}_T(\mathsf{XU})$.

Proof. 1. Let $K = \sigma(L \cap (b\Gamma)^\infty)$. We first show that $L \cap (b\Gamma)^\infty = \sigma^{-1}(K)$. The inclusion $\subseteq$ is clear. Conversely, let $x = bx_1bx_2\cdots \in L \cap (b\Gamma)^\infty$ and $y \in \sigma^{-1}(\sigma(x))$. Then, $y = by_1by_2\cdots \in (b\Gamma)^\infty$ with $h(bx_i) = h(by_i)$ for all i. Therefore, $y \sim_h x$ and $x \in L$ implies $y \in L$.

Next, we show that K is recognized by the evaluation morphism $e : T^* \to S$ defined by $e(u) = u$ for all $u \in T$.

Let $s,t \in T^\infty$ with $s \sim_e t$ and $s \in K$. Write $s = s_1 s_2 \cdots \in T^\infty$ and $t = t_1 t_2 \cdots \in T^\infty$. Since $T = h(b\Gamma)$ we find $x = bx_1 bx_2 \cdots \in L$ with $x_i \in \Gamma$ and $h(bx_i) = s_i$ for all $i > 1$; and $y = by_1 by_2 \cdots$ with $y_i \in \Gamma$ and $h(by_i) = t_i$ for all $i > 1$. From $s \sim_e t$ we deduce that $x \sim_h y$. And since $x \in L$, we get $y \in L$ and then $t = \sigma(y) \in K$.

Since S is an aperiodic monoid, $K \subseteq T^\infty$ is an aperiodic word language over T and by Proposition 10 we get K expressible in $\mathrm{LTL}_T(\mathsf{XU})$. Note that this is the only place where we use S aperiodic.

2. Similar. □

The next step in the proof is to show that if $K \subseteq T^\infty$ is expressible in $\mathrm{LTL}_T(\mathsf{XU})$ then $\sigma^{-1}(K)$ is expressible in LocTL_Σ.

Lemma 12. *Let $f \in \mathrm{LTL}_T(\mathsf{XU})$. There exists $\widetilde{f} \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma$ such that for all $t = t_1 t'$ with $t_1 \in \mathbb{M}$ and $t' \in (b\Gamma)^\infty \setminus \{1\}$, we have*

$$\sigma(t') \models f \textit{ iff } t, \min(t') \models \widetilde{f}.$$

Proof. We use an induction on f. Clearly, $\widetilde{\neg f} = \neg \widetilde{f}$ and $\widetilde{f \vee g} = \widetilde{f} \vee \widetilde{g}$. Now, we consider the case $f \,\mathsf{XU}\, g$ and we claim that

$$\widetilde{f \,\mathsf{XU}\, g} = \mathsf{EX}\big((\neg b \vee \widetilde{f})\,\mathcal{U}\,(b \wedge \widetilde{g})\big).$$

Let $t' = t_2 t_3 \cdots$ with $t_i \in b\Gamma$ for all $i > 1$ and let $x = \min(t')$.

Assume that $\sigma(t') \models f \,\mathsf{XU}\, g$ and let $j > 2$ be such that $\sigma(t_j t_{j+1} \cdots) \models g$ and $\sigma(t_k t_{k+1} \cdots) \models f$ for all $2 < k < j$. Let $x' \in t$ with $x \lessdot x'$ and $z = \min(t_j)$. We have $x' \leq z$, $\lambda(z) = b$ and $t, z \models \widetilde{g}$ by induction. Now, let $y \in t$ with $x' \leq y < z$. Either $\lambda(y) \neq b$ and $t, y \models \neg b$. Or there exists $2 < k < j$ with $y = \min(t_k)$ and by induction $t, y \models \widetilde{f}$. Therefore, $t, x' \models (\neg b \vee \widetilde{f})\,\mathcal{U}\,(b \wedge \widetilde{g})$ and $t, x \models \mathsf{EX}\big((\neg b \vee \widetilde{f})\,\mathcal{U}\,(b \wedge \widetilde{g})\big)$.

Conversely, assume that $t, x \models \mathsf{EX}((\neg b \vee \widetilde{f})\,\mathcal{U}\,(b \wedge \widetilde{g}))$. Let $x', z \in t$ with $x \lessdot x' \leq z$, $t, z \models b \wedge \widetilde{g}$ and $t, y \models \neg b \vee \widetilde{f}$ for all $x' \leq y < z$. Since $\lambda(z) = b$, there exists $j > 2$ with $z = \min(t_j)$ and by induction, we get $\sigma(t_j t_{j+1} \cdots) \models g$. Now, let $2 < k < j$ and $y = \min(t_k)$. We have $x' \leq y < z$ and since $\lambda(y) = b$ we get $t, y \models \widetilde{f}$ and $\sigma(t_k t_{k+1} \cdots) \models f$ by induction. Therefore, $\sigma(t') \models f \,\mathsf{XU}\, g$.

It remains to deal with a formula of the form $f = s \in T$. For all $r \in S$, the trace language $h_B^{-1}(r) \subseteq \mathbb{M}_B$ is aperiodic and, by induction on $|\Sigma|$, $h_B^{-1}(r) = \mathcal{L}_B(\alpha_r)$ for some formula $\alpha_r \in \mathrm{LocTL}_B$. Let $\widetilde{\alpha_r} \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma$ be the formula obtained using Lemma 13. We claim that

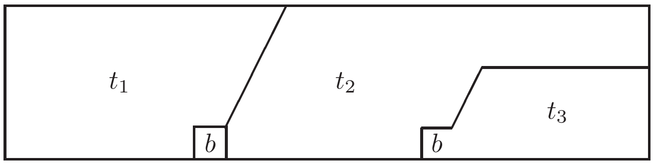
$$\widetilde{s} = \bigvee_{h(b)\cdot r = s} \widetilde{\alpha_r}.$$

We write $t' = bt_2 t_3$ with $t_2 \in \Gamma$ and $t_3 \in (b\Gamma)^\infty$. Let $r = h(t_2)$. Then, $t_2 \models \alpha_r$ and by Lemma 13 we obtain $t, \min(t') \models \widetilde{\alpha_r}$. Now, if $\sigma(t') \models s$ then $s = h(bt_2) = h(b)r$ and we get $t, \min(t') \models \widetilde{s}$.

Conversely, assume that $t, \min(t') \models \widetilde{\alpha_r}$ for some r with $s = h(b)r$. By Lemma 13 we get $t_2 \models \alpha_r$ and $h(t_2) = r$. Hence, $h(bt_2) = s$ and $\sigma(t') \models s$. □

Lemma 13. *For all $\alpha \in \mathrm{LocTL}_B$, there is a formula $\widetilde{\alpha} \in \mathrm{LocTL}^{\mathrm{i}}_{\Sigma}$ such that for all $t = t_1 b t_2 t_3 \in \mathbb{R}$ with $t_1 \in \mathbb{M}$, $t_2 \in \mathbb{R}_B$, $\min(t_2) \subseteq D(b)$ and $\min(t_3) \subseteq \{b\}$ we have*

$$t_2 \models \alpha \text{ iff } t, \min(bt_2t_3) \models \widetilde{\alpha}.$$



Proof. As for the proof of Lemma 8, we need a similar result for internal formulae: for all $\varphi \in \mathrm{LocTL}^{\mathrm{i}}_B$, there exists a formula $\widetilde{\varphi} \in \mathrm{LocTL}^{\mathrm{i}}_{\Sigma}$ such that for all $t = t_1 b t_2 t_3 \in \mathbb{R}$ with $t_1 \in \mathbb{M}$, $t_2 \in \mathbb{R}_B$, $\min(t_2) \subseteq D(b)$ and $\min(t_3) \subseteq \{b\}$ and for all $x \in t_2$, we have

$$t_2, x \models \varphi \text{ iff } t, x \models \widetilde{\varphi}.$$

We proceed by structural induction on α and φ. As always, we have $\widetilde{\neg\alpha} = \neg\widetilde{\alpha}$, $\widetilde{\alpha \vee \beta} = \widetilde{\alpha} \vee \widetilde{\beta}$, $\widetilde{\neg\varphi} = \neg\widetilde{\varphi}$, and $\widetilde{\varphi \vee \psi} = \widetilde{\varphi} \vee \widetilde{\psi}$. Now, $\widetilde{\mathsf{EM}\,\varphi} = \mathsf{EX}(\neg b \wedge \widetilde{\varphi})$, $\widetilde{a} = a$, $\widetilde{\mathsf{EX}\,\varphi} = \mathsf{EX}_b\,\widetilde{\varphi}$, $\widetilde{\mathsf{EY}\,\varphi} = \mathsf{EY}_b(\neg b \wedge \widetilde{\varphi})$,

$$\begin{aligned}
\widetilde{\varphi\,\mathcal{U}_C\,\psi} &= \bigvee_{E \subseteq \Sigma} \widetilde{\varphi}\,\mathcal{U}_{(C \cap E) \cup \{b\}}\,(\widetilde{\psi} \wedge E\,\mathcal{S}\,b) \\
\widetilde{\varphi\,\mathcal{S}_C\,\psi} &= \bigvee_{E \subseteq \Sigma} (E\,\mathcal{S}\,b) \wedge (\widetilde{\varphi}\,\mathcal{S}_{(C \cap E) \cup \{b\}}\,(\widetilde{\psi} \wedge \neg b)).
\end{aligned}$$

