Pure future local temporal logics are expressively complete for Mazurkiewicz traces*

Volker Diekert[1] and Paul Gastin[2]

[1] FMI, Universität Stuttgart, Universitätsstrasse 38, D-70569 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. The paper settles a long standing problem for Mazurkiewicz traces: the pure future local temporal logic defined with the basic modalities *exists-next* and *until* is expressively complete. The analogous result with a global interpretation was solved some years ago by Thiagarajan and Walukiewicz (1997) and in its final form without any reference to past tense constants by Diekert and Gastin (2000). Each, the (previously known) global or the (new) local result generalizes Kamp's Theorem for words, because for sequences local and global viewpoints coincide. But traces are labelled partial orders and then the difference between an interpretation globally over cuts (configurations) or locally at points (events) is significant. For global temporal logics the satisfiability problem is non-elementary (Walukiewicz 1998), whereas for local temporal logics both the satisfiability problem and the model checking problem are sovable in PSPACE (Gastin and Kuske 2003) as in the case of words. This makes local temporal logics much more attractive.
Keywords: Temporal logics, Mazurkiewicz traces, concurrency.

1 Introduction

In various applications, the behaviour of a concurrent process is not represented by a string, but more accurately by some labelled partial order. This led Mazurkiewicz to the formulation of trace theory [16] which became a popular setting to study concurrency, see [8].

One advantage is that formal specifications of concurrent systems by temporal logic formulae have a direct (either global or local) interpretation for Mazurkiewicz traces. It is therefore no surprise that temporal logics for traces have received quite an attention, see [9, 2, 18–20, 23, 25]. In [26] (resp. finally in [5, 7]) it was shown that the basic global temporal logic with future tense operators and with (resp. without) past tense constants is expressively complete with respect to the first order theory. However the satisfiability problem for these global logics is non-elementary [28]. The main reason for this high complexity is that

* Work done while the second author stayed in Stuttgart within the MERCATOR program of the German Research Foundation DFG. Partial support of ACI Sécurité Informatique 2003-22 (VERSYDIS) is gratefully acknowledged.

the interpretation of a formula is defined with respect to a global configuration of the system, i.e., a finite prefix of the trace – and the prefix structure of traces is much more complex than in the case of linear orders (words). On the contrary, a local logic formula is evaluated at a local event of the system, i.e., at some vertex of the trace. The main advantage is that all local temporal logics over traces whose modalities are definable in monadic second order logic are decidable in PSPACE [12], i.e., both the satifiability problem and the model checking problem are decidable in PSPACE. This is optimal since the PSPACE-hardness occurs already for words.

The better complexity makes local temporal logic much more attractive; and several attempts were made to prove expressive completeness. In [6] expressive completeness for the basic pure future local temporal logic is established, if the underlying dependence alphabet is a cograph. Moreover, one can hope to go beyond cographs, only if each trace is equipped with some bottom element or if we allow past tense modalities. This second approach is used in [13, 14] to obtain expressive completeness for all dependence alphabet. In [13], the full power of *exists-previous* and *since* modalities equipped with filters is used. The result is improved in [14] where only past constants are necessary. Another temporal logic which is not local and based on more involved mocalities (including both past tense and future tense) was shown to be expressively complete and decidable in PSPACE [1]. However, the most basic question remained open: whether expressive completeness holds for a pure future local temporal logic based upon *exists-next* and *until*, only. The present paper gives a positive answer to this question.

Note that the focus of this paper is only to obtain the simplest possible pure future and expressively complete local temporal logic. In order to express easily properties of systems one should instead introduce all convenient MSO modalities since the satisfiability and the model checking problem remains decidable in PSPACE regardless of the fixed set of modalities used [12].

For lack of space we give only main ideas and skip several proofs including interesting new techniques in Section 5. They can be found in the full version at `http://www.liafa.jussieu.fr/~gastin/Articles/diegas03-2.html`.

2 Preliminaries

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 $a \in \Sigma$, the set of letters dependent of a is denoted by $D(a) = \{b \in \Sigma \mid (a, b) \in D\}$.

A *Mazurkiewicz trace* is an equivalence class of 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, and for all $x, y \in V$, $(\lambda(x), \lambda(y)) \in D$ 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. For $x \in V$, we also define $\Downarrow x = \{y \in V \mid y < x\}$, $\uparrow x = \{y \in V \mid x \leq y\}$, and $\Uparrow x = \{y \in V \mid x < y\}$.

The trace t is finite if V is finite and we denote by $\mathbb{M}(\Sigma, D)$ (or simply $\mathbb{M}$) the set of finite traces. By $\mathbb{R}(\Sigma, D)$ (or simply $\mathbb{R}$), we denote the set of finite or infinite traces (also called *real traces*). Let $\mathrm{alph}(t) = \lambda(V)$ be the alphabet of t and $\mathrm{alphinf}(t) = \{a \in \Sigma \mid \lambda^{-1}(a) \text{ is infinite}\}$ be the alphabet at infinity of t. For $A \subseteq \Sigma$, we let $\mathbb{R}_A = \{t \in \mathbb{R} \mid \mathrm{alph}(t) \subseteq A\}$ and $\mathbb{M}_A = \{t \in \mathbb{M} \mid \mathrm{alph}(t) \subseteq A\}$.

We denote by $\min(t)$ the set of minimal vertices of t. We let $\mathbb{R}^1 = \{t \in \mathbb{R} \mid |\min(t)| = 1\}$ be the set of traces with exactly one minimal vertex. To simplify the notation, we also use $\min(t)$ for the set $\lambda(\min(t))$ of labels of the minimal vertices of t. What we actually mean is always clear from the context.

If $U \subseteq V$ is an interval ($\uparrow x \cap \downarrow y \subseteq U$ for all $x, y \in U$) then $[U, \leq, \lambda]$ is a factor of t. We often identify U with $[U, \leq, \lambda]$. In particular, if $x \in V$ then $\downarrow x$ and $\Downarrow x$ are prefixes of t, and $\uparrow x$ and $\Uparrow x$ are suffixes of t. For $A \subseteq \Sigma$, the maximal prefix of t using actions from A only is $\mu_A(t) = \{x \in V \mid \lambda(\downarrow x) \subseteq A\}$.

For $x \in t$ and $C \subseteq \Sigma$ with $C \times C \subseteq D$, we denote by x_C the unique minimal vertex of $\Uparrow x \cap \lambda^{-1}(C)$ if it exists, i.e., when $\Uparrow x \cap \lambda^{-1}(C) \neq \emptyset$. Note that $x < x_C$ if x_C exists. If $C = \{c\}$ is a singleton, then we simply write x_c instead of $x_{\{c\}}$. When $a, b \in \Sigma$ are such that $\Uparrow x \cap \lambda^{-1}(b) \neq \emptyset$ and $\Uparrow x_b \cap \lambda^{-1}(c) \neq \emptyset$, we let $x_{bc} = (x_b)_c$ be the minimal vertex of $\Uparrow x_b \cap \lambda^{-1}(c)$.

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 w.l.o.g. 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 $\mathbb{M}$ of finite traces is then a monoid with the empty trace $1 = (\emptyset, \emptyset, \emptyset)$ as unit.

The concatenation of two trace languages $K, L \in \mathbb{R}$ is $K \cdot L = \{r \cdot s \mid r \in K, s \in L \text{ and } \mathrm{alphinf}(r) \times \mathrm{alph}(s) \subseteq I\}$.

We also use the infinite product $t = \prod_{i>0} t_i$ where $(t_i)_{i>0} \subseteq \mathbb{R}$ is a sequence of real traces such that $\mathrm{alphinf}(t_i) \times \mathrm{alph}(t_j) \subseteq I$ for all $i < j$.

3 Local temporal logics

Our main focus is on the local temporal logic based upon the two classical modalities *exists-next* and *until*. However, due to our proof techniques and in order to include other approaches we start with a logic which uses a more involved syntax. The more classical viewpoint and a process based viewpoint are treated in Sect. 7. Our main result appears as Thm. 17.

Let $\mathcal{C} \subseteq 2^{\Sigma} \setminus \{\emptyset\}$ be a *covering of Σ by (dependence-)cliques*, this means that $C \times C \subseteq D$ for all $C \in \mathcal{C}$, and for all $a \in \Sigma$, we have $a \in C$ for some $C \in \mathcal{C}$.

The syntax of the local temporal logic $\mathrm{LocTL}(\mathcal{C})$ is given by

$$\varphi ::= a \mid (\mathsf{X}_a \leq \mathsf{X}_b) \mid (\mathsf{X}_a \leq \mathsf{X}_{bc}) \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}_C\, \varphi \mid \varphi \,\mathsf{U}_C\, \varphi$$

where a, b, c range over Σ and C ranges over $\mathcal{C}$.

The semantics of LocTL($\mathcal{C}$) is defined as follows. Let $t \in \mathbb{R}^1$ and x be the minimal vertex of t.

$$
\begin{array}{ll}
t \models a & \text{if } \lambda(x) = a \\
t \models (\mathsf{X}_a \leq \mathsf{X}_b) & \text{if } x_a, x_b \text{ exist and } x_a \leq x_b \\
t \models (\mathsf{X}_a \leq \mathsf{X}_{bc}) & \text{if } x_a, x_{bc} \text{ exist and } x_a \leq x_{bc} \\
t \models \neg\varphi & \text{if } t \not\models \varphi \\
t \models \varphi \vee \psi & \text{if } t \models \varphi \text{ or } t \models \psi \\
t \models \mathsf{X}_C\, \varphi & \text{if } x_C \text{ exists and } \uparrow x_C \models \varphi \\
t \models \varphi\, \mathsf{U}_C\, \psi & \text{if } \exists z \in t\ [\lambda(z) \in C \text{ and } \uparrow z \models \psi \text{ and} \\
& \qquad\qquad \forall y \in \Downarrow z,\ \lambda(y) \in C \Rightarrow \uparrow y \models \varphi]
\end{array}
$$

We define some abbreviations. First, $A = \bigvee_{a\in A} a$ for $A \subseteq \Sigma$ and the strict version of U_C is $\varphi\, \mathsf{XU}_C\, \psi = \mathsf{X}_C(\varphi\, \mathsf{U}_C\, \psi)$.

