Local safety and local liveness for distributed systems*

Volker Diekert[1] and Paul Gastin[2]

[1] FMI, Universität Stuttgart, Universitätsstraße 38, D-70569 Stuttgart
Volker.Diekert@fmi.uni-stuttgart.de
[2] LSV, ENS Cachan, CNRS, France
Paul.Gastin@lsv.ens-cachan.fr

Abstract. We introduce local safety and local liveness for distributed systems whose executions are modeled by Mazurkiewicz traces. We characterize local safety by local closure and local liveness by local density. Restricting to first-order definable properties, we prove a decomposition theorem in the spirit of the separation theorem for linear temporal logic. We then characterize local safety and local liveness by means of canonical local temporal logic formulae.
Keywords: Local temporal logics, safety, liveness, Mazurkiewicz traces, concurrency.

1 Introduction

Distributed systems are widely used nowadays in almost all application fields of computer science such as telecomunication systems or embedded systems. Since most of these are safety critical systems it is important to develop theory and tools to formally specify and verify them.

Abstract models of distributed systems such as Petri nets have been introduced. Concurrent executions of such systems are naturally described with partial orders such as Mazurkiewicz traces [20, 21] or event structures [30] and Thiagarajan with other authors [17, 25–27] described the relationships between Petri nets and Mazurkiewicz traces or event structures.

When it comes to specification languages, temporal logics are amongst the best formalisms both because they are intuitive and enjoy good algorithmic properties [19]. It turns out that any property can be written as a conjunction of some *safety* and some *liveness* properties [4]. For sequential systems, we can characterize safety and liveness by topological closure and density as well as by canonical temporal logic formulae [1, 2, 4].

When dealing with distributed systems, temporal logics should be extended in order to specify properties of partial orders instead of linear orders. Several temporal logics were introduced [18, 23] but with a global semantics and an existential until these logics are undecidable. One of the first decidable temporal

* This work has been partially supported by projects ANR-06-SETIN-003 DOTS, and P2R MODISTE-COVER/RNP Timed-DISCOVERI.

logic for traces was introduced by Ebinger [11] in his PhD-Thesis and another one (TrPTL) was introduced by Mukund and Thiagarajan [22]. Characterizing the expressive power of temporal logics for traces was a major open problem for several years until Thiagarajan with Walukiewicz showed that the global temporal logic LTrL with universal semantics and some past constants has the same expressive power as first-order logic [28]. Unfortunately, the satisfiability problem for this logic is non-elementary [29]. The expressivity result was later improved with algebraic techniques by showing that the pure future global temporal logic is also expressively complete for first-order logic [7]. This opened the way to the characterization of safety and liveness properties for trace languages with global temporal logic [8].

The global semantics means that we are interested in properties a system may satisfy at global configurations corresponding to possible snapshots or global views. Another interesting approach is to look at local configurations, which describe what a local process may know about the current state of the system. A global configuration correspond to some finite partial order possibly having several maximal events whereas local configurations only have a single maximal event, and can thus be identified with events of the partial order describing an execution. Several local temporal logics for traces were introduced and studied [3, 6]. A major achievement was to establish that the simplest pure future local temporal logic is expressively complete for first-order logic [9]. Contrary to global temporal logics for which the satisfiability problem is either undecidable or non-elementary, local temporal logics enjoy much better algorithmic properties since both satisfiability and model checking are decidable in PSPACE [14, 15].

in the present paper, we introduce and study local safety and local liveness for distributed systems. Intuitively, a system is *locally safe* if all local configurations it can reach are "good" configurations. We characterize local safety by *local closure*. A local configuration is *locally live* with respect to some property if it can be extended to a distributed execusion which meets the property. Local liveness properties (those for which all local configurations are locally live) are characterized by *local density*. In order to obtain local temporal logic characterizations, we restrict to first-order definable properties. Building on the expressive completeness of local temporal logic for traces [9], one of our main result is a *decomposition* theorem for all first-order properties in the spirit of the separation theorem of [13] for linear temporal logic. We also generalize this separation theorem to local temporal logic over traces. Using our decomposition theorem, we are able to characterize local safety and local liveness by canonical local temporal logic formulae. We also introduce a stronger notion of local liveness and give a characterization with special local decomposition formulae.

The paper is organized as follows. In the first section we give some general remarks, next we recall basic definitions on Mazurkiewicz traces. In Section 4 we define local temporal logic and recall the main result of [9] on which this work is based. Our local decomposition theorem and the generalization of the separation theorem to local temporal logic are established in Section 5. The last

two sections are devoted to the definitions and characterizations of local safety and local liveness respectively.

2 General remarks

Before we dive into our specific setting, let us try to explain some basic ideas from a general viewpoint. Some background is useful to understand the following lines. But the reader is free to skip this section since its aim is only to put our work in perspective and it is not needed to understand the rest or the paper.