We prove the formula for $\mathcal{U}_C$. Let $x \in t_2$ with $t_2, x \models \varphi\,\mathcal{U}_C\,\psi$. Let $z \in t_2$ be such that $x \leq z$, $t_2, z \models \psi$, $t_2, y \models \varphi$ for all $x \leq y < z$, and for all $y \in t_2$, $y \leq z$ and $\lambda(y) \in C$ implies $y \leq x$. By induction, we get $t, z \models \widetilde{\psi}$ and $t, y \models \widetilde{\varphi}$ for all $x \leq y < z$.

Let $E = \mathrm{alph}(\downarrow_{t_2} z) = \mathrm{alph}(\{y \in t_2 \mid y \leq z\})$. Clearly, $t, z \models E\,\mathcal{S}\,b$. Let $y \in t$ with $y \leq z$. If $\lambda(y) = b$, then $y \leq \min(bt_2t_3) \leq x$. If $\lambda(y) \in C \cap E$, then there exists $y' \in t_2$ such that $\lambda(y') = \lambda(y) \in C$ and $y \leq y'$. From $y' \leq z$, we deduce $y' \leq x$ and then $y \leq y' \leq x$.

Conversely, let $x \in t_2$ and $E \subseteq \Sigma$ with $t, x \models \widetilde{\varphi}\,\mathcal{U}_{(C \cap E) \cup \{b\}}\,(\widetilde{\psi} \wedge E\,\mathcal{S}\,b)$. Let $z \in t$ with $x \leq z$, $t, z \models \widetilde{\psi} \wedge E\,\mathcal{S}\,b$, $t, y \models \widetilde{\varphi}$ for all $x \leq y < z$, and for all $y \leq z$, $\lambda(y) \in (C \cap E) \cup \{b\}$ implies $y \leq x$. If $z \in t_3$ then $y = \min(t_3)$ satisfies $\lambda(y) = b$, $y \leq z$ and $y \not\leq x$, a contradiction. Therefore, $z \in t_2$ and by induction, we get $t_2, z \models \psi$ and $t_2, y \models \varphi$ for all $x \leq y < z$. Now, let $y \in t_2$ with $y \leq z$ and $\lambda(y) \in C$. Since $t, z \models E\,\mathcal{S}\,b$ we deduce that $\mathrm{alph}(\downarrow_{t_2} z) \subseteq E$. Therefore, $\lambda(y) \in C \cap E$ and we obtain $y \leq x$.

The proof for $\mathcal{S}_C$ is similar. Let $z \in t_2$ with $t_2, z \models \varphi\,\mathcal{S}_C\,\psi$. Let $x \in t_2$ be such that $x \leq z$, $t_2, x \models \psi$, $t_2, y \models \varphi$ for all $x < y \leq z$, and for all $y \in t_2$, $y \leq z$ and $\lambda(y) \in C$ implies $y \leq x$. By induction, we get $t, x \models \widetilde{\psi}$ and $t, y \models \widetilde{\varphi}$ for all $x < y \leq z$.

Let $E = \mathrm{alph}(\downarrow_{t_2} z)$. Clearly, $t, z \models (E \mathrel{\mathcal{S}} b)$ and $t, x \models \neg b$. Let $y \in t$ with $y \leq z$. If $\lambda(y) = b$, then $y \leq \min(bt_2t_3) \leq x$. If $\lambda(y) \in C \cap E$, then there exists $y' \in t_2$ such that $\lambda(y') = \lambda(y) \in C$ and $y \leq y' \leq z$. We deduce that $y \leq y' \leq x$.

Conversely, let $z \in t_2$ with $t, z \models (E \mathrel{\mathcal{S}} b) \wedge (\widetilde{\varphi} \mathrel{\mathcal{S}_{(C \cap E) \cup \{b\}}} (\widetilde{\psi} \wedge \neg b))$ for some $E \subseteq \Sigma$. Let $x \in t$ with $x \leq z$, $t, x \models \widetilde{\psi} \wedge \neg b$, $t, y \models \widetilde{\varphi}$ for all $x < y \leq z$, and for all $y \leq z$, $\lambda(y) \in (C \cap E) \cup \{b\}$ implies $y \leq x$. If $x \in t_1$ then $y = \min(bt_2t_3)$ satisfies $\lambda(y) = b$, $y \leq z$ and $y \not\leq x$, a contradiction. Since $\lambda(x) \neq b$, we deduce that $x \in t_2$ and by induction, we get $t_2, x \models \psi$ and $t_2, y \models \varphi$ for all $x < y \leq z$. Now, let $y \in t_2$ with $y \leq z$ and $\lambda(y) \in C$. Since $t, z \models E \mathrel{\mathcal{S}} b$ we have $\mathrm{alph}(\downarrow_{t_2} z) \subseteq E$. Therefore, $\lambda(y) \in C \cap E$ and we obtain $y \leq x$. $\square$

We are now able to conclude the first case.

Proof of Proposition 9. 1. By Lemma 11, $L \cap (b\Gamma)^\infty = \sigma^{-1}(\mathcal{L}_T(f))$ for some $f \in \mathrm{LTL}_T(\mathsf{XU})$. Let $\widetilde{f} \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma$ be given by Lemma 12 and define the formula $\gamma = \mathsf{EM}(b \wedge \widetilde{f}) \vee \mathsf{EM}(\neg b \wedge \mathsf{F}_b \, \mathsf{EX}(b \wedge \widetilde{f}))$. We claim that

$$\mathbb{M}_B \cdot (L \cap (b\Gamma)^\infty \setminus \{1\}) = \mathbb{M}_B \cdot (b\Gamma)^\infty \cap \mathcal{L}(\gamma).$$

Indeed, let $t = t_1t'$ with $t_1 \in \mathbb{M}_B$, $t' \in L \cap (b\Gamma)^\infty \setminus \{1\}$ and let $x = \min(t')$. We have $\sigma(t') \models f$ and by Lemma 12 we get $t, x \models \widetilde{f}$. Now, if $x \in \min(t)$ we have $t \models \mathsf{EM}(b \wedge \widetilde{f})$ and otherwise we have $t \models \mathsf{EM}(\neg b \wedge \mathsf{F}_b \, \mathsf{EX}(b \wedge \widetilde{f}))$.

Conversely, let $t = t_1t'$ with $t_1 \in \mathbb{M}_B$, $t' \in (b\Gamma)^\infty$ and assume that $t \models \gamma$. Necessarily, $b \in \mathrm{alph}(t)$ and therefore $t' \neq 1$. From $t \models \gamma$ we deduce that $t, x \models b \wedge \widetilde{f}$ with $x = \min(t')$. By Lemma 12 we get $\sigma(t') \models f$. Therefore, $t' \in \sigma^{-1}(\mathcal{L}_T(f)) \subseteq L$ and t belongs to the left-hand side.

Since $\mathbb{M}_B \cdot (b\Gamma)^\infty$ and $\mathbb{M}_B$ are both expressible in LocTL_Σ, we deduce that $\mathbb{M}_B \cdot (L \cap (b\Gamma)^\infty)$ is expressible in LocTL_Σ.