Note that $\mathsf{X}_C\, \varphi = \bot \mathsf{XU}_C \varphi$ and $\varphi \mathsf{U}_C \psi = (C \wedge \psi) \vee ((\neg C \vee \varphi) \wedge \varphi \mathsf{XU}_C \psi)$. Hence, we could use XU_C as basic modality instead of X_C and U_C. In proofs, we often use an induction on the formula and depending on what is more convenient, we will either deal with XU_C or with X_C and U_C.

Second, $\mathsf{F}\,\varphi = \bigvee_{C\in\mathcal{C}} \top\, \mathsf{U}_C\, \varphi$ means that φ holds in the future. Similarly, $\mathsf{XF}\,\varphi = \bigvee_{C\in\mathcal{C}} \top\, \mathsf{XU}_C\, \varphi$ means that φ holds in the strict future. We also use $\mathsf{F}^\infty\, a = \mathsf{F}\, a \wedge \neg\, \mathsf{F}(a \wedge \neg\, \mathsf{XF}\, a)$ which means that there are infinitely many a's in the future.

Now, we want to define *initial satisfiability*, i.e., when does a trace $t \in \mathbb{R}$ satisfies a local temporal logic formula φ. Since a trace t does not necessarily have a unique minimal position, there is no canonical way to choose an initial position in t. Two approaches have been used.

In [6], an initial modality $\mathsf{EM}\,\varphi$ was introduced with the meaning $t \models \mathsf{EM}\,\varphi$ if there is a minimal position x in t with $\uparrow x \models \varphi$. Then, an initial formula α is a boolean combination of initial modalities. The local temporal logic based on EM, EX and U is expressively complete iff the dependence alphabet (Σ, D) is a cograph [6]. Actually, a local temporal logic based on EM and pure future modalities only cannot be expressively complete on P_4, i.e., on the dependence alphabet $a - b - c - d$ [6]. Hence, in order to get a pure future expressively complete local temporal logic as aimed in the present paper, we have to consider another definition for initial satisfiability.

The other approach used in e.g. [4] is to consider *rooted traces*. Let $\# \notin \Sigma$ and $t = [V, \leq, \lambda] \in \mathbb{R}(\Sigma, D)$. The rooted trace associated with t is

$$\#t = [V \cup \{\#\}, \leq \cup (\{\#\} \times (V \cup \{\#\})), \lambda \cup (\# \mapsto \#)].$$

It is a trace over the alphabet $\Sigma' = \Sigma \cup \{\#\}$ and the dependence relation $D' = D \cup (\{\#\} \times \Sigma) \cup (\Sigma \times \{\#\})$. Then, for a formula $\varphi \in$ LocTL($\mathcal{C}$), we define $\mathcal{L}_\Sigma(\varphi) = \{t \in \mathbb{R}(\Sigma, D) \mid \#t \models \varphi\}$. We simply write $\mathcal{L}(\varphi)$ when there is no ambiguity on the alphabet.

Alphabetic conditions can be easily expressed in LocTL($\mathcal{C}$): $\#t \models \mathsf{F}\, a$ iff $a \in \mathrm{alph}(t)$ and $\#t \models \mathsf{F}^\infty\, a$ iff $a \in \mathrm{alphinf}(t)$. Also, t has a minimal vertex labelled a iff $\#t \models \mathsf{F}\, a \wedge \bigwedge_{b \neq a} \neg(\mathsf{X}_b \leq \mathsf{X}_a)$. Therefore, for $A \subseteq \Sigma$, the languages $\mathbb{M}_A$, $\mathbb{R}_A$, $(\mathrm{alphinf} = A) = \{t \in \mathbb{R} \mid \mathrm{alphinf}(t) = A\}$ and $(\min \subseteq A) = \{t \in \mathbb{R} \mid \min(t) \subseteq A\}$ are definable in $\mathrm{LocTL}(\Sigma)$.

In the following, we show that $\mathrm{LocTL}(\mathcal{C})$ is expressively complete for each covering $\mathcal{C}$ of Σ by cliques of (Σ, D). The following lemmas will be useful.

Lemma 1. *Let $a, b \in \Sigma \cup \{\#\}$. For all $\varphi \in \mathrm{LocTL}(\mathcal{C})$, there exists $\widehat{\varphi} \in \mathrm{LocTL}(\mathcal{C})$ such that for all $t \in \mathbb{R}$ with $\min(t) \subseteq D(a) \cap D(b)$, then the assertions $at \models \varphi$ and $bt \models \widehat{\varphi}$ are equivalent. Moreover, if φ does not use the constants a and b then $\widehat{\varphi} = \varphi$.*

Proof. We proceed by induction on φ. We have $\widehat{a} = b$, $\widehat{b} = a$ and $\widehat{c} = c$ for all $c \neq a, b$. Now, $\widehat{(\mathsf{X}_a \leq \mathsf{X}_b)} = (\mathsf{X}_a \leq \mathsf{X}_b)$ and $\widehat{(\mathsf{X}_a \leq \mathsf{X}_{bc})} = (\mathsf{X}_a \leq \mathsf{X}_{bc})$. Finally, $\widehat{\varphi \,\mathsf{XU}_C\, \psi} = \varphi \,\mathsf{XU}_C\, \psi$. ☐

Lemma 2. *For all $\varphi \in \mathrm{LocTL}(\mathcal{C})$, there exists $\widehat{\varphi} \in \mathrm{LocTL}(\mathcal{C})$ such that for all $t \in \mathbb{R}^1$, we have $\#t \models \varphi$ iff $t \models \widehat{\varphi}$.*

Proof. We proceed by induction on φ. We have $\widehat{a} = \bot$, $\widehat{\varphi \,\mathsf{XU}_C\, \psi} = \varphi \,\mathsf{U}_C\, \psi$ and

$$\begin{aligned} \widehat{(\mathsf{X}_a \leq \mathsf{X}_b)} &= (a \wedge \mathsf{F}\, b) \vee (\neg b \wedge (\mathsf{X}_a \leq \mathsf{X}_b)) \\ \widehat{(\mathsf{X}_a \leq \mathsf{X}_{bc})} &= (a \wedge \mathsf{F}(b \wedge \mathsf{XF}\, c)) \vee (b \wedge (\mathsf{X}_a \leq \mathsf{X}_c)) \vee (\neg b \wedge (\mathsf{X}_a \leq \mathsf{X}_{bc})). \end{aligned}$$

☐

For $A \subseteq \Sigma$, the restriction of $\mathcal{C}$ to A is $\mathcal{C}_{|A} = \{C \cap A \mid C \in \mathcal{C}\} \setminus \{\emptyset\}$.

Lemma 3. *Let $A \subseteq \Sigma$. For all $\varphi \in \mathrm{LocTL}(\mathcal{C}_{|A})$, there exists $\widetilde{\varphi} \in \mathrm{LocTL}(\mathcal{C})$ such that for all $t \in \mathbb{R}_A$, we have $\#t \models \varphi$ if and only if $\#t \models \widetilde{\varphi}$.*

Proof. We use an induction on φ and we prove simultaneously that for all $t \in \mathbb{R}^1_A$, $t \models \varphi$ iff $t \models \widetilde{\varphi}$. We have $\widetilde{a} = a$, $\widetilde{(\mathsf{X}_a \leq \mathsf{X}_b)} = (\mathsf{X}_a \leq \mathsf{X}_b)$, $\widetilde{(\mathsf{X}_a \leq \mathsf{X}_{bc})} = (\mathsf{X}_a \leq \mathsf{X}_{bc})$. For each $C \in \mathcal{C}_{|A}$, we choose $\widetilde{C} \in \mathcal{C}$ such that $C = \widetilde{C} \cap A$. Then, we have $\widetilde{\varphi \,\mathsf{XU}_C\, \psi} = \widetilde{\varphi} \,\mathsf{XU}_{\widetilde{C}}\, \widetilde{\psi}$. ☐

4 Lifting Lemma

This section contains an important technical contribution. For $t \in \mathbb{R}^1$ with minimal vertex x and $A \subseteq \Sigma$, we let $t_A = \{x\} \cup \mu_A(\Uparrow x)$. We show by structural induction how to lift a formula φ to a formula $\overline{\varphi}^A$ in such a way that for all $t \in \mathbb{R}^1$, we have $t \models \overline{\varphi}^A$ if and only if $t_A \models \varphi$.

Proposition 4. *For all $\varphi \in \mathrm{LocTL}(\mathcal{C})$ and $A \subseteq \Sigma$, there exists $\overline{\varphi}^A \in \mathrm{LocTL}(\mathcal{C})$ such that for all $t \in \mathbb{R}^1$, we have $t_A \models \varphi$ if and only if $t \models \overline{\varphi}^A$.*

The rest of this section is devoted to the proof by structural induction of this proposition. The boolean connectives commute with the lifting: $\overline{\varphi \vee \psi}^A = \overline{\varphi}^A \vee \overline{\psi}^A$ and $\overline{\neg\varphi}^A = \neg\overline{\varphi}^A$. For the atomic formulae, we have $\overline{a}^A = a$ and

$$\overline{(\mathsf{X}_b \leq \mathsf{X}_c)}^A = (\mathsf{X}_b \leq \mathsf{X}_c) \wedge \bigwedge_{a \notin A} \neg(\mathsf{X}_a \leq \mathsf{X}_c)$$

$$\overline{(\mathsf{X}_b \leq \mathsf{X}_{cd})}^A = (\mathsf{X}_b \leq \mathsf{X}_{cd}) \wedge \bigwedge_{a \notin A} \neg(\mathsf{X}_a \leq \mathsf{X}_{cd})$$

We prove the second equality. The first one is simpler and left to the reader. Let $t \in \mathbb{R}^1$ and x be the minimal vertex of t. We have $t_A \models (\mathsf{X}_b \leq \mathsf{X}_{cd})$ iff x_b and x_{cd} exist, $x_b \leq x_{cd}$ and $x_{cd} \in t_A$. Now, $t \models (\mathsf{X}_b \leq \mathsf{X}_{cd})$ iff x_b and x_{cd} exist and $x_b \leq x_{cd}$. Finally, $x_{cd} \in t_A$ iff $\lambda(\Uparrow x \cap \downarrow x_{cd}) \subseteq A$ iff $t \models \bigwedge_{a \notin A} \neg(\mathsf{X}_a \leq \mathsf{X}_{cd})$.