If X is a topological space, then the topology can be defined in terms of the closure operator $L \mapsto \overline{L}$. Clearly, for $L \subseteq X$ we have $L = \overline{L} \cap (L \cup (X \setminus \overline{L}))$. Hence, in a topological space, every set is the intersection of a closed set and a dense set, because $\overline{L}$ is closed by definition and $L \cup (X \setminus \overline{L})$ is dense, because its closure contains at least $\overline{L} \cup (X \setminus \overline{L})$, hence its closure is X.

Now, if $\mathcal{C}$ is a family of subsets of X which forms a Boolean algebra and which contains $\overline{L}$ for every $L \in \mathcal{C}$, then, trivially, every $L \in \mathcal{C}$ can be written as an intersection $L = A \cap B$ where $A, B \in \mathcal{C}$ and A is closed and B is dense.

In the setting of infinite words and temporal logic, X is the (compact) Cantor space Σ^ω and $\mathcal{C}$ is the set of LTL-definable languages. Closed LTL-definable languages are called *safety properties* and dense LTL-definable languages are called *liveness properties*. Safety properties play an important role in applications. In some sense they are simpler to handle, e.g., they can be recognized by deterministic Büchi automata. We note that actually we could replace $\mathcal{C}$ by other varieties like FO^2-definable languages and we would have a similar statement.

Now, everything transfers smoothly to Mazurkiewicz traces, provided we use the Scott topology which is defined by saying that infinite traces are (very) close, if they agree on (very) long finite prefixes. This means however that we work with a global semantics where we have control over prefixes. This is not natural in the setting of Mazurkiewicz traces, because events are ordered by the information flow and the model should reflect this. As a consequence in a purely distributed setting we do not know the global state and therefore we cannot control global prefexes. This is one of the reasons to favor a local temporal logic. Another one is that these logics allow much better complexities for model checking. From an abstract viewpoint this means that we have to switch from a topology setting to a domain theoretical setting. The *local closure* of a language L now includes all traces where every finite prefix which has a single maximal vertex is a prefix of some element in L. By the very definition every locally closed language is closed in the Scott topology, but the converse does not hold. Consider $\{a^\infty\}$ and $\{b^\infty\}$. Both sets are locally closed, but the union is not, as soon as a and b are independent. Thus, a *local safety property* is a stronger condition than a *global safety property* and cannot be investigated from a purely topological viewpoint. Things become more complicate and we need some careful analysis. On the other hand, knowing that a language satisfies a *local safety property* makes a distributed model checking possible. Our results show that, indeed, local safety is a robust and natural concept.

3 Mazurkiewicz traces

We recall some standard notations from trace theory which will be used in the paper. The reader is refered to [10] for more details.

A *dependence alphabet* is a pair (Σ, D) where the alphabet Σ is a finite set and the *dependence relation* $D \subseteq \Sigma \times \Sigma$ is reflexive and symmetric. The *independence relation* I is the complement of D. For $a \in \Sigma$, we let $I(a) = \{b \in \Sigma \mid (a,b) \in I\}$ be the set of letters independent from a.

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 $x, y \in V$ we have that $(\lambda(x), \lambda(y)) \in D$ implies $x \leq y$ or $y \leq x$, and that $x \lessdot y$ 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 $\uparrow x = \{y \in V \mid x \leq y\}$.

The trace t is finite if V is finite and we denote the set of finite traces by $\mathbb{M}(\Sigma, D)$ (or simply $\mathbb{M}$). By $\mathbb{R}(\Sigma, D)$ (or simply $\mathbb{R}$), we denote the set of finite or infinite traces (also called *real traces*). We write $\mathrm{alph}(t) = \lambda(V)$ for the alphabet of t and we let $\mathrm{alphinf}(t) = \{a \in \Sigma \mid \lambda^{-1}(a) \text{ is infinite}\}$ be the set of letters occurring infinitely often in t.

We define the concatenation for traces $t_1 = [V_1, \leq_1, \lambda_1]$ and $t_2 = [V_2, \leq_2, \lambda_2]$, provided $\mathrm{alphinf}(t_1) \times \mathrm{alph}(t_2) \subseteq I$. It is given by $t_1 \cdot t_2 = [V, \leq, \lambda]$ where V is the disjoint union of V_1 and V_2, $\lambda = \lambda_1 \cup \lambda_2$, 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 $K, L \subseteq \mathbb{R}$ is $K \cdot L = \{r \cdot s \mid r \in K, s \in L \text{ and } \mathrm{alphinf}(r) \times \mathrm{alph}(s) \subseteq I\}$.