2. Using Lemma 4, we know that $L \cap \mathbb{M}_B(b\Gamma)^\infty$ is a finite union of languages of the form $(L_1 \cap \mathbb{M}_B) \cdot (L_2 \cap (b\Gamma)^\infty)$ where the trace languages $L_i \subseteq \mathbb{R}$ are recognized by h. Now, since the product $\mathbb{M}_B(b\Gamma)^\infty$ is unambiguous, we have

$$\begin{aligned}(L_1 \cap \mathbb{M}_B) \cdot (L_2 \cap (b\Gamma)^\infty) &= (L_1 \cap \mathbb{M}_B) \cdot (b\Gamma)^\infty \cap \mathbb{M}_B \cdot (L_2 \cap (b\Gamma)^\infty) \\ (L_1 \cap \mathbb{M}_B) \cdot (b\Gamma)^\infty &= (\mathbb{M}_B \cdot (b\Gamma)^\infty) \cap ((L_1 \cap \mathbb{R}_B) \cdot (\min \subseteq \{b\})).\end{aligned}$$

We conclude using Lemma 3, Proposition 7 and Proposition 9(1). $\square$

It is important to note that until now, we are able to carry out the proof of Proposition 9 without using the modality EY and with alphabetic since's only. Recall that an *alphabetic since* is a formula of the form $A \mathrel{\mathcal{S}_C} a$, where $A, C \subseteq \Sigma$ and $a \in \Sigma$.

Theorem 14. *Let $L \subseteq \mathbb{M}$ be a trace languages recognized by h. Then, L is expressible in* LocTL_Σ *without using the modality* EY *and with alphabetic since's only.*

Proof. Note that if $L \subseteq \mathbb{M}$ then $L = L \cap \mathbb{M}_B(b\Gamma)^*$ therefore, we only need to consider the first case in the decomposition. In order to prove the theorem, we apply the proof contained in this section but in the proofs of Lemmas 8 and 13 we remove the cases for EY and we replace the cases for $\mathcal{S}_C$ respectively by $\widetilde{A\,\mathcal{S}_C\,a} = A\,\mathcal{S}_C\,a$ and

$$\widetilde{A\,\mathcal{S}_C\,a} = \bigvee_{E\subseteq\Sigma} (E\,\mathcal{S}\,b) \wedge (A\,\mathcal{S}_{(C\cap E)\cup\{b\}}\,a).$$

Note that in the last case we have $a \neq b$ since $A\,\mathcal{S}_C\,a \in \mathrm{LocTL}^{\mathrm{i}}_B$. $\square$

5.2 Second case: $L \cap \mathbb{R}^A_B \Omega_A$

Proposition 15. *Let $A \subseteq \Sigma$ be non-empty and let $L \subseteq \mathbb{R}$ be a trace language recognized by h. Then,*

1. $\mathbb{R}^A_B \cdot (L \cap \Omega_A)$ *is expressible in* LocTL_Σ.
2. $L \cap \mathbb{R}^A_B \Omega_A$ *is expressible in* LocTL_Σ.

Proof. 1. The language $L \cap \mathbb{R}_{I(A)}$ is recognized by $h\restriction_{\mathbb{M}_{I(A)}}$ (Section 2) and by induction we have $L \cap \mathbb{R}_{I(A)} = \mathcal{L}_{I(A)}(\alpha)$ for some $\alpha \in \mathrm{LocTL}_{I(A)}$. Let $\widetilde{\alpha} \in \mathrm{LocTL}_\Sigma$ be the formula given by Lemma 16 and define $\delta = \mathsf{EM}(b \wedge \widetilde{\alpha}) \vee \mathsf{EM}(\neg b \wedge \mathsf{F}_b\,\mathsf{EX}(b \wedge \widetilde{\alpha}))$. We claim that

$$\mathbb{R}^A_B \cdot (L \cap \Omega_A \setminus \{1\}) = \mathbb{R}^A_B \cdot \Omega_A \cap \mathcal{L}_\Sigma(\delta)$$

Indeed, let $t = t_1 t_2$ with $t_1 \in \mathbb{R}^A_B$, $t_2 \in L \cap \Omega_A \setminus \{1\}$ and let $x = \min(t_2)$. We have $t_2 \models \alpha$ and by Lemma 16 we get $t, x \models \widetilde{\alpha}$. We deduce that $t \models \delta$.

Conversely, let $t = t_1 t_2$ with $t_1 \in \mathbb{R}^A_B$, $t_{\in}\Omega_A$ and assume that $t \models \delta$. Necessarily, $b \in \mathrm{alph}(t)$ and therefore $t_2 \neq 1$. From $t \models \delta$ we deduce that $t, x \models b \wedge \widetilde{\alpha}$ with $x = \min(t_2)$. By Lemma 16 we get $t_2 \models \alpha$ and t belongs to the left-hand side.

Since $\mathbb{R}^A_B \cdot \Omega_A$ and $\mathbb{R}^A_B$ are expressible in LocTL_Σ, we deduce that $\mathbb{R}^A_B \cdot (L \cap \Omega_A)$ is expressible in LocTL_Σ.

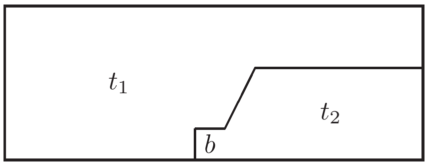
2. Using Lemma 4, we know that $L \cap \mathbb{R}^A_B \Omega_A$ is a finite union of languages of the form $(L_1 \cap \mathbb{R}^A_B) \cdot (L_2 \cap \Omega_A)$ where the trace languages $L_i \subseteq \mathbb{R}$ are recognized by h. Now, since the product $\mathbb{R}^A_B \Omega_A$ is unambiguous, we have

$$\begin{aligned}(L_1 \cap \mathbb{R}^A_B) \cdot (L_2 \cap \Omega_A) &= ((L_1 \cap \mathbb{R}^A_B) \cdot \Omega_A) \cap (\mathbb{R}^A_B \cdot (L_2 \cap \Omega_A)) \\ (L_1 \cap \mathbb{R}^A_B) \cdot \Omega_A &= (\mathbb{R}^A_B \cdot \Omega_A) \cap ((L_1 \cap \mathbb{R}_B) \cdot (\min \subseteq \{b\})).\end{aligned}$$

We conclude using Lemma 3, Proposition 7 and Proposition 15(1). $\square$

Lemma 16. *For all $\alpha \in \mathrm{LocTL}_\Sigma$, there is a formula $\widetilde{\alpha} \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma$ such that for all $t = t_1 t_2$ with $t_1 \in \mathbb{R}_B$ and $\min(t_2) = \{b\}$, we have*

$$t_2 \models \alpha \text{ iff } t, \min(t_2) \models \widetilde{\alpha}.$$



Proof. We prove simultaneously a similar result for internal formulae: for all $\varphi \in \mathrm{LocTL}^{\mathrm{i}}_{\Sigma}$, there exists a formula $\widetilde{\varphi} \in \mathrm{LocTL}^{\mathrm{i}}_{\Sigma}$ such that for all $t = t_1 t_2$ with $t_1 \in \mathbb{R}_B$, $\min(t_2) \subseteq \{b\}$ and for all $x \in t_2$, we have

$$t_2, x \models \varphi \text{ iff } t, x \models \widetilde{\varphi}.$$

We proceed by structural induction on α and φ. As always, we have $\widetilde{\neg\alpha} = \neg\widetilde{\alpha}$, $\widetilde{\alpha \vee \beta} = \widetilde{\alpha} \vee \widetilde{\beta}$, $\widetilde{\neg\varphi} = \neg\widetilde{\varphi}$, and $\widetilde{\varphi \vee \psi} = \widetilde{\varphi} \vee \widetilde{\psi}$. Now, $\widetilde{\mathsf{EM}\,\varphi} = \widetilde{\varphi}$, $\widetilde{a} = a$, and $\widetilde{\mathsf{EX}\,\varphi} = \mathsf{EX}\,\widetilde{\varphi}$. For the remaining cases, we consider the formula $\xi = b \wedge \neg\,\mathsf{EY}\,\mathsf{F}^{-1}\,b$ which characterizes the vertex $\min(t_2)$ within the trace t. We have,

$$\begin{aligned}
\widetilde{\mathsf{EY}\,\varphi} &= \mathsf{EY}(\widetilde{\varphi} \wedge \mathsf{F}^{-1}\,\xi),\\
\widetilde{\varphi\,\mathcal{U}_C\,\psi} &= \bigvee_{b \in E \subseteq \Sigma} \widetilde{\varphi}\,\mathcal{U}_{C\cap E}\,(\widetilde{\psi} \wedge E\,\mathcal{S}\,\xi),\\
\widetilde{\varphi\,\mathcal{S}_C\,\psi} &= \bigvee_{b \in E \subseteq \Sigma} (E\,\mathcal{S}\,\xi) \wedge (\widetilde{\varphi}\,\mathcal{S}_{C\cap E}\,(\widetilde{\psi} \wedge \mathsf{F}^{-1}\,\xi)).
\end{aligned}$$

We prove the formula for $\mathcal{U}_C$. Let $x \in t_2$ with $t_2, x \models \varphi\,\mathcal{U}_C\,\psi$. Let $z \in t_2$ be such that $x \le z$, $t_2, z \models \psi$, $t_2, y \models \varphi$ for all $x \le y < z$, and for all $y \in t_2$, $y \le z$ and $\lambda(y) \in C$ implies $y \le x$. By induction, we get $t, z \models \widetilde{\psi}$ and $t, y \models \widetilde{\varphi}$ for all $x \le y < z$.