The difficult part is indeed to lift the modalities X_C and U_C. We start with X_C which is simpler and allows to introduce the main ideas that will be used for U_C as well. For this we introduce a new macro and characterize its semantics.

$$\mathrm{Switch}_{A,B,c} = \overline{(\mathsf{X}_c \leq \mathsf{X}_c)}^A \wedge \bigwedge_{b \in B} \overline{(\mathsf{X}_c \leq \mathsf{X}_{cb})}^A \wedge \bigwedge_{b \notin B} \neg\overline{(\mathsf{X}_c \leq \mathsf{X}_{cb})}^A$$

Lemma 5. *Let $t \in \mathbb{R}^1$ and x be the minimal vertex of t. Then, $t \models \mathrm{Switch}_{A,B,c}$ if and only if x_c exists, $x_c \in t_A$ and $B = \lambda(\Uparrow x_c \cap t_A)$. In particular, $t \models \mathrm{Switch}_{A,B,c}$ implies $B \subseteq A$ and $\uparrow x_c \cap t_A = \{x_c\} \cup \mu_B(\Uparrow x_c) = (\uparrow x_c)_B$.*

Proof. Note that $t \models \overline{(\mathsf{X}_c \leq \mathsf{X}_c)}^A$ iff x_c exists and $x_c \in t_A$. Similarly, we have $t \models \overline{(\mathsf{X}_c \leq \mathsf{X}_{cb})}^A$ iff x_{cb} exists and $x_{cb} \in t_A$. Therefore, assuming that x_c exists, we have $\lambda(\Uparrow x_c \cap t_A) = \{b \in \Sigma \mid t \models \overline{(\mathsf{X}_c \leq \mathsf{X}_{cb})}^A\}$. The equivalence of the lemma is proved.

Now, assume that $t \models \mathrm{Switch}_{A,B,c}$. We have $B = \lambda(\Uparrow x_c \cap t_A) \subseteq A$ and $\uparrow x_c \cap t_A \subseteq (\uparrow x_c)_B$. Conversely, assume by contradiction that the inclusion is strict. Let y be minimal in $(\uparrow x_c)_B \setminus t_A$. Since $y \notin t_A$ we have $x_a \leq y$ for some $a \notin A$. We show that $y = x_{cb}$ where $b = \lambda(y)$. Since $b \in B \subseteq A$, we have $x_a < y$ and we find z such that $x_a \leq z \lessdot y$. Since $\lambda(z)$ and $b = \lambda(y)$ are dependent, the two vertices z and x_{cb} must be ordered. If $x_{cb} \leq z$ then $z \in (\uparrow x_c)_B \setminus t_A$, a contradiction with the minimality of y. Hence, $z < x_{cb}$ and we obtain $y \leq x_{cb}$. Since $x_c < y$, we deduce $y = x_{cb}$ as desired. Finally, we have $b \in B$, $a \notin A$ and $x_a \leq x_{cb}$, a contradiction with $t \models \mathrm{Switch}_{A,B,c}$. $\square$

For $C \in \mathcal{C}$ and $A, B \subseteq \Sigma$, we define the formula

$$\mathrm{Switch}_{A,B,C} = \bigvee_{c \in C} \left(\mathrm{Switch}_{A,B,c} \wedge \bigwedge_{d \in C \setminus \{c\}} \neg(\mathsf{X}_d \leq \mathsf{X}_c) \right).$$

Lemma 6.

$$\overline{\mathsf{X}_C\, \varphi}^A = \bigvee_{B \subseteq A} \mathrm{Switch}_{A,B,C} \wedge \mathsf{X}_C\, \overline{\varphi}^B.$$

Proof. Let $t \in \mathbb{R}^1$ and x be the minimal vertex of t. Note that $t \models \overline{\mathsf{X}_C\,\varphi}^A$ implies that x_C exists and $x_C \in t_A$. Also, by Lemma 5, $t \models \mathrm{Switch}_{A,B,C}$ implies that x_C exists and $x_C \in t_A$.

Assume that x_C exists and $x_C \in t_A$. Let $c = \lambda(x_C) \in C$ so that $x_C = x_c$. We have $t \models \bigwedge_{d \in C\setminus\{c\}} \neg(\mathsf{X}_d \le \mathsf{X}_c)$. Let $B = \lambda(\Uparrow x_c \cap t_A)$. By Lemma 5, we have $t \models \mathrm{Switch}_{A,B,c}$ and $\uparrow x_c \cap t_A = (\uparrow x_c)_B$. Hence, $t \models \mathrm{Switch}_{A,B,C}$.

Using the above results we get $t \models \overline{\mathsf{X}_C\,\varphi}^A$ iff $(\uparrow x_c)_B = \uparrow x_c \cap t_A \models \varphi$ iff $\uparrow x_c \models \overline{\varphi}^B$ (by induction of the formula) iff $t \models \mathsf{X}_C\,\overline{\varphi}^B$. ☐

It remains to deal with U_C. Note that $\neg C \wedge (\varphi \mathbin{\mathsf{U}_C} \psi) = \mathsf{X}_C(\varphi \mathbin{\mathsf{U}_C} \psi)$, hence, using Lemma 6, we get

$$\overline{\neg C \wedge (\varphi \mathbin{\mathsf{U}_C} \psi)}^A = \bigvee_{B \subseteq A} \mathrm{Switch}_{A,B,C} \wedge \mathsf{X}_C\,\overline{\varphi \mathbin{\mathsf{U}_C} \psi}^B$$

To deal with $\overline{C \wedge (\varphi \mathbin{\mathsf{U}_C} \psi)}^A$, we use an induction both on the formula and on A. First, we have $\overline{C \wedge (\varphi \mathbin{\mathsf{U}_C} \psi)}^\emptyset = C \wedge \overline{\psi}^\emptyset$ since $\mu_\emptyset(s) = 1$ for all $s \in \mathbb{R}$.

Lemma 7. *If $A \neq \emptyset$ then, $\overline{C \wedge (\varphi \mathbin{\mathsf{U}_C} \psi)}^A$ is equal to*

$$C \wedge \left(\overline{\varphi}^A \wedge \mathrm{Switch}_{A,A,C}\right) \mathsf{U}_C \left(\overline{\psi}^A \vee \left(\overline{\varphi}^A \wedge \bigvee_{B \subsetneq A} \mathrm{Switch}_{A,B,C} \wedge \mathsf{X}_C\,\overline{\varphi \mathbin{\mathsf{U}_C} \psi}^B\right)\right).$$

Proof. Let x be the minimal vertex of $t \in \mathbb{R}^1$. Assume that $t \models \overline{C \wedge (\varphi \mathbin{\mathsf{U}_C} \psi)}^A$. We find a sequence $x = x_0 < x_1 < \cdots < x_n$ in t_A such that $\downarrow x_n \cap \lambda^{-1}(C) = \{x_0, x_1, \ldots, x_n\}$, $t_A \cap \uparrow x_n \models \psi$ and $t_A \cap \uparrow x_k \models \varphi$ for all $0 \le k < n$. Let $0 \le j \le n$ be maximal such that $\uparrow x_i \models \mathrm{Switch}_{A,A,C}$ for all $0 \le i < j$. Using Lemma 5, we get easily by induction that $t_A \cap \uparrow x_i = (\uparrow x_i)_A$ for all $0 \le i \le j$. By induction on the formula, we get $\uparrow x_i \models \overline{\varphi}^A$ for $0 \le i < j$, and either $j < n$ and $\uparrow x_j \models \overline{\varphi}^A$ or $\uparrow x_n \models \overline{\psi}^A$. Hence, if $j = n$ we get $t \models (\overline{\varphi}^A \wedge \mathrm{Switch}_{A,A,C}) \mathbin{\mathsf{U}_C} \overline{\psi}^A$.

Assume now that $j < n$. Let $B = \lambda(\Uparrow x_{j+1} \cap t_A)$. By Lemma 5, we deduce $\uparrow x_j \models \mathrm{Switch}_{A,B,C}$ and $t_A \cap \uparrow x_{j+1} = (\uparrow x_{j+1})_B$. By maximality of j we must have $B \subsetneq A$.

We have $(\uparrow x_{j+1})_B = t_A \cap \uparrow x_{j+1} \models \varphi \mathbin{\mathsf{U}_C} \psi$. We obtain $\uparrow x_{j+1} \models \overline{\varphi \mathbin{\mathsf{U}_C} \psi}^B$ by induction on the alphabet. Therefore, $\uparrow x_j \models \overline{\varphi}^A \wedge \mathrm{Switch}_{A,B,C} \wedge \mathsf{X}_C\,\overline{\varphi \mathbin{\mathsf{U}_C} \psi}^B$ and $t \models (\overline{\varphi}^A \wedge \mathrm{Switch}_{A,A,C}) \mathbin{\mathsf{U}_C} \left(\overline{\varphi}^A \wedge \mathrm{Switch}_{A,B,C} \wedge \mathsf{X}_C\,\overline{\varphi \mathbin{\mathsf{U}_C} \psi}^B\right)$.

Conversely, assume that $t \models C \wedge (\overline{\varphi}^A \wedge \mathrm{Switch}_{A,A,C}) \mathbin{\mathsf{U}_C} \overline{\psi}^A$. We find a sequence $x = x_0 < x_1 < \cdots < x_n$ in t such that $\downarrow x_n \cap \lambda^{-1}(C) = \{x_0, x_1, \ldots, x_n\}$, $\uparrow x_n \models \overline{\psi}^A$ and $\uparrow x_i \models \overline{\varphi}^A \wedge \mathrm{Switch}_{A,A,C}$ for all $0 \le i < n$. Using Lemma 5 we get easily by induction that $x_i \in t_A$ and $t_A \cap \uparrow x_i = (\uparrow x_i)_A$ for all $0 \le i \le n$. By induction on the formula, we deduce that $t_A \cap \uparrow x_n = (\uparrow x_n)_A \models \psi$ and $t_A \cap \uparrow x_i = (\uparrow x_i)_A \models \varphi$ for all $0 \le i < n$. We get $t_A \models \varphi \mathbin{\mathsf{U}_C} \psi$.

Suppose $t \models C \wedge (\overline{\varphi}^A \wedge \mathrm{Switch}_{A,A,C}) \mathbin{\mathsf{U}_C} \left(\overline{\varphi}^A \wedge \mathrm{Switch}_{A,B,C} \wedge \mathsf{X}_C\,\overline{\varphi \mathbin{\mathsf{U}_C} \psi}^B\right)$ for some $B \subsetneq A$. We find a sequence $x = x_0 < x_1 < \cdots < x_n$ in t such that $\downarrow x_n \cap \lambda^{-1}(C) = \{x_0, x_1, \ldots, x_n\}$, $\uparrow x_n \models \overline{\varphi}^A \wedge \mathrm{Switch}_{A,B,C} \wedge \mathsf{X}_C\, \overline{\varphi \mathbin{\mathsf{U}_C} \psi}^B$ and $\uparrow x_k \models \overline{\varphi}^A \wedge \mathrm{Switch}_{A,A,C}$ for all $0 \le k < n$. As above, we deduce that $t_A \cap \uparrow x_k \models \varphi$ for all $0 \le k \le n$ and $t_A \cap \uparrow x_n = (\uparrow x_n)_A$. Since $\uparrow x_n \models \mathsf{X}_C\, \overline{\varphi \mathbin{\mathsf{U}_C} \psi}^B$, we have $\uparrow x_{n+1} \models \overline{\varphi \mathbin{\mathsf{U}_C} \psi}^B$ where x_{n+1} is the minimal vertex of $\Uparrow x_n \cap \lambda^{-1}(C)$. By induction on the alphabet, we deduce $(\uparrow x_{n+1})_B \models \varphi \mathbin{\mathsf{U}_C} \psi$. Now, $\uparrow x_n \models \mathrm{Switch}_{A,B,C}$ and from Lemma 5 we get $x_{n+1} \in (\uparrow x_n)_A \subseteq t_A$ and $(\uparrow x_{n+1})_B = (\uparrow x_n)_A \cap \uparrow x_{n+1} = t_A \cap \uparrow x_{n+1}$. We deduce that $t_A \cap \uparrow x_{n+1} \models \varphi \mathbin{\mathsf{U}_C} \psi$. Therefore $t_A \models \varphi \mathbin{\mathsf{U}_C} \psi$. $\square$

5 Expressive completeness of LocTL($\mathcal{C}$)

This section is devoted to the proof of our main theorem. We start with the definition of first order trace languages.

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 $t = [V, \le, \lambda]$ 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 $L = \mathcal{L}(\varphi)$.

Theorem 8. *Let $\mathcal{C}$ be a covering of Σ by cliques of (Σ, D). A real trace language over $\mathbb{R}(\Sigma, D)$ is expressible in $\mathrm{FO}_\Sigma(<)$ if and only if it is expressible in* LocTL($\mathcal{C}$).

From the very definition of the semantics of LocTL($\mathcal{C}$), each real trace language expressible in LocTL($\mathcal{C}$) is also expressible in $\mathrm{FO}_\Sigma(<)$. The rest of this section is devoted to the converse. For this, we use the well-known equivalence between first-order definability and aperiodicity for trace languages.

We use the algebraic notion of recognizability. Let $h : \mathbb{M} \to M$ be a morphism to a finite monoid M. For $s, t \in \mathbb{R}$, we say that s and t are h-similar, denoted $s \sim_h t$, if we can write $s = \prod_{i>0} s_i$ and $t = \prod_{i>0} t_i$ with $s_i, t_i \in \mathbb{M}$ and $h(s_i) = h(t_i)$ for all $i > 0$. The transitive closure $\approx_h$ of $\sim_h$ is an equivalence relation. For $t \in \mathbb{R}$, we denote by $[t]_{\approx_h}$ the equivalence class of t. When there is no ambiguity, we simply write $\sim$, $\approx$ and $[t]$. Since M is finite, the equivalence relation $\approx$ is of finite index with at most $|M|^2 + |M|$ equivalence classes. A trace language $L \subseteq \mathbb{R}$ is *recognized* by h if it is saturated by $\approx$ (or equivalently by $\sim$), i.e., $t \in L$ implies $[t] \subseteq L$ for all $t \in \mathbb{R}$.

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

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

Theorem 9 ([9, 10]). *A language $L \subseteq \mathbb{R}(\Sigma, D)$ is expressible in $\mathrm{FO}_\Sigma(<)$ if and only if it is aperiodic.*

Note that $(2^\Sigma, \cup)$ is an aperiodic monoid and the mapping $\mathrm{alph} : \mathbb{M} \to 2^\Sigma$ is a morphism. Hence, an aperiodic language $L \subseteq \mathbb{R}$ can be recognized by a surjective morphism h from $\mathbb{M}$ onto an aperiodic monoid M such that for all $r, s \in \mathbb{M}$, $h(r) = h(s)$ implies $\mathrm{alph}(r) = \mathrm{alph}(s)$. In this case, we say that h is an alphabetic morphism.

We fix an alphabetic morphism h from $\mathbb{M}$ to a finite aperiodic monoid M. Consider the finite set $N = \{[t] \mid t \in \mathbb{R}\} \cup \{0\}$. Define a product on N by $0 \cdot u = u \cdot 0 = 0$ for all $u \in N$ and

$$[s][t] = \begin{cases} [st] & \text{if } \mathrm{alphinf}(s) \times \mathrm{alph}(t) \subseteq I \\ 0 & \text{otherwise.} \end{cases}$$

We first check that this product is well-defined. Let $s^1 \sim s^2$ and $t^1 \sim t^2$. We find factorizations $s^j = \prod_{i>0} s_i^j$ and $t^j = \prod_{i>0} t_i^j$ for $j = 1, 2$ such that $h(s_i^1) = h(s_i^2)$ and $h(t_i^1) = h(t_i^2)$ for $i > 0$ and $\mathrm{alph}(s_i^1) \subseteq \mathrm{alphinf}(s^1)$ for $i > 1$. Since h is alphabetic, we have $\mathrm{alphinf}(s^1) = \mathrm{alphinf}(s^2)$, $\mathrm{alph}(t^1) = \mathrm{alph}(t^2)$ and $\mathrm{alph}(s_i^2) \subseteq \mathrm{alphinf}(s^2)$ for $i > 1$. Hence, either $[s^1][t^1] = 0 = [s^2][t^2]$ or $s^1 t^1 = \prod_{i>0} s_i^1 t_i^1$ and $s^2 t^2 = \prod_{i>0} s_i^2 t_i^2$. Then, we have $h(s_i^1 t_i^1) = h(s_i^2 t_i^2)$ for all $i > 0$ and we get $s^1 t^1 \sim s^2 t^2$. We deduce that $[s^1][t^1] = [s^1 t^1] = [s^2 t^2] = [s^2][t^2]$. Since $\approx$ is the transitive closure of $\sim$, an easy induction show that the product is well-defined.

One can easily check that the product operation on N is associative, hence $(N, \cdot)$ is a monoid. Moreover, if $t \in \mathbb{R}$ is infinite then $[t]^2 = 0 = [t]^3$ and when $t \in \mathbb{M}$ is finite then $[t]^n = [t^n] = h^{-1}(h(t^n)) = h^{-1}(h(t)^n)$. Since M is aperiodic, we deduce that N is aperiodic too. We also use the following result.

Lemma 10. *Let $t = \prod_{i<n} t_i$ with $n \in \overline{\mathbb{N}} = \mathbb{N} \cup \{\omega\}$ and $t_i \in \mathbb{R}$ for all $i < n$. Let $(s_i)_{i<n} \subseteq \mathbb{R}$ with $[s_i] = [t_i]$ for all $i < n$. Then, the product $s = \prod_{i<n} s_i$ is well-defined and $[s] = [t]$.*

Proof. Since $t = \prod_{i<n} t_i$ is a well-defined product, it contains finitely many infinite factors. Hence, we find $j \in \mathbb{N}$ with $j \leq n$ and the trace t_i is finite for all $j \leq i < n$. Since h is alphabetic, we also have s_i finite for all $j \leq i < n$. Hence, $h(s_i) = h(t_i)$ for all $j \leq i < n$ and we get $s' = \prod_{j \leq i<n} s_i \sim \prod_{j \leq i<n} t_i = t'$. We deduce that $[s_0] \cdots [s_{j-1}][s'] = [t_0] \cdots [t_{j-1}][t'] = [t] \neq 0$. Hence, the product $s_0 \cdots s_{j-1} s' = s$ is well-defined and $[s] = [t]$. $\square$

To prove that aperiodic trace languages are expressible in $\mathrm{LocTL}(\mathcal{C})$, we use an induction on Σ. If $\Sigma = \emptyset$ then there are only two trace languages: $\emptyset$ and $\mathbb{R} = \{1\}$ which are respectively defined by $\bot$ and $\top$. Assume now that $\Sigma \neq \emptyset$ and fix a covering $\mathcal{C}$ of Σ by cliques of (Σ, D). By induction, each aperiodic language

$L \subseteq \mathbb{R}_E$ with $E \subsetneq \Sigma$ is expressible in $\mathrm{LocTL}(\mathcal{C}_{|E})$. In the following, we fix some $C \in \mathcal{C}$ and we let $A = \Sigma \setminus C \subsetneq \Sigma$. We use the unambiguous decomposition $\mathbb{R} = \mathbb{R}_A(\min \subseteq C)$. Recall that $(\min \subseteq C) = \{t \in \mathbb{R} \mid \min(t) \subseteq C\}$.