Let $E = \mathrm{alph}(\downarrow_{t_2} z) = \mathrm{alph}(\{y \in t_2 \mid y \le z\})$. Clearly, $b \in E$ and $t, z \models E\,\mathcal{S}\,\xi$. Let $y \in t$ with $y \le z$ and $\lambda(y) \in C \cap E$. There exists $y' \in t_2$ such that $\lambda(y') = \lambda(y) \in C$ and $y \le y'$. From $y' \le z$, we deduce $y' \le x$ and then $y \le y' \le x$.

Conversely, let $x \in t_2$ be such that $t, x \models \widetilde{\varphi}\,\mathcal{U}_{C\cap E}\,(\widetilde{\psi} \wedge E\,\mathcal{S}\,\xi)$ for some $E \subseteq \Sigma$ with $b \in E$. Let $z \in t$ with $x \le z$, $t, z \models \widetilde{\psi} \wedge E\,\mathcal{S}\,\xi$, $t, y \models \widetilde{\varphi}$ for all $x \le y < z$, and for all $y \le z$, $\lambda(y) \in C \cap E$ implies $y \le x$. By induction, we get $t_2, z \models \psi$ and $t_2, y \models \varphi$ for all $x \le y < z$. Now, let $y \in t_2$ with $y \le z$ and $\lambda(y) \in C$. Since $t, z \models E\,\mathcal{S}\,\xi$ and $b \in E$, we deduce that $\mathrm{alph}(\downarrow_{t_2} z) \subseteq E$. Therefore, $\lambda(y) \in C \cap E$ and we obtain $y \le x$.

The proof for $\mathcal{S}_C$ is similar. Let $z \in t_2$ with $t_2, z \models \varphi\,\mathcal{S}_C\,\psi$. Let $x \in t_2$ be such that $x \le z$, $t_2, x \models \psi$, $t_2, y \models \varphi$ for all $x < y \le z$, and for all $y \in t_2$, $y \le z$ and $\lambda(y) \in C$ implies $y \le x$. By induction, we get $t, x \models \widetilde{\psi}$ and $t, y \models \widetilde{\varphi}$ for all $x < y \le z$.

Let $E = \mathrm{alph}(\downarrow_{t_2} z)$. Clearly, $b \in E$ and $t, z \models E\,\mathcal{S}\,\xi$. Let $y \in t$ with $y \le z$ and $\lambda(y) \in C \cap E$. There exists $y' \in t_2$ such that $\lambda(y') = \lambda(y) \in C$ and $y \le y'$. From $y' \le z$, we deduce $y' \le x$ and then $y \le y' \le x$. Now, $\min(t_2) \le x$ and $t, \min(t_2) \models \xi$ implies that $t, x \models \mathsf{F}^{-1}\,\xi$.

Conversely, let $z \in t_2$ be such that $t, z \models (E\,\mathcal{S}\,\xi) \wedge (\widetilde{\varphi}\,\mathcal{S}_{C\cap E}\,(\widetilde{\psi} \wedge \mathsf{F}^{-1}\,\xi))$ for some $E \subseteq \Sigma$ with $b \in E$. Let $x \in t$ with $x \le z$, $t, x \models \widetilde{\psi} \wedge \mathsf{F}^{-1}\,\xi$, $t, y \models \widetilde{\varphi}$ for all $x < y \le z$, and for all $y \le z$, $\lambda(y) \in C \cap E$ implies $y \le x$. From $t, x \models \mathsf{F}^{-1}\,\xi$ we deduce that $x \in t_2$. Hence, by induction, we get $t_2, x \models \psi$ and $t_2, y \models \varphi$ for all $x < y \le z$. Now, let $y \in t_2$ with $y \le z$ and $\lambda(y) \in C$. Since $t, z \models E\,\mathcal{S}\,\xi$ and $b \in E$, we deduce that $\mathrm{alph}(\downarrow_{t_2} z) \subseteq E$. Therefore, $\lambda(y) \in C \cap E$ and we obtain $y \le x$. □

5.3 Third case: $L \cap \mathbb{M} b \Gamma^A \Omega_A$

Our goal is to prove

Proposition 17. *Let $A \subseteq \Sigma$ be non-empty and let $L \subseteq \mathbb{R}$ be a trace language recognized by h. Then,*

1. *$\mathbb{M}_B(L \cap (b\Gamma)^*) b\Gamma^A \Omega_A$ is expressible in LocTL_Σ.*
2. *$\mathbb{M} b(L \cap \Gamma^A)\Omega_A$ is expressible in LocTL_Σ.*
3. *$\mathbb{M} b\Gamma^A(L \cap \Omega_A)$ is expressible in LocTL_Σ.*
4. *$L \cap \mathbb{M} b\Gamma^A \Omega_A$ is expressible in LocTL_Σ.*

We start with two lemmas.

Lemma 18. *Let $A \subseteq \Sigma$ be non-empty and let $f \in \mathrm{LTL}_T(\mathsf{XU})$. There exists $\widetilde{f} \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma$ such that for all $t = t_1 t_2 t_3$ with $t_1 \in \mathbb{M}$, $t_2 \in (b\Gamma)^*$, $t_3 \in b\Gamma^A \Omega_A$, we have*

$$\sigma(t_2) \models f \text{ iff } t, \min(t_2 t_3) \models \widetilde{f}.$$

Proof. Similar to that of Lemma 12. For the case $f \,\mathsf{XU}\, g$, we need to ensure that we stay within t_2. For this, we chose some letter $a \in A$ and we consider the formula $b \wedge \mathsf{F}^\infty_b\, a$ which characterizes the vertex $\min(t_3)$ within the trace t. We claim that

$$\widetilde{f \,\mathsf{XU}\, g} = \mathsf{EX}((\neg b \vee \widetilde{f})\,\mathcal{U}\,(b \wedge \widetilde{g} \wedge \mathsf{F}(b \wedge \mathsf{F}^\infty_b\, a))).$$

Now, we consider the case $f = s \in T$. For all $r \in S$, the trace language $h_B^{-1}(r) \subseteq \mathbb{M}_B$ is aperiodic and, by induction on $|\Sigma|$, $h_B^{-1}(r) = \mathcal{L}_B(\alpha_r)$ for some formula $\alpha_r \in \mathrm{LocTL}_B$. Let $\widetilde{\alpha_r} \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma$ be the formula obtained using Lemma 13. We prove as in Lemma 12 that

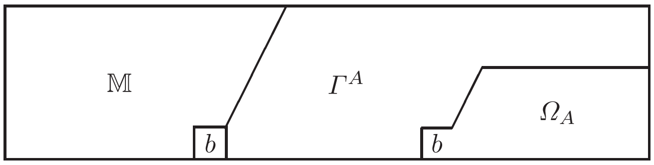
$$\widetilde{s} = \mathsf{EX}\,\mathsf{F}(b \wedge \mathsf{F}^\infty_b\, a) \wedge \bigvee_{h(b)\cdot r = s} \widetilde{\alpha_r}.$$

If $\sigma(t_2) \models s$ then $t_2 \in (b\Gamma)^+$ and $t, \min(t_2 t_3) \models \mathsf{EX}\,\mathsf{F}(b \wedge \mathsf{F}^\infty_b\, a)$. Now, write $t_2 = bt't''$ with $t' \in \Gamma$, $t'' \in (b\Gamma)^*$ and let $r = h(t')$. We have $h(b)r = s$. Since $t' \in h_B^{-1}(r)$, we have $t' \models \alpha_r$ and by Lemma 13 we get $t, \min(t_2 t_3) \models \widetilde{\alpha_r}$. Therefore, $t, \min(t_2 t_3) \models \widetilde{s}$.