Lemma 11. *Let $L \subseteq \mathbb{R}$ be a trace language recognized by h. Then, L is a finite union of languages of the form $(L_1 \cap \mathbb{R}_A)(L_2 \cap (\min \subseteq C))$, where the languages $L_1, L_2 \subseteq \mathbb{R}$ are recognized by h.*

Proof. We claim that $L = \bigcup ([t^1] \cap \mathbb{R}_A)([t^2] \cap (\min \subseteq C))$ where the union ranges over all $t^1 \in \mathbb{R}_A$ and $t^2 \in (\min \subseteq C)$ such that $t = t^1 t^2 \in L$. The inclusion $\subseteq$ is clear. Conversely, let $t = t^1 t^2 \in L$ with $t^1 \in \mathbb{R}_A$ and $t^2 \in (\min \subseteq C)$. Let $s^1 \in [t^1] \cap \mathbb{R}_A$ and $s^2 \in [t^2] \cap (\min \subseteq C)$. We have $[s^1][s^2] = [t^1][t^2] = [t] \neq 0$ hence the product $s = s^1 s^2$ is well defined and $[s] = [t]$. Since $t \in L$ and L is recognized by h, we deduce that $s \in L$. $\square$

In view of Lemma 11, a crucial step in the proof of Theorem 8 is to show that if L is an aperiodic trace language then $L \cap (\min \subseteq C)$ is expressible in $\mathrm{LocTL}(\mathcal{C})$. This is the aim of the next two lemmas.

Let $T = \{[t] \mid t \in \mathbb{R}^1 \cap C\mathbb{R}_A\}$. We consider T as a finite alphabet. Each trace $t \in (\min \subseteq C)$ has a unique *C-factorization* $t = \prod_{i<n} t_i$ with $n \in \mathbb{N} \cup \{\omega\}$ and $t_i \in \mathbb{R}^1 \cap C\mathbb{R}_A$ for all $i < n$. Hence, we can define a mapping $\sigma : (\min \subseteq C) \to T^\infty$ by $\sigma(t) = \prod_{i<n}[t_i]$ where $\prod_{i<n} t_i$ is the C-factorization of t.

Lemma 12. *Let $L \subseteq \mathbb{R}$ be recognized by h. Then, $L \cap (\min \subseteq C) = \sigma^{-1}(K)$ for some aperiodic word language $K \subseteq T^\infty$.*

Proof. Let $K = \sigma(L \cap (\min \subseteq C))$. We have clearly $L \cap (\min \subseteq C) \subseteq \sigma^{-1}(K)$. Conversely, let $t \in L \cap (\min \subseteq C)$ and $s \in \sigma^{-1}(\sigma(t)) \subseteq (\min \subseteq C)$. We have to show that $s \in L$. Let $t = \prod_{i<n} t_i$ and $s = \prod_{i<n} s_i$ be the C-factorizations of t and s. We have $[t_i] = [s_i]$ for $i < n$. By Lemma 10 we get $[s] = [t]$ and since $t \in L$ and L is recognized by h, we deduce that $s \in L$.

In order to prove that K is aperiodic, we use the evaluation morphism $e : T^* \to N$ defined by $e(u) = u$ for all $u \in T$. Note that, if $t \in (\min \subseteq C)$ has a finite C-factorization, i.e., $\sigma(t) \in T^*$, then $e(\sigma(t)) = [t]$. We deduce that if $r, s \in \sigma^{-1}(T^*)$ then, $e(\sigma(r)\sigma(s)) \neq 0$ if and only if $\mathrm{alphinf}(r) \times \mathrm{alph}(s) \subseteq I$.

We claim that for each $w \in T^*$ such that $e(w) \neq 0$, there exists $t \in (\min \subseteq C)$ such that $\sigma(t) = w$. We use an induction on $|w|$. If $w = 1$ then we take $t = 1$. Assume now that $w = uv$ with $u \in T^*$ and $v \in T$. By induction, we find $r \in (\min \subseteq C)$ such that $\sigma(r) = u$. By definition of T, there exists $s \in \mathbb{R}^1 \cap C\mathbb{R}_A$ such that $v = [s] = \sigma(s)$. We have $e(\sigma(r)\sigma(s)) = e(uv) = e(w) \neq 0$ hence $\mathrm{alphinf}(r) \times \mathrm{alph}(s) \subseteq I$ and we can consider $t = rs \in (\min \subseteq C)$. We have $\sigma(t) = \sigma(r)\sigma(s) = uv = w$.

Let $v, w \in T^\infty$ with $v \sim_e w$ and $v \in K$. Let $s \in L \cap (\min \subseteq C)$ with $\sigma(s) = v$. Write $v = \prod_{i>0} v_i$ and $w = \prod_{i>0} w_i$ with $v_i, w_i \in T^*$ and $e(v_i) = e(w_i)$ for all $i > 0$. We can write $s = \prod_{i>0} s_i$ with $s_i \in (\min \subseteq C)$ and $\sigma(s_i) = v_i$ for all $i > 0$. We have $e(w_i) = e(v_i) = [s_i] \neq 0$ and by the claim above we find $t_i \in (\min \subseteq C)$ with $\sigma(t_i) = w_i$ for all $i > 0$. We have $[t_i] = e(w_i) = e(v_i) = [s_i]$ hence by

Lemma 10 we deduce that the product $t = \prod_{i>0} t_i$ is well-defined and $[t] = [s]$. Since $s \in L$ and L is recognized by h, we get $t \in L$. We have $t \in (\min \subseteq C)$ and $w = \sigma(t) \in K$. $\square$

Lemma 13. *Suppose that each aperiodic trace language over A is expressible in* $\mathrm{LocTL}(\mathcal{C}_{|A})$. *Let $K \subseteq T^\infty$ be an aperiodic word language. There exists $\varphi \in$* $\mathrm{LocTL}(\mathcal{C})$ *such that for all $t \in (\min \subseteq C) \setminus \{1\}$, we have $\sigma(t) \in K$ if and only if $t \models \varphi$.*

Proof. Each aperiodic word language $K \subseteq T^\infty$ is expressible in $\mathrm{LTL}_T(\mathsf{XU})$ ((This is of course a classical result: [24, 15, 17, 27, 11, 21, 22, 3]). Therefore, we have to show that for all $f \in \mathrm{LTL}_T(\mathsf{XU})$, there exists $\widetilde{f} \in \mathrm{LocTL}(\mathcal{C})$ such that for all $t \in (\min \subseteq C) \setminus \{1\}$, we have $\sigma(t) \models f$ if and only if $t \models \widetilde{f}$. Clearly, we have $\widetilde{\bot} = \bot$, $\widetilde{\neg f} = \neg\widetilde{f}$ and $\widetilde{f \vee g} = \widetilde{f} \vee \widetilde{g}$.

Now, we claim that $\widetilde{f \,\mathsf{XU}\, g} = \widetilde{f} \,\mathsf{XU}_C\, \widetilde{g}$. Let $t \in (\min \subseteq C) \setminus \{1\}$ and consider its C-factorization $t = \prod_{i<n} s_i$. For $i \leq n$, we let $t_i = \prod_{i \leq j < n} s_j$. Then, $\sigma(t) \models f \,\mathsf{XU}\, g$ iff there exists $0 < j < n$ with $\sigma(t_j) \models g$ and $\sigma(t_k) \models f$ for all $0 < k < j$. By induction, we have $\sigma(t_j) \models g$ iff $t_j \models \widetilde{g}$ and $\sigma(t_k) \models f$ iff $t_k \models \widetilde{f}$ for all $0 < k < j$. Therefore, $\sigma(t) \models f \,\mathsf{XU}\, g$ iff $t \models \widetilde{f} \,\mathsf{XU}_C\, \widetilde{g}$.

Now, we consider the case $f = u \in T$. Let $v \in N$. There exists $s \in \mathbb{R}$ such that $v = [s] \subseteq \mathbb{R}$ is recognized by h. Hence the language $v \cap \mathbb{R}_A$ is aperiodic. Using the hypothesis of the lemma: we find $\alpha_v \in \mathrm{LocTL}(\mathcal{C}_{|A})$ such that for all $s \in \mathbb{R}_A$, $[s] = v$ if and only if $\#s \models \alpha_v$. Let $\overline{\alpha_v}^A \in \mathrm{LocTL}(\mathcal{C})$ be the formula obtained using the Lifting Lemma (Proposition 4).

We claim that $\widetilde{u} = \bigvee\ c \wedge \overline{\alpha_v}^A$ where the disjunction ranges over all $c \in C$ and $v \in N$ such that $[c]v = u$. Note that, if $t \in (\min \subseteq C) \setminus \{1\}$ then the first factor of the C-factorization of t is t_A as defined in Sect. 4.

Assume that $\sigma(t) \models u$. Then, $[t_A] = u$. Write $t_A = cs$ with $c \in C$ and let $v = [s] \in N$. We have $u = [c]v$ and since $s \in \mathbb{R}_A$ we get $\#s \models \alpha_v$. Using Lemma 1 we deduce that $t_A = cs \models \alpha_v$ and by Proposition 4 we obtain $t \models c \wedge \overline{\alpha_v}^A$.

Conversely, assume that $t \models c \wedge \overline{\alpha_v}^A$ for some $c \in C$ and $v \in N$ such that $u = [c]v$. Since $t \models c$, we can write $t_A = cs$ with $s \in \mathbb{R}_A$. Using Proposition 4, we deduce from $t \models \overline{\alpha_v}^A$ that $t_A = cs \models \alpha_v$. We get $\#s \models \alpha_v$ by Lemma 1 and since $s \in \mathbb{R}_A$ we obtain $[s] = v$. Therefore, $[t_A] = [c]v = u$ and we get $\sigma(t) \models u$. $\square$

We now turn to the proof of Theorem 8. Using Lemma 11, we have to show that if L_1 and L_2 are recognized by h then $L = (L_1 \cap \mathbb{R}_A)(L_2 \cap (\min \subseteq C))$ is expressible in $\mathrm{LocTL}(\mathcal{C})$.

Since $L_1 \cap \mathbb{R}_A$ is aperiodic, using the induction on the alphabet, we find $\varphi_1 \in \mathrm{LocTL}(\mathcal{C}_{|A})$ such that for all $t_1 \in \mathbb{R}_A$, $t_1 \in L_1$ iff $\#t_1 \models \varphi_1$. Let $\overline{\varphi_1}^A \in \mathrm{LocTL}(\mathcal{C})$ be the formula given by the Lifting Lemma (Proposition 4). For all $t \in \mathbb{R}$, we have $\#t \models \overline{\varphi_1}^A$ iff $\#\mu_A(t) \models \varphi_1$ iff $\mu_A(t) \in L_1$.

Since L_2 is recognized by h, using Lemma 12 we have $L_2 \cap (\min \subseteq C) = \sigma^{-1}(K)$ for some aperiodic word language $K \subseteq T^\infty$. By Lemma 13, we find $\varphi_2 \in \mathrm{LocTL}(\mathcal{C})$ such that for all $t_2 \in (\min \subseteq C) \setminus \{1\}$ we have $t_2 \models \varphi_2$ iff

$\sigma(t_2) \in K$ iff $t_2 \in L_2$. Let $\widetilde{\varphi_2} = \neg \mathsf{X}_C \top \vee \mathsf{X}_C \varphi_2$ if $1 \in L_2$ and $\widetilde{\varphi_2} = \mathsf{X}_C \varphi_2$ otherwise.

We claim that $L = \mathcal{L}_\Sigma(\varphi)$ where $\varphi = \overline{\varphi_1}^A \wedge \widetilde{\varphi_2}$.

Let $t \in L$ and write $t = t_1 t_2$ with $t_1 \in L_1 \cap \mathbb{R}_A$ and $t_2 \in L_2 \cap (\min \subseteq C)$. We have $t_1 = \mu_A(t)$ hence we get $\#t \models \overline{\varphi_1}^A$. Either $t_2 = 1$ and $\#t \models \neg \mathsf{X}_C \top$ or $t_2 \neq 1$ and $t_2 \models \varphi_2$ and $\#t \models \mathsf{X}_C \varphi_2$. Therefore, $\#t \models \varphi$.

Conversely, assume that $\#t \models \varphi$ and write $t = t_1 t_2$ with $t_1 \in \mathbb{R}_A$ and $t_2 \in (\min \subseteq C)$. We have $t_1 = \mu_A(t) \in \mathbb{R}_A$ and since $\#t \models \overline{\varphi_1}^A$ we get $t_1 \in L_1$. Now, if $t_2 \neq 1$ then $\#t \models \widetilde{\varphi_2}$ implies $\#t \models \mathsf{X}_C \varphi_2$. Hence $t_2 \models \varphi_2$ and we get $t_2 \in L_2$. Therefore, $t = t_1 t_2 \in L$. Finally, if $t_2 = 1$ then $\#t \models \widetilde{\varphi_2}$ implies $\widetilde{\varphi_2} = \neg \mathsf{X}_C \top \vee \mathsf{X}_C \varphi_2$. Hence, $t_2 = 1 \in L_2$ and we get $t = t_1 t_2 \in L$.

6 Removing constants

In this section, we show how to remove all constants of the form $(\mathsf{X}_a \leq \mathsf{X}_{bc})$. We use notation like $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc})$ for $\bowtie \in \{\leq, =\}$ with the expected meaning that x_{ac} and x_{bc} exist and are ordered as specified.

For $a \in \Sigma$, choose $C \in \mathcal{C}$ such that $a \in C$ and define $\mathsf{X}_a \varphi := \neg a \, \mathsf{X} \mathsf{U}_C \, (a \wedge \varphi)$. As expected, if $t \in \mathbb{R}^1$ and x is the minimal vertex of t, then $t \models \mathsf{X}_a \varphi$ means both x_a exists and $\uparrow x_a \models \varphi$. Similarly, we define $\varphi \, \mathsf{U}_a \, \psi := (\neg a \vee \varphi) \, \mathsf{U}_C \, (a \wedge \psi)$, and $t \models \varphi \, \mathsf{U}_a \, \psi$ means: there exists $z \geq x$ with $\lambda(z) = a$ and $\uparrow z \models \psi$ and for all $x \leq y < z$ such that $\lambda(y) = a$ we have $\uparrow y \models \varphi$.

Clearly, we may assume that in all constants of the form $(\mathsf{X}_a \leq \mathsf{X}_{bc})$ we have $b \neq a \neq c$, because for $a = b$ or $a = c$ the meaning of $(\mathsf{X}_a \leq \mathsf{X}_{bc})$ is simply the statement $\mathsf{X}_b \mathsf{X}_c \top$. For $a \neq c$ we can identify $(\mathsf{X}_a \leq \mathsf{X}_{bc})$ with $(\mathsf{X}_{ac} \leq \mathsf{X}_{bc})$.

For $a, b, c \in \Sigma$, we introduce also the following macros.

$$\begin{aligned}
(\mathsf{X}_a = \mathsf{X}_b) &= (\mathsf{X}_a \leq \mathsf{X}_b) \wedge (\mathsf{X}_b \leq \mathsf{X}_a) \\
(\mathsf{X}_a < \mathsf{X}_b) &= (\mathsf{X}_a \leq \mathsf{X}_b) \wedge \neg(\mathsf{X}_b \leq \mathsf{X}_a) \\
(\mathsf{X}_a \parallel \mathsf{X}_b) &= \mathsf{X}_a \top \wedge \mathsf{X}_b \top \wedge \neg(\mathsf{X}_a \leq \mathsf{X}_b) \wedge \neg(\mathsf{X}_b \leq \mathsf{X}_a) \\
(\mathsf{X}_c < \mathsf{X}_{ac}) &= \mathsf{X}_a \mathsf{X}_c \top \wedge \neg(\mathsf{X}_a < \mathsf{X}_c)
\end{aligned}$$

Lemma 14. *We may assume that $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc})$ always occur in conjunction with*

$$(\mathsf{X}_a \parallel \mathsf{X}_b) \wedge (\mathsf{X}_c < \mathsf{X}_{ac}) \wedge (\mathsf{X}_c < \mathsf{X}_{bc}) \wedge \neg((\mathsf{X}_c < \mathsf{X}_a) \wedge (\mathsf{X}_c < \mathsf{X}_b)).$$

Proof. This is a consequence of the following equalities:

$$(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}) = ((\mathsf{X}_c < \mathsf{X}_a) \wedge (\mathsf{X}_c < \mathsf{X}_b)) \mathsf{U}_c ((\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}) \wedge \neg((\mathsf{X}_c < \mathsf{X}_a) \wedge (\mathsf{X}_c < \mathsf{X}_b)))$$

$$\begin{aligned}
(\mathsf{X}_{ac} \leq \mathsf{X}_{bc}) \wedge (\mathsf{X}_a \leq \mathsf{X}_b) &= \mathsf{X}_b \mathsf{X}_c \top \wedge (\mathsf{X}_a \leq \mathsf{X}_b), \\
(\mathsf{X}_{ac} \leq \mathsf{X}_{bc}) \wedge (\mathsf{X}_b < \mathsf{X}_a) &= \mathsf{X}_b (\mathsf{X}_a < \mathsf{X}_c) \wedge (\mathsf{X}_b < \mathsf{X}_a), \\
(\mathsf{X}_{ac} \leq \mathsf{X}_{bc}) \wedge (\mathsf{X}_a < \mathsf{X}_c) &= \mathsf{X}_b \mathsf{X}_c \top \wedge (\mathsf{X}_a < \mathsf{X}_c), \\
(\mathsf{X}_{ac} \leq \mathsf{X}_{bc}) \wedge (\mathsf{X}_b < \mathsf{X}_c) &= (\mathsf{X}_a < \mathsf{X}_c) \wedge (\mathsf{X}_b < \mathsf{X}_c).
\end{aligned}$$

Note that $(\mathsf{X}_{ac} \leq \mathsf{X}_{bc}) \wedge \neg(\mathsf{X}_c < \mathsf{X}_{ac}) = (\mathsf{X}_{ac} \leq \mathsf{X}_{bc}) \wedge (\mathsf{X}_a < \mathsf{X}_c)$. ∎

The following technical lemma is crucial to remove the constant $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc})$.

Lemma 15. *1. Let z be a vertex such that $\lambda(z) = a$ and z_c exists. There exist letters $\{a_1, \ldots, a_{k-1}\} \subseteq \Sigma \setminus \{a, c\}$ such that $z < z_{a_1} < \cdots < z_{a_{k-1}} < z_c$ and $a = a_0 — a_1 — \cdots — a_{k-1} — a_k = c$ in (Σ, D).*

2. Let x be a vertex and $\{a_1, \ldots, a_{k-1}\} \subseteq \Sigma \setminus \{a, c\}$ such that $x_a < x_{aa_1} < \cdots < x_{aa_{k-1}} < x_{ac}$ and $a = a_0 — a_1 — \cdots — a_{k-1} — a_k = c$ in (Σ, D). If $x_a \parallel x_c$ and $x_c < x_{ac}$ then $x_{aa_i} = x_{ca_i}$ for some $1 \leq i < k$.

Proof. 1. The proof is done by induction on $\uparrow z \cap \Downarrow z_c$. Let $y \in \uparrow z \cap \Downarrow z_c$ be minimal such that $\lambda(y)$ depends on c. If $y = z$ then we have $\lambda(y) = a — c$ and we take $k = 1$. Assume now that $z < y$. By definition of y, we have $b = \lambda(y) \in \Sigma \setminus \{a, c\}$ and $y = z_b < z_c$. By induction, we find letters $\{a_1, \ldots, a_{k-2}\} \subseteq \Sigma \setminus \{a, b\}$ such that $z < z_{a_1} < \cdots < z_{a_{k-2}} < z_b$ and $a = a_0 — a_1 — \cdots — a_{k-2} — a_{k-1} = b$ in (Σ, D). We conclude easily since $z_b < z_c$, $b — c$ and $c \notin \{a_1, \ldots, a_{k-2}\}$ by minimality of y.