Conversely, if $t, \min(t_2 t_3) \models \mathsf{EX}\,\mathsf{F}(b \wedge \mathsf{F}^\infty_b\, a)$ then $t_2 \in (b\Gamma)^+$ and we can write $t_2 = bt't''$ with $t' \in \Gamma$ and $t'' \in (b\Gamma)^*$. Assume that $t, \min(bt't''t_3) \models \widetilde{\alpha_r}$ for some $r \in S$ with $s = h(b)r$. From Lemma 13 we get $t' \models \alpha_r$ and $h(t') = r$. We deduce $h(bt') = s$ and $\sigma(t_2) \models s$. □

Lemma 19. *Let $A \neq \emptyset$ and $\alpha \in \mathrm{LocTL}_\Sigma$. There exists $\widetilde{\alpha} \in \mathrm{LocTL}^{\mathrm{i}}_\Sigma$ such that for all $t = t_1 t_2$ with $t_1 \in \mathbb{M}b\Gamma^A$ and $t_2 \in \Omega_A \setminus \{1\}$,*

$$t_2 \models \alpha \textit{ iff } t, \min(t_2) \models \widetilde{\alpha}.$$



Proof. Similar to that of Lemma 16 except for the formula ξ which characterizes the vertex $\min(t_2)$ within the trace t. Here we chose $a \in A$ and we let $\xi = b \wedge \mathsf{EY}\,\mathsf{F}^{-1}{}_b(b \wedge \mathsf{F}^\infty_b\, a)$. $\square$

Proof of Proposition 17. 1. By Lemma 11, $L \cap (b\Gamma)^* = \sigma^{-1}(\mathcal{L}_T(f))$ for some $f \in \mathrm{LTL}_T(\mathsf{XU})$. Let $\widetilde{f}$ be given by Lemma 18 and $\gamma = \mathsf{EM}(b \wedge \widetilde{f}) \vee \mathsf{EM}(\neg b \wedge \mathsf{F}_b\, \mathsf{EX}(b \wedge \widetilde{f}))$. We claim that

$$\mathbb{M}_B(L \cap (b\Gamma)^*)b\Gamma^A\Omega_A = \mathbb{M}b\Gamma^A\Omega_A \cap \mathcal{L}_\Sigma(\gamma).$$

Indeed, let $t = t_1t_2t_3$ with $t_1 \in \mathbb{M}_B$, $t_2 \in (b\Gamma)^*$ and $t_3 \in b\Gamma^A\Omega_A$. We have

$$t_2 \in L \text{ iff } \sigma(t_2) \models f \text{ iff } t, \min(t_2t_3) \models \widetilde{f} \text{ iff } t \models \gamma.$$

By Lemma 3, the language $\mathbb{M}b\Gamma^A\Omega_A$ is expressible in LocTL_Σ. We deduce that $\mathbb{M}_B(L \cap (b\Gamma)^*)b\Gamma^A\Omega_A$ is expressible in LocTL_Σ.

2. The language $L \cap \mathbb{R}_B$ is recognized by $h\restriction_{\mathbb{M}_B}$ (Section 2) and by induction we have $L \cap \mathbb{R}_B = \mathcal{L}_B(\alpha)$ for some formula $\alpha \in \mathrm{LocTL}_B$. Let $\widetilde{\alpha} \in \mathrm{LocTL}_\Sigma$ be the formula given by Lemma 13 and define the formula $\delta = \mathsf{EM}\,\mathsf{F}(b \wedge \mathsf{F}^\infty_b\, a \wedge \widetilde{\alpha})$. We claim that

$$\mathbb{M}b(L \cap \Gamma^A)\Omega_A = \mathbb{M}b\Gamma^A\Omega_A \cap \mathcal{L}_\Sigma(\delta).$$

Indeed, let $t = t_1bt_2t_3$ with $t_1 \in \mathbb{M}$, $t_2 \in \Gamma^A$ and $t_3 \in \Omega_A$. We have $t, z \models b \wedge \mathsf{F}^\infty_b\, a$ iff $z = \min(bt_2t_3)$. Therefore,

$$t_2 \in L \text{ iff } t_2 \models \alpha \text{ iff } t, \min(bt_2t_3) \models \widetilde{\alpha} \text{ iff } t \models \delta.$$

We conclude as above.

3. The language $L \cap \mathbb{R}_{I(A)}$ is recognized by $h\restriction_{\mathbb{M}_{I(A)}}$ (Section 2) and by induction we have $L \cap \mathbb{R}_{I(A)} = \mathcal{L}_{I(A)}(\alpha)$ for some $\alpha \in \mathrm{LocTL}_{I(A)}$. Let $\widetilde{\alpha} \in \mathrm{LocTL}_\Sigma$ be the formula given by Lemma 19 and define the formula $\delta = \mathsf{EM}\,\mathsf{F}(b \wedge \mathsf{F}^\infty_b\, a \wedge \mathsf{F}_b\, \mathsf{EX}(b \wedge \widetilde{\alpha}))$. We claim that

$$\mathbb{M}b\Gamma^A(L \cap \Omega_A \setminus \{1\}) = \mathbb{M}b\Gamma^A\Omega_A \cap \mathcal{L}_\Sigma(\delta).$$

Indeed, let $t = t_1bt_2t_3$ with $t_1 \in \mathbb{M}$, $t_2 \in \Gamma^A$ and $t_3 \in \Omega_A$. We have $t, z \models b \wedge \mathsf{F}^\infty_b\, a$ iff $z = \min(bt_2t_3)$. Note that $t \models \delta$ implies $t_3 \neq 1$. Now, if $t_3 \neq 1$, we have

$$t_3 \in L \text{ iff } t_3 \models \alpha \text{ iff } t, \min(t_3) \models \widetilde{\alpha} \text{ iff } t \models \delta.$$

Note that $\mathbb{M}b\Gamma^A = \mathcal{L}(\mathsf{EM}\,\mathsf{F}(b \wedge \mathsf{F}_b^\infty\, a \wedge \neg\, \mathsf{EX}\,\mathsf{F}\, b)$. Hence, we can conclude as above.

4. Using Lemma 4, we know that $L \cap \mathbb{M}b\Gamma^A\Omega_A$ is a finite union of languages of the form $(L_1 \cap \mathbb{M}_B)(L_2 \cap (b\Gamma)^*)b(L_3 \cap \Gamma^A)(L_4 \cap \Omega_A)$ where the trace languages $L_i \subseteq \mathbb{R}$ are recognized by h. Now, since the product $\mathbb{M}b\Gamma^A\Omega_A$ is unambiguous, we have

$$\begin{aligned}(L_1 \cap \mathbb{M}_B)(L_2 \cap (b\Gamma)^*)b(L_3 \cap \Gamma^A)(L_4 \cap \Omega_A) = \\ &(L_1 \cap \mathbb{R}_B) \cdot (\min \subseteq \{b\}) \\ &\cap \mathbb{M}_B(L_2 \cap (b\Gamma)^*)b\Gamma^A\Omega_A \\ &\cap \mathbb{M}b(L_3 \cap \Gamma^A)\Omega_A \\ &\cap \mathbb{M}b\Gamma^A(L_4 \cap \Omega_A).\end{aligned}$$

We conclude using Proposition 7 and the three points above. □

6 Satisfiability

To decide satisfiability, we work with alternating automata over sequentializations of traces. The states of the alternating automaton consist of subformulas of the main formula together with bookkeeping information about the letters traversed while satisfying until and since requirements. The basic technique we use was introduced in [1] and refined in [5]. The additional complication for our logic is that we have to deal with past requirements, both explicitly in the operators EY and $\mathcal{S}_C$, and implicitly, because of the alphabetic filters in $\mathcal{U}_C$ and $\mathcal{S}_C$. We thus need two-way alternating automata. Nevertheless, checking emptiness for this class remains within PSPACE [11].

We assume the reader is familiar with two-way alternating automata (see e.g. [13, 14, 28]) and the usual translation from LTL formulae to alternating automata (see e.g. [27]).

Let $\alpha \in \mathrm{LocTL}_\Sigma$. 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 $\varphi\,\mathcal{U}_C\,\psi$ or $\mathsf{EY}\,\varphi$ or $\varphi\,\mathcal{S}_C\,\psi$. For $E \subseteq \Sigma$, we denote by $\widetilde{E}$ the set $\{a \in \Sigma \mid D(a) \subseteq D(E)\}$. We construct an alternating automaton $\mathcal{A}_\alpha$ as follows. The set of states is $Q \cup \overline{Q}$ where $\overline{Q} = \{\neg p \mid p \in Q\}$ and $Q = Q_1 \cup Q_2 \cup Q_3 \cup Q_4 \cup Q_5 \cup Q_6 \cup Q_7$ (see Table 1). The intuition behind these states is given below together with the definition of the transition function.