2. Since $x_a \parallel x_c$, we have $(a, c) \in I$ and $k \geq 2$. The vertices x_c and $x_{aa_{k-1}}$ must be ordered. If $x_{aa_{k-1}} \leq x_c$ then $x_c = x_{ac}$, a contradiction. Hence, $x_c < x_{aa_{k-1}}$ and we can choose $0 < i < k$ minimal with $x_c < x_{aa_i}$. This implies $x_{ca_i} \leq x_{aa_i}$. We show that $x_{ca_i} = x_{aa_i}$. If $i = 1$ we let $y = x_a$ and if $i > 1$ we let $y = x_{aa_{i-1}}$. So, $(\lambda(y), a_i) \in D$ and y and x_{ca_i} are ordered. If $x_{ca_i} \leq y$ then $x_c < y$ and this excludes the case $i = 1$ since $x_a \parallel x_c$. Then, we get $x_c < x_{ca_i} \leq y = x_{aa_{i-1}}$ which contradicts the minimality of i. Therefore, $y < x_{ca_i}$ and using $y_{a_i} = x_{aa_i}$, we deduce that $x_{aa_i} \leq x_{ca_i}$ and therefore $x_{ca_i} = x_{aa_i}$. $\square$

Let $a, c \in \Sigma$ and let $t \in \mathbb{R}$, $x \in t$ such that x_{ac} exists. Define $\delta_x(a, c)$ as the smallest integer $k \geq 1$ such that there exist letters $a_1, \cdots, a_{k-1}$ such that $x_a < x_{aa_1} < \cdots < x_{aa_{k-1}} < x_{ac}$ and $a = a_0 — a_1 — \cdots — a_{k-1} — a_k = c$ in (Σ, D). Note that such an integer k exists by Lemma 15.

We also introduce the set $F_x(a, c)$ which consists of all pairs (d, e) such that either x_{de} does not exist or $x_{ac} < x_{de}$. Throughout we use the fact that if $x \leq y$ and $y_{fg} \leq x_{ac}$, then $F_x(a, c) \subseteq F_y(f, g)$. This is trivial since if $x \leq y$ and y_{de} exists, then x_{de} exists and $x_{de} \leq y_{de}$. Moreover, if $x \leq y$ and $y_{fg} < x_{ac}$, then $F_x(a, c) \subsetneq F_y(f, g)$ since $(a, c) \in F_y(f, g)$ (even if y_{ac} does not exist).

We have to remove the constants $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc})$. For this, we introduce two parameters $0 \leq m \leq |\Sigma|^2$, $1 \leq \mu \leq 2|\Sigma|$ and the formula $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}, m, \mu)$. The semantics is exactly the same as $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc})$ but the pair (m, μ) refers to some context information insuring that whenever we test $\uparrow x \models (\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}, m, \mu)$ (and the evaluation depends on this test), then we have $|F_x(a, c)| \geq m$, $|F_x(b, c)| \geq m$ and $\delta_x(a, c) + \delta_x(b, c) \leq \mu$. Clearly, we have $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}) = (\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}, 0, 2|\Sigma|)$. The formula $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}, m, \mu)$ is defined by induction on the pair (m, μ). In each round we either increase m or m remains unchanged and μ decreases. Note that we can replace $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}, m, 1)$ by *false* since $\delta_x(a, c) + \delta_x(b, c) \leq 1$ is not possible. This replacement forms the basis of the inductive definition.

Let $t \in \mathbb{R}$ be a trace and $x \in t$ such that both x_{ac} and x_{bc} exist. By Lemma 14 we may assume that $x_a \parallel x_b$, $x_c < x_{ac}$, $x_c < x_{bc}$ and $\neg((x_c < x_a) \wedge (x_c < x_b))$

whenever we perform a test $\uparrow x \models (\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc})$, and this works also for a test $\uparrow x \models (\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}, m, \mu)$ without changing the value of the pair (m, μ).

We can *compute*[3] $k = \delta_x(a, c)$ and the corresponding letters $a_1, \ldots, a_{k-1}$. If $x_c < x_a$ then we let $i = 0$ and $a_0 = a$. If $x_a \parallel x_c$, by Lemma 15 we know that there is some $1 \leq i < k$ with $x_{aa_i} = x_{ca_i}$. We show how to *compute* i.

Since $(a_{k-1}, c) \in D$, x_c and $x_{aa_{k-1}}$ are ordered and $x_{aa_{k-1}} \leq x_c$ is not possible since $x_c < x_{ac}$. Therefore, $x_c < x_{aa_{k-1}}$ and $x_{ca_{k-1}} \leq x_{aa_{k-1}} < x_{ac}$. This implies $|F_x(c, a_{k-1})| > |F_x(a, c)| \geq m$ and $|F_x(a, a_{k-1})| > |F_x(a, c)| \geq m$ as seen above. Hence, by induction, we can use the formula $(\mathsf{X}_{aa_{k-1}} = \mathsf{X}_{ca_{k-1}}, m + 1, 2|\Sigma|)$ to test whether $x_{aa_{k-1}} = x_{ca_{k-1}}$. If yes then we let $i = k - 1$. Otherwise, we must have $x_{ca_{k-1}} < x_{aa_{k-1}}$. Since $(a_{k-2}, a_{k-1}) \in D$, $x_{ca_{k-1}}$ and $x_{aa_{k-2}}$ are ordered and $x_{aa_{k-2}} < x_{ca_{k-1}}$ is not possible since $x_{ca_{k-1}} < x_{aa_{k-1}}$. Hence, $x_c < x_{ca_{k-1}} \leq x_{aa_{k-2}}$ and therefore $x_{ca_{k-2}} \leq x_{aa_{k-2}} < x_{ac}$. As above, we get $|F_x(c, a_{k-2})| > |F_x(a, c)| \geq m$ and $|F_x(a, a_{k-2})| > |F_x(a, c)| \geq m$. By induction, we can use the formula $(\mathsf{X}_{aa_{k-2}} = \mathsf{X}_{ca_{k-2}}, m + 1, 2|\Sigma|)$ to test whether $x_{aa_{k-2}} = x_{ca_{k-2}}$. If yes then we let $i = k - 2$. We continue in this way until we find $1 \leq i < k$ with $x_{aa_i} = x_{ca_i}$.

Similarly, we *compute* $\ell = \delta_x(b, c)$ and the corresponding letters $b_1, \ldots, b_{\ell-1}$. If $x_c < x_b$ then we let $j = 0$ and $b_0 = b$. If $x_b \parallel x_c$, by Lemma 15 we know that there is some $1 \leq j < \ell$ with $x_{bb_j} = x_{cb_j}$. We can *compute* j as above.

In the following, we let $y = x_c$.

Assume $b = c$. We have $x_a \parallel x_b = x_c$ and we can use the integer $1 \leq i < k$ such that $x_{aa_i} = x_{ca_i}$. Then $x_{bc} = y_c \leq y_{a_i c} = x_{ac}$ and we may replace $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}, m, \mu)$ by $\mathsf{X}_c(\mathsf{X}_{a_i} < \mathsf{X}_c)$.

Similarly, if $a = c$ then we have $x_b \parallel x_c$ and we have computed $1 \leq j < \ell$ such that $x_{bb_j} = x_{cb_j}$. We have $x_{ac} = y_c \leq y_{b_j c} = x_{bc}$ hence we may replace $(\mathsf{X}_{ac} \leq \mathsf{X}_{bc}, m, \mu)$ by *true* and $(\mathsf{X}_{ac} = \mathsf{X}_{bc}, m, \mu)$ by $\mathsf{X}_c(\mathsf{X}_{b_j} < \mathsf{X}_c)$.

The last case is $a \neq c \neq b$. Note that $x_a < x_c$ is not possible since $x_c < x_{ac}$. Hence $a \neq c$ implies either $x_c < x_a$ or $x_a \parallel x_c$. In both cases, we have *computed* $0 \leq i < k$ with $x_{aa_i} = x_{ca_i}$. For the same reason, we have *computed* $0 \leq j < \ell$ with $x_{bb_j} = x_{cb_j}$.

We do not have $i = j = 0$. Hence $\delta_y(a_i, c) + \delta_y(b_j, c) < \delta_x(a, c) + \delta_x(b, c)$. Also, $|F_y(a_i, c)| \geq |F_x(a, c)| \geq m$ and $|F_y(b_j, c)| \geq |F_x(b, c)| \geq m$. Therefore, we can replace $(\mathsf{X}_{ac} \bowtie \mathsf{X}_{bc}, m, \mu)$ by $\mathsf{X}_c(\mathsf{X}_{a_i c} \bowtie \mathsf{X}_{b_j c}, m, \mu - 1)$ which is already defined by induction.

Theorem 16. *Let $\mathcal{C}$ be a covering of Σ by cliques of (Σ, D). The local temporal logic* $\mathrm{LocTL}(\mathsf{X}_C, \mathsf{U}_C, (\mathsf{X}_a \leq \mathsf{X}_b))$ *using modalities X_C and U_C for $C \in \mathcal{C}$ and constants* $(\mathsf{X}_a \leq \mathsf{X}_b)$ *with $\{a, b\} \not\subseteq C$ for all $C \in \mathcal{C}$ is expressively complete.*

Proof. We have seen in this section that we can remove all constants of the form $(\mathsf{X}_a \leq \mathsf{X}_{bc})$. We have $(\mathsf{X}_a \leq \mathsf{X}_a) = \mathsf{X}_a \top$ and $(\mathsf{X}_a \leq \mathsf{X}_b) = \mathsf{X}_b \top \wedge \neg(\neg a \,\mathsf{X}\mathsf{U}_C\, b)$ when $a \neq b$ and $\{a, b\} \subseteq C \in \mathcal{C}$. Hence, this is a corollary of Theorem 8. $\square$

[3] Computing means here creating a huge disjunction.

7 Other expressively complete logics

In this section, we introduce other natural local temporal logics over traces and show that they are expressively complete using Theorem 16.

We want to replace the C-modalities X_C and U_C by more classical modalities. We are interested in the usual *exists-next* and *until* modalities whose semantics is given for a trace $t \in \mathbb{R}^1$ with minimal vertex x by

$$\begin{aligned} t &\models \mathsf{EX}\,\varphi \ \text{ if } \ \exists y\ (x \lessdot y \text{ and } {\uparrow} y \models \varphi) \\ t &\models \varphi\,\mathsf{U}\,\psi \ \text{ if } \ \exists z\ (x \leq z \text{ and } {\uparrow} z \models \psi \text{ and } \forall y\ (x \leq y < z) \Rightarrow {\uparrow} y \models \varphi). \end{aligned}$$

Theorem 17. *The local temporal logic* LocTL(EX, U) *based on the modalities* EX *and* U *is expressively complete.*

Proof. This results is a consequence of Theorem 16 applied with the covering of Σ by singletons: $\mathcal{C} = \{\{c\} \mid c \in \Sigma\}$. We have

$$\begin{aligned} \varphi\,\mathsf{U}_c\,\psi &= (\neg c \vee \varphi)\,\mathsf{U}\,(c \wedge \psi), \\ \mathsf{X}_c\,\varphi &= (\neg c \wedge (\perp \mathsf{U}_c\,\varphi)) \vee (c \wedge \mathsf{EX}(\perp \mathsf{U}_c\,\varphi)). \end{aligned}$$

Hence, it remains to show how to express constants of the form $(\mathsf{X}_a \leq \mathsf{X}_b)$. We have

$$(\mathsf{X}_a \leq \mathsf{X}_b) = \bigvee_{c \in \Sigma} (\mathsf{X}_c \leq \mathsf{X}_a) \wedge (\mathsf{X}_c \leq \mathsf{X}_b) \wedge \mathsf{EX}(c \wedge \neg(\neg a\,\mathsf{U}\,b)).$$

Thus, it is enough to consider a conjunction $(\mathsf{X}_c \leq \mathsf{X}_a) \wedge \mathsf{EX}\,c$. For $\lambda(x) = a$, this is $\mathsf{EX}(c \wedge \mathsf{F}\,a)$. For $\lambda(x) \neq a$, this is $\mathsf{EX}(c \wedge \mathsf{F}\,a) \wedge \neg(\neg c\,\mathsf{U}\,a)$. ☐

We could also consider the *strict-until* SU whose semantics is given for a trace $t \in \mathbb{R}^1$ with minimal vertex x by

$$t \models \varphi\,\mathsf{SU}\,\psi \text{ if } \exists z\ (x < z \text{ and } {\uparrow} z \models \psi \text{ and } \forall y\ (x < y < z) \Rightarrow {\uparrow} y \models \varphi).$$

The local temporal logic LocTL(SU) based on the modality SU is expressively complete since we have $\mathsf{EX}\,\varphi = \perp \mathsf{SU}\,\varphi$ and $\varphi\,\mathsf{U}\,\psi = \psi \vee (\varphi \wedge (\varphi\,\mathsf{SU}\,\psi))$.

Finally, we show that we can deal also with process-based logics as introduced in [25]. In this framework, we start with a finite set of processes $\mathcal{P} = \{1, \ldots, n\}$ and a mapping $p : \Sigma \to 2^{\mathcal{P}} \setminus \{\emptyset\}$. The execution of an action $a \in \Sigma$ requires the participation of all processes in the nonempty set $p(a)$. If $p(a) = \{i\}$ is a singleton then the action a is local to process i. Otherwise, the execution of a requires the synchronization of all processes in $p(a)$. The dependence relation is $D = \{(a, b) \in \Sigma^2 \mid p(a) \cap p(b) \neq \emptyset\}$. Hence the set $\mathcal{C} = \{p^{-1}(i) \mid i \in \mathcal{P}\}$ is a covering of Σ by cliques of (Σ, D).

Thanks to this more concrete view of the dependence alphabet based on processes, we can define temporal modalities that involve locations of actions. In [25], the formula $\mathcal{O}_i\varphi$ means that φ holds at the first event of process i that is not in the past of the current vertex. Clearly, this is not a future modality.

Here, we use a future variant $\mathsf{X}_i\,\varphi$ meaning that φ holds at the first event of process i which is strictly above the current vertex. More formally, we define $\mathsf{X}_i\,\varphi := \mathsf{X}_{p^{-1}(i)}\,\varphi$. The until modality introduced in [25] is also not pure future. Here we use a future variant $\varphi\,\mathsf{U}_i\,\psi$ which means that on the sequence of events located on process i and above the current vertex we observe φ until ψ. More formally, we define $\varphi\,\mathsf{U}_i\,\psi := \varphi\,\mathsf{U}_{p^{-1}(i)}\,\psi$.

Since the set $\mathcal{C} = \{p^{-1}(i) \mid i \in \mathcal{P}\}$ is a covering of Σ by cliques of (Σ, D), a reformulation of Theorem 16 yields

Theorem 18. *Let $\mathcal{P}$ be a finite set of processes and $p : \Sigma \to 2^{\mathcal{P}} \setminus \{\emptyset\}$ be a location map. The process-based local logic* $\mathrm{LocTL}(\mathsf{X}_i, \mathsf{U}_i, (\mathsf{X}_a \leq \mathsf{X}_b))$ *based on the modalities X_i and U_i for $i \in \mathcal{P}$ and using only constants $(\mathsf{X}_a \leq \mathsf{X}_b)$ with $p(a) \cap p(b) = \emptyset$ is expressively complete.*

References

1. B. Adsul and M. Sohoni. Complete and tractable local linear time temporal logics over traces. In *Proc. of ICALP'02*, number 2380 in LNCS, pages 926–937. Springer Verlag, 2002.
2. R. Alur, D. Peled, and W. Penczek. Model-checking of causality properties. In *Proc. of LICS'95*, pages 90–100, 1995.
3. 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.
4. V. Diekert. A pure future local temporal logic beyond cograph-monoids. In M. Ito, editor, *Proc. of the RIMS Symposium on Algebraic Systems, Formal Languages and Conventional and Unconventional Computation Theory, Kyoto, Japan 2002*, 2002.
5. V. Diekert and P. Gastin. LTL is expressively complete for Mazurkiewicz traces. In *Proc. of ICALP'00*, number 1853 in LNCS, pages 211–222. Springer Verlag, 2000.
6. V. Diekert and P. Gastin. Local temporal logic is expressively complete for cograph dependence alphabets. In *Proc. of LPAR'01*, number 2250 in LNAI, pages 55–69. Springer Verlag, 2001.
7. V. Diekert and P. Gastin. LTL is expressively complete for Mazurkiewicz traces. *Journal of Computer and System Sciences*, 64:396–418, 2002.
8. V. Diekert and G. Rozenberg, editors. *The Book of Traces*. World Scientific, Singapore, 1995.
9. W. Ebinger. *Charakterisierung von Sprachklassen unendlicher Spuren durch Logiken.* Dissertation, Institut für Informatik, Universität Stuttgart, 1994.
10. W. Ebinger and A. Muscholl. Logical definability on infinite traces. *Theoretical Computer Science*, 154:67–84, 1996.
11. D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Proc. of PoPL'80*, pages 163–173, Las Vegas, Nev., 1980.
12. P. Gastin and D. Kuske. Satisfiability and model checking for MSO-definable temporal logics are in PSPACE. In *Proc. of CONCUR'03*, number 2761 in LNCS, pages 222–236. Springer Verlag, 2003.
13. P. Gastin and M. Mukund. An elementary expressively complete temporal logic for Mazurkiewicz traces. In *Proc. of ICALP'02*, number 2380 in LNCS, pages 938–949. Springer Verlag, 2002.

14. P. Gastin, M. Mukund, and K. Narayan Kumar. Local LTL with past constants is expressively complete for Mazurkiewicz traces. In *Proc. of MFCS'03*, number 2747 in LNCS, pages 429–438. Springer Verlag, 2003.
15. J.A.W. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, California, 1968.
16. A. Mazurkiewicz. Concurrent program schemes and their interpretations. DAIMI Rep. PB 78, Aarhus University, Aarhus, 1977.
17. R. McNaughton and S. Papert. *Counter-Free Automata*. MIT Press, 1971.
18. M. Mukund and P.S. Thiagarajan. Linear time temporal logics over Mazurkiewicz traces. In *Proc. of MFCS'96*, number 1113 in LNCS, pages 62–92. Springer Verlag, 1996.
19. P. Niebert. A ν-calculus with local views for sequential agents. In *Proc. of MFCS'95*, number 969 in LNCS, pages 563–573. Springer Verlag, 1995.
20. W. Penczek. Temporal logics for trace systems: On automated verification. *International Journal of Foundations of Computer Science*, 4:31–67, 1993.
21. D. Perrin. Recent results on automata and infinite words. In *Proc. of MFCS'84*, number 176 in LNCS, pages 134–148. Springer Verlag, 1984.
22. D. Perrin and J.-E. Pin. First order logic and star-free sets. *Journal of Computer and System Sciences*, 32:393–406, 1986.
23. R. Ramanujam. Locally linear time temporal logic. In *Proc. of LICS'96*, pages 118–128, 1996.
24. M.-P. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8:190–194, 1965.
25. P.S. Thiagarajan. A trace based extension of linear time temporal logic. In *Proc. of LICS'94*, pages 438–447. IEEE Computer Society Press, 1994.
26. P.S. Thiagarajan and I. Walukiewicz. An expressively complete linear time temporal logic for Mazurkiewicz traces. In *Proc. of LICS'97*, pages 183–194, 1997.
27. W. Thomas. Star-free regular sets of ω-sequences. *Information and Control*, 42:148–156, 1979.
28. I. Walukiewicz. Difficult configurations – on the complexity of LTrL. In *Proc. of ICALP'98*, number 1443 in LNCS, pages 140–151. Springer Verlag, 1998.