The set of positive boolean combinations of elements in a set Z is denoted $\mathbb{B}^+(Z)$. The initial state is $\alpha \in \mathbb{B}^+(Q_1 \cup \overline{Q}_1)$. The automaton needs to sense when its head moves backward past the first letter of the word. We use the symbol $\sharp \notin \Sigma$ to signal the beginning of the word. Hence, the automaton reads the letter $\sharp$ when its head has moved backward past the first letter of the word. We will see that in such a case the state must be in $Q_4 \cup Q_6 \cup Q_7 \cup \overline{Q}_4 \cup \overline{Q}_6 \cup \overline{Q}_7$. Hence, we need not define the transition function on $\sharp$ for the other states.

The extended transition function is a mapping $\delta : \mathbb{B}^+(Q \cup \overline{Q}) \times (\Sigma \cup \{\sharp\}) \to \mathbb{B}^+(\{\mathsf{X}, \mathsf{Y}\} \times (Q \cup \overline{Q}))$. The actual transition function of $\mathcal{A}_\alpha$ is the restriction of

$$
\begin{aligned}
Q_1 = \{\xi \mid \xi & \text{ is a subformula of } \alpha \text{ of the form } \mathsf{EM}\,\varphi \text{ or } b \text{ or} \\
& \mathsf{EX}\,\varphi \text{ or } \varphi\,\mathcal{U}_C\,\psi \text{ or } \mathsf{EY}\,\varphi \text{ or } \varphi\,\mathcal{S}_C\,\psi\} \\
Q_2 = \{(D(B), \mathsf{EM}\,\varphi) & \mid \mathsf{EM}\,\varphi \text{ is a subformula of } \alpha,\ B \subseteq \Sigma \text{ and } D(B) \neq \Sigma\} \\
Q_3 = \{(D(b), D(B), \mathsf{EX}\,\varphi) & \mid \mathsf{EX}\,\varphi \text{ is a subformula of } \alpha,\ b \in \Sigma,\ B \subseteq \Sigma, \\
& \{b\} \cup B \text{ connected and } D(b) \setminus D(B) \neq \emptyset\} \\
Q_4 = \{(D(b), D(B), \mathsf{EY}\,\varphi) & \mid \mathsf{EY}\,\varphi \text{ is a subformula of } \alpha,\ b \in \Sigma,\ B \subseteq \Sigma, \\
& \{b\} \cup B \text{ connected and } D(b) \setminus D(B) \neq \emptyset\} \\
Q_5 = \{(D(A), D(B), D(E), \varphi\,\mathcal{U}_C\,\psi) & \mid \varphi\,\mathcal{U}_C\,\psi \text{ is a subformula of } \alpha, \\
& A, B, E \subseteq \Sigma,\ A, E \text{ connected},\ D(A) \subseteq D(E) \text{ and} \\
& (D(A) \cap \widetilde{E}) \setminus (D(B) \cup C) \neq \emptyset\} \\
Q_6 = \{(D(A), D(E), C) & \mid \mathcal{U}_C \text{ or } \mathcal{S}_C \text{ is used in } \alpha,\ A \subseteq E \subseteq \Sigma, \\
& A, E \text{ connected and } D(E) \setminus D(A) \neq \emptyset\} \\
Q_7 = \{(D(A), D(B), \varphi\,\mathcal{S}_C\,\psi) & \mid \varphi\,\mathcal{S}_C\,\psi \text{ is a subformula of } \alpha, \\
& B \subseteq A \subseteq \Sigma,\ A \text{ connected and } D(A) \setminus D(B) \neq \emptyset\}.
\end{aligned}
$$

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

δ to $(Q \cup \overline{Q}) \times (\Sigma \cup \{\sharp\})$. To indicate that the head of the two-way automaton has to move to the right, we mark a state with X. Similarly, Y means that the head has to move to the left. If $p \in Q \cup \overline{Q}$, we simply write $\mathsf{X}\,p$ and $\mathsf{Y}\,p$ intead of (X, p) and (Y, p). For instance, if $\delta(p, a) = \mathsf{X}\,q \wedge \mathsf{Y}\,r$ and we are in state p reading letter a then the automaton forks with one copy moving its head to the right and entering state q while the other copy moves its head to the left and goes to state r. As usual, we define δ inductively. For $f, g \in \mathbb{B}^+(Q)$, $p \in Q$ and $a \in \Sigma \cup \{\sharp\}$ we define

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

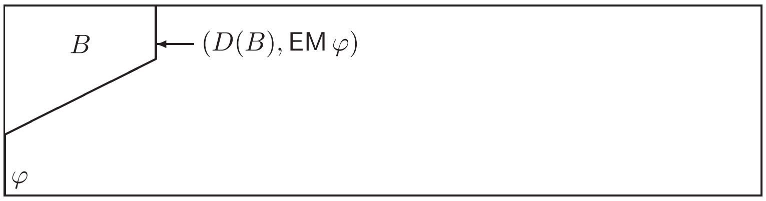
where $\widetilde{f \vee g} = \widetilde{f} \wedge \widetilde{g}$ and $\widetilde{f \wedge g} = \widetilde{f} \vee \widetilde{g}$, for $f, g \in \mathbb{B}^+(\{\mathsf{X}, \mathsf{Y}\} \times (Q \cup \overline{Q}))$; $\widetilde{\mathsf{X}\,f} = \mathsf{X}\,\widetilde{f}$ and $\widetilde{\mathsf{Y}\,f} = \mathsf{Y}\,\widetilde{f}$, for $f, g \in Q \cup \overline{Q}$; 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 $a \in \Sigma$:

$$
\begin{aligned}
\delta(\mathsf{EM}\,\varphi, a) &= \begin{cases} \quad \delta(\varphi, a) & \\ \vee\ \mathsf{X}(D(a), \mathsf{EM}\,\varphi) & \text{if } D(a) \neq \Sigma. \end{cases} \\
\delta((D(B), \mathsf{EM}\,\varphi), a) &= \begin{cases} \quad \delta(\varphi, a) & \text{if } a \notin D(B) \\ \vee\ \mathsf{X}(D(B) \cup D(a), \mathsf{EM}\,\varphi) & \text{if } D(B) \cup D(a) \neq \Sigma. \end{cases}
\end{aligned}
$$

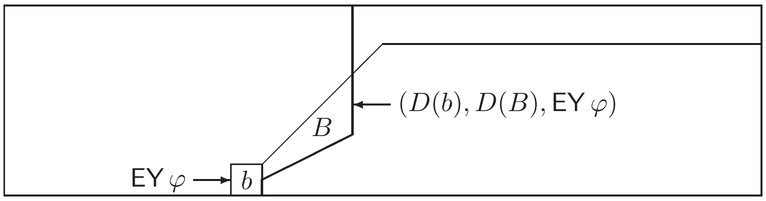


A transition from state $\mathsf{EM}\,\varphi$ only occurs on the first position of the word and looks for a minimal vertex in the associated trace satisfying φ. Since this minimal vertex needs not be the first letter of the word, we may have to skip some letters before starting the verification of φ. Once some letters have been skipped, all dependent letters have to be skipped as well. In a state $(D(B), \mathsf{EM}\,\varphi)$ the first component remembers the letters that have to be skipped. If $D(B) = \Sigma$ then the verification of φ cannot be started. This explains the restriction $D(B) \neq \Sigma$ in the definition of Q_2 and the corresponding conditions in the definition of δ.

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

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

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



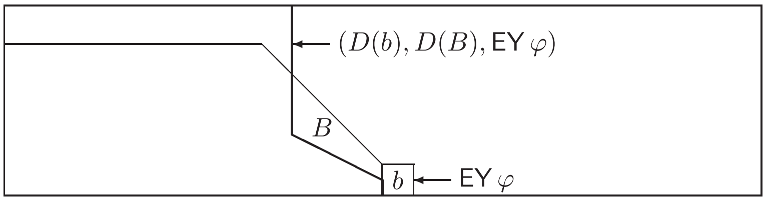
A transition from state $\mathsf{EX}\,\varphi$ looks in the associated trace for an immediate successor of the current vertex satisfying φ. This immediate successor has to be labeled with a letter dependent on the label (say b) of the current vertex. In a state $(D(b), D(B), \mathsf{EX}\,\varphi)$, the first component allows 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.

Note that, by the definition of δ we always have $D(b) \cup D(B)$ connected if the state $(D(b), D(B), \mathsf{EX}\,\varphi)$ is accessible. Moreover, if $D(b) \setminus D(B) = \emptyset$ then the verification of φ cannot be started. This explains the restrictions in the definition of Q_3 and the corresponding conditions in the definition of δ.

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

$$\begin{aligned}
\delta(\mathsf{EY}\,\varphi, a) &= \mathsf{Y}(D(a), \emptyset, \mathsf{EY}\,\varphi) \\
\delta((D(b), D(B), \mathsf{EY}\,\varphi), a) &=
\end{aligned}$$

$$\begin{cases}
\quad \delta(\varphi, a) & \text{if } a \in D(b) \setminus D(B), \\
\vee\ \mathsf{Y}(D(b), D(B), \mathsf{EY}\,\varphi) & \text{if } a \notin D(b) \cup D(B), \\
\vee\ \mathsf{Y}(D(b), D(B) \cup D(a), \mathsf{EY}\,\varphi) & \text{if } a \in D(b) \cup D(B) \text{ and} \\
& \quad D(b) \setminus (D(B) \cup D(a)) \neq \emptyset.
\end{cases}$$



The explanations in this case are similar to those for $\mathsf{EX}\,\varphi$. Note that we may move backward past the first position of the word. In which case the oblication to verify φ has not been fulfilled and the branch in the run-tree should be rejected. Hence we put

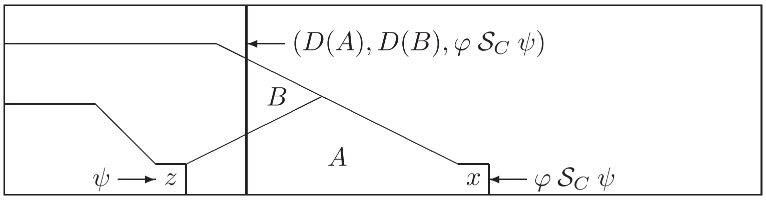
$$\delta((D(b), D(B), \mathsf{EY}\,\varphi), \sharp) = \bot.$$

Transitions for $\varphi\,\mathcal{S}_C\,\psi$ and $a \in \Sigma$:

$$\delta(\varphi\,\mathcal{S}_C\,\psi, a) = \begin{cases}
\quad \delta(\psi, a) & \\
\vee\ \delta(\varphi, a) \wedge \mathsf{Y}(D(a), \emptyset, \varphi\,\mathcal{S}_C\,\psi) & \text{if } a \notin C.
\end{cases}$$

$$\delta((D(A), D(B), \varphi\,\mathcal{S}_C\,\psi), a) =$$

$$\begin{cases}
\quad \mathsf{Y}(D(A), D(B), \varphi\,\mathcal{S}_C\,\psi) & \text{if } a \notin D(A) \\
\vee\ \delta(\psi, a) \wedge \mathsf{Y}(D(a), D(A) \cup D(a), C) & \text{if } a \in D(A) \setminus D(B), \\
\vee\ \delta(\varphi, a) \wedge \mathsf{Y}(D(A) \cup D(a), D(B), \varphi\,\mathcal{S}_C\,\psi) & \text{if } a \in D(A) \setminus (D(B) \cup C), \\
\vee\ \mathsf{Y}(D(A) \cup D(a), D(B) \cup D(a), \varphi\,\mathcal{S}_C\,\psi) & \text{if } a \in D(A) \setminus C \text{ and} \\
& \quad D(A) \setminus (D(B) \cup D(a)) \neq \emptyset.
\end{cases}$$



In order to check that $\varphi\mathcal{S}_C\psi$ holds at the current vertex (say x), we have to move backward in the word until we find some vertex (say z) causally in the past of x that satisfies ψ. The first component of $(D(A), D(B), \varphi\,\mathcal{S}_C\,\psi)$ gives the letters that must be causally in the past of x. Hence, if we read a letter that is not in $D(A)$, we just skip it and continue to move backward. Otherwise, we check that the letter is not in C and we guess whether it will be causally in the future 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 past. This is the purpose of the second component in a state $(D(A), D(B), \varphi\, \mathcal{S}_C\, \psi)$. 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 past of x and that we have checked φ for all vertices causally between z and x. At this point, the filter condition is not completely checked. This is why we move backward to the state $(D(a), D(A) \cup D(a), C)$.

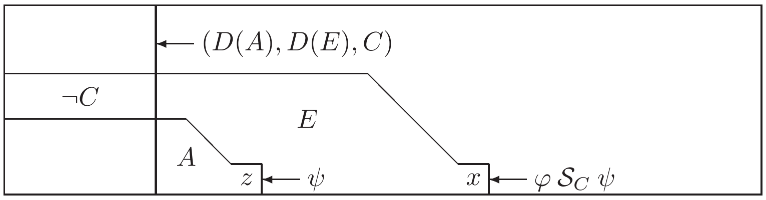
Note that, by the definition of δ, all accessible states in Q_7 are of the form $(D(A), D(B), \varphi\, \mathcal{S}_C\, \psi)$ for some $B \subseteq A$ and A connected. Moreover, if $D(A) \setminus D(B) = \emptyset$ then the verification of ψ cannot be started. This explains the restrictions in the definition of Q_7 and the corresponding conditions in the definition of δ.

Also, we may move backward past the first position of the word. In which case the oblication to verify ψ has not been fulfilled and the branch in the run-tree should be rejected. Hence we put

$$\delta((D(A), D(B), \varphi\, \mathcal{S}_C\, \psi), \sharp) = \bot.$$

Transitions for $(D(A), D(E), C)$ and $a \in \Sigma$:

$$\delta((D(A), D(E), C), a) \; = \begin{cases} \quad \mathsf{Y}(D(A), D(E), C) & \text{if } a \notin D(E) \\ \vee\; \mathsf{Y}(D(A), D(E) \cup D(a), C) & \text{if } a \in D(E) \setminus (D(A) \cup C) \\ \vee\; \mathsf{Y}(D(A) \cup D(a), D(E) \cup D(a), C) & \text{if } a \in D(A) \text{ and} \\ & \quad D(E) \setminus (D(A) \cup D(a)) \neq \emptyset, \\ \vee\; \top & \text{if } a \in D(A) \text{ and} \\ & \quad D(E) \subseteq D(A) \cup D(a). \end{cases}$$



This case is rather simple. In a state $(D(A), D(E), C)$, the first component gives the letters that must be causally in the past of z while the second component gives the letters that must be causally in the past of x. If a letter is not in $D(E)$ then we just skip it. If a letter is in $D(E) \setminus D(A)$ then it should not be in C.

Note that, by the definition of δ, all accessible states in Q_6 are of the form $(D(A), D(E), C)$ for some subalphabets A, E with $A \subseteq E$ and A, E connected. Moreover, if $D(A) = D(E)$ then the filter condition has been completely verified. This explains the restrictions in the definition of Q_6 and the corresponding conditions in the definition of δ.

Also, we may move backward past the first position of the word. In which case the filter condition has been completely verified and we can accept the branch in the run-tree. Hence we put

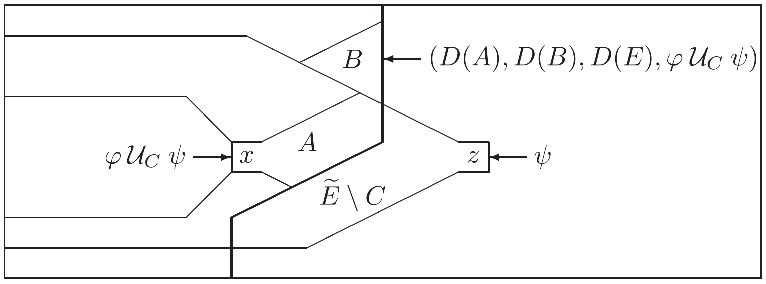
$$\delta((D(A), D(E), C), \sharp) = \top.$$

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

$$\delta(\varphi\mathcal{U}_C\psi, a) = \delta(\psi, a) \vee \delta(\varphi, a) \wedge \bigvee_E (\mathsf{X}(D(a), \emptyset, D(E), \varphi\mathcal{U}_C\psi) \wedge \mathsf{Y}(D(a), D(E), C))$$

where the disjunction ranges over all E connected such that $a \in E$.

$$\delta((D(A), D(B), D(E), \varphi\,\mathcal{U}_C\,\psi), a) \ = \begin{cases} \quad \delta(\psi, a) & \text{if } a \in (D(A) \cap \widetilde{E}) \setminus (D(B) \cup C) \\ \vee\ \delta(\varphi, a) \wedge \mathsf{X}(D(A) \cup D(a), D(B), D(E), \varphi\,\mathcal{U}_C\,\psi) & \text{if } a \in (D(A) \cap \widetilde{E}) \setminus (D(B) \cup C), \\ \vee\ \mathsf{X}(D(A), D(B), D(E), \varphi\,\mathcal{U}_C\,\psi) & \text{if } a \in \widetilde{E} \setminus (D(A) \cup D(B) \cup C), \\ \vee\ \mathsf{X}(D(A), D(B) \cup D(a), D(E), \varphi\,\mathcal{U}_C\,\psi) & \text{if } (D(A) \cap \widetilde{E}) \setminus (D(B) \cup D(a) \cup C) \neq \emptyset. \end{cases}$$



This is the most difficult case. In order to check $\varphi\,\mathcal{U}\,\psi$ at some vertex x, we first guess the set E of letters that will occur in the future of the word but causally in the past of the vertex z where ψ eventually holds. We move backwards and check the filter condition starting with $(D(a), D(E), C)$. When moving forward, we use also alphabetic informations. Assume that we are in state $(D(A), D(B), D(E), \varphi\mathcal{U}_C\,\psi)$. If the next letter is causally in the future of x then it must be in $D(A) \cup D(B)$. If the next letter is in $D(B)$ then it cannot be in the past of z. We have to add $D(a)$ to D(B) if $a \in D(B) \cup C$ or if $a \notin \widetilde{E}$ and we may add it as well if we chose not to verify φ from the current vertex.

Note that, by the definition of δ, all accessible states in Q_5 are of the form $(D(A), D(B), D(E), \varphi\mathcal{U}_C\,\psi)$ for some subalphabets A, B, E with A, E connected and $D(A) \subseteq D(E)$. Moreover, if $(D(A) \cap \widetilde{E}) \setminus (D(B) \cup C) = \emptyset$ then the verification of ψ cannot be started. This explains the restrictions in the definition of Q_5 and the corresponding conditions in the definition of δ.

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 containment ordering on the subalphabets. The non increasing property is clear from the definition of the transition function. Note also that whenever the automaton changes its head direction then the next state is strictly smaller. Therefore, there are only finitely many changes of direction and each branch in the run-tree eventually moves to the left only or to the right only.

If a branch eventually moves to the left only then one of the $\sharp$ transition has to be taken which decides whether this branch is accepting or not.

If a branch eventually moves to the right only then eventually the state remains constant even if the branch is infinite since the automaton is very weak. The branch is accepting if its *ultimate* state is not in $Q_2 \cup Q_3 \cup Q_5$ since these states represent obligations that have to be eventually fulfilled. Note that states from Q_1 are only transient and a loop on a state from $Q_4 \cup Q_6 \cup Q_7$ moves to the left.

Proposition 20. *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 $w \in \Sigma^\infty$.

The number of states of the automaton $\mathcal{A}_\alpha$ is $|Q \cup \overline{Q}| \leq M|\alpha|$ where M is a polynomial in $|\{D(A) \mid A \subseteq \Sigma\}|$ whose degree is at most 3. Note that when the dependence relation is full, that is, when traces are actually words, then $|\{D(A) \mid A \subseteq \Sigma\}| = 2$ and the size of our automaton does not depend on the size of the alphabet.

Theorem 21. *Let M be defined as above and let M be bounded by some constant which is not part of the input. Then the satisfiability problem for the local logic* LocTL_Σ *is PSPACE-complete.*

Proof. The PSPACE-hardness follows from the word case. The PSPACE algorithm reduces the satisfiability problem for a formula $\alpha \in \mathrm{LocTL}_\Sigma$ to the non-emptiness problem for the two-way alternating automaton $\mathcal{A}_\alpha$ according to Proposition 20. Checking emptiness for a two-way alternating automaton can be done in PSPACE with respect to the size the automaton [11]. $\square$

7 Open problems

As we have seen in Section 3, the introduction of past modalities is unavoidable when constructing an expressively complete local temporal logic over traces. However, a natural question is whether unfiltered versions of the operators $\mathcal{U}_C$ and $\mathcal{S}_C$ are sufficient for expressive completeness. Also, we have seen in Theorem 14 that no EY are required for finite traces and that alphabetic sinces are sufficient. We do not know whether the same restriction applies to infinite traces. Another open issue is how to use our logic to express natural properties of traces.

References

1. R. Alur, D. Peled, and W. Penczek. Model-checking of causality properties. In *Proceedings of LICS'95*, pages 90–100, 1995.

2. J. Cohen, D. Perrin, and J.-E. Pin. On the expressive power of temporal logic. *Journal of Computer and System Sciences*, 46:271–295, 1993.
3. C. Courcoubetis, M.Y. Vardi, P. Wolper, and M. Yannakakis. Memory efficient algorithms for the verification of temporal properties. *formal Methods in System Design*, 1:275–288, 1992.
4. V. Diekert and P. Gastin. LTL is expressively complete for Mazurkiewicz traces. In *Proceedings of ICALP 2000*, number 1853 in LNCS, pages 211–222. Springer Verlag, 2000.
5. V. Diekert and P. Gastin. Local temporal logic is expressively complete for cograph dependence alphabets. In *Proceedings of LPAR'01*, number 2250 in LNAI, pages 55–69. Springer Verlag, 2001.
6. V. Diekert and P. Gastin. LTL is expressively complete for Mazurkiewicz traces. *Journal of Computer and System Sciences*, 2002. To appear.
7. V. Diekert and G. Rozenberg, editors. *The Book of Traces*. World Scientific, Singapore, 1995.
8. W. Ebinger and A. Muscholl. Logical definability on infinite traces. *Theoretical Computer Science*, 154:67–84, 1996.
9. W. Ebinger. *Charakterisierung von Sprachklassen unendlicher Spuren durch Logiken*. Dissertation, Institut für Informatik, Universität Stuttgart, 1994.
10. D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Proceedings of PoPL'80*, pages 163–173, Las Vegas, Nev., 1980.
11. 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.
12. J.A.W. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, California, 1968.
13. S. Ladner, R. Lipton, and L. Stockmeyer. Alternating pushown automata. In *Proceedings of FOCS'78*, pages 92–106, 1978.
14. S. Ladner, R. Lipton, and L. Stockmeyer. Alternating pushown automata. *SIAM Journal of Computing*, 13(1):135–155, 1984.
15. A. Mazurkiewicz. Concurrent program schemes and their interpretations. DAIMI Rep. PB 78, Aarhus University, Aarhus, 1977.
16. R. McNaughton and S. Papert. *Counter-free Automata*. MIT Press, 1971.
17. P. Niebert. A ν-calculus with local views for sequential agents. In *Proceedings of MFCS'95*, number 969 in LNCS, pages 563–573. Springer Verlag, 1995.
18. D. Perrin and J.-E. Pin. First order logic and star-free sets. *Journal of Computer and System Sciences*, 32:393–406, 1986.
19. D. Perrin. Recent results on automata and infinite words. In *Proceedings of MFCS'84*, number 176 in LNCS, pages 134–148. Springer Verlag, 1984.
20. A. Pnueli. The temporal logics of programs. In *Proceedings of the 18th IEEE FOCS, 1977*, pages 46–57, 1977.
21. R. Ramanujam. Locally linear time temporal logic. In *Proceedings of LICS'96*, pages 118–128, 1996.
22. M.-P. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8:190–194, 1965.
23. P.S. Thiagarajan and I. Walukiewicz. An expressively complete linear time temporal logic for Mazurkiewicz traces. In *Proceedings of LICS'97*, pages 183–194, 1997.
24. P.S. Thiagarajan. A trace based extension of linear time temporal logic. In *Proceedings of LICS'94*, pages 438–447, 1994.
25. W. Thomas. Star-free regular sets of ω-sequences. *Information and Control*, 42:148–156, 1979.

26. M.Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proceedings of LICS'86*, pages 322–331, 1986.
27. 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.
28. M.Y. Vardi. Reasonning about the past with two-way automata. In *Proceedings of ICALP'98*, number 1443 in LNCS, pages 628–641. Springer Verlag, 1998.
29. I. Walukiewicz. Difficult configurations – on the complexity of LTrL. In *Proceedings of ICALP'98*, number 1443 in LNCS, pages 140–151. Springer Verlag, 1998.
30. I. Walukiewicz. Private communication, 2001.
31. I. Walukiewicz. Local logics for traces. *Journal of Automata, Languages and Combinatorics*, 2002. To appear.
32. Th. Wilke. Classifying discrete temporal properties. Habilitationsschrift (post-doctoral thesis), April 1998.
33. Th. Wilke. Classifying discrete temporal properties. In *Proceedings of STACS'99*, number 1563 in LNCS, pages 32–46. Springer Verlag, 1999.
34. W. Zielonka. Notes on finite asynchronous automata. *R.A.I.R.O. — Informatique Théorique et Applications*, 21:99–135, 1987.