Let $r, t \in \mathbb{R}$ be traces. We say that r is a *prefix* of t and we write $r \leq t$ if $t = r \cdot s$ for some $s \in \mathbb{R}$. Prefixes of $t = [V, \leq, \lambda]$ can be identified with downward closed subsets $U = \downarrow U$ of V. We denote by $\mathrm{Pref}(t)$ the set of *finite prefixes* of t. This notation is extended to languages in the obvious way. The set $\mathbb{R}$ endowed with the prefix partial order relation is a coherently complete domain. In particular, any real trace t is the *least upper bound* of its finite prefixes: $t = \sqcup \mathrm{Pref}(t)$.

A trace p is called *prime*, if it is finite and has a unique maximal element. The maximal event of a prime trace t is denoted $\max(t)$. The set of all prime traces in $\mathbb{R}$ is denoted by $\mathbb{P}$. The set of *prime prefixes* of a trace t is denoted $\mathbb{P}\mathrm{pref}(t) = \mathrm{Pref}(t) \cap \mathbb{P}$. Prime prefixes of $t = [V, \leq, \lambda]$ can be identified with downward closures $\downarrow x$ of events $x \in V$. Any real trace t is the *least upper bound* of its prime prefixes: $t = \sqcup \mathbb{P}\mathrm{pref}(t)$.

4 Local temporal logic

Our characterization of local safety and local liveness will be in terms of temporal logic formulae under a *local* semantics. Contrary to the *global* semantics which defines when a formula holds at some *global configuration* in some trace, the

local semantics characterizes the *local events* in a trace satisfying some temporal formulae. In this section, we recall the definition and semantics of local temporal logic over traces, together with the main expressivity result from [9] which is needed for our characterizations of local safety and local liveness.

The syntax of local temporal logic $\mathrm{LocTL}_\Sigma[\mathsf{EX}, \mathsf{U}, \mathsf{EY}, \mathsf{S}]$ is given by

$$\varphi ::= \top \mid a \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{EX}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \mid \mathsf{EY}\,\varphi \mid \varphi\,\mathsf{S}\,\varphi$$

where a ranges over Σ and $\top$ denotes *true*. Here EX denotes the usual (existential) *next*-operator and U means *until*. Their past versions are EY and S meaning *Yesterday* and *Since*.

Formally, the locally defined semantics of $\mathrm{LocTL}_\Sigma[\mathsf{EX}, \mathsf{U}, \mathsf{EY}, \mathsf{S}]$ is given as follows. Let $t = [V, \leq, \lambda] \in \mathbb{R} \setminus \{1\}$ be a real trace and $x \in V$ be a *local event* (we also write $x \in t$ instead of $x \in V$). Then we define:

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

As ususal, we use $\mathsf{F}\,\varphi$ as an abbreviation for $\top\,\mathsf{U}\,\varphi$ and $\mathsf{G}\,\varphi = \neg\,\mathsf{F}\,\neg\varphi$. The meaning of $\mathsf{F}\,\varphi$ is that somewhere in the future φ holds, whereas $\mathsf{G}\,\varphi$ means that always in the future φ holds. By $\bot$ we mean $\neg\top$, which denotes *false*.

Henceforth formulae in $\mathrm{LocTL}_\Sigma[\mathsf{EY}, \mathsf{S}]$ using only the past modalities are called *past formulae* whereas we refer to formulae in $\mathrm{LocTL}_\Sigma[\mathsf{EX}, \mathsf{U}]$ using only future modalities as *future formulae*. It is easy to see by structural induction that if φ is a past formula then for all $t \in \mathbb{R}$ and $x \in t$ we have $t, x \models \varphi$ if and only if $\downarrow x, x \models \varphi$. Similarly, if φ is a future formula then for all $t \in \mathbb{R}$ and $x \in t$ we have $t, x \models \varphi$ if and only if $\uparrow x, x \models \varphi$.

Formulae of the form $\mathsf{F}\,\varphi$ and $\mathsf{G}\,\varphi$ with $\varphi \in \mathrm{LocTL}_\Sigma[\mathsf{EX}, \mathsf{U}, \mathsf{EY}, \mathsf{S}]$ are called F and G formulae respectively. A formula in $\mathrm{LocTL}_\Sigma[\mathsf{EX}, \mathsf{U}, \mathsf{EY}, \mathsf{S}]$ can be viewed, by definition, as a first-order formula in one free variable. However, for F and G formulae, it is also natural to give a direct interpretation on traces. Let $t \in \mathbb{R}$, we define:

$$\begin{array}{lll}
t \models_\ell \mathsf{F}\,\varphi & \text{if} & \exists x \in t,\ t, x \models \varphi \\
t \models_\ell \mathsf{G}\,\psi & \text{if} & \forall x \in t,\ t, x \models \psi.
\end{array}$$

More generally, if γ is any Boolean combination of F and G formulae, then we extend the meaning of $t \models_\ell \gamma$ in the obvious way; and this defines a language $\mathcal{L}(\gamma) = \{t \in \mathbb{R} \mid t \models_\ell \gamma\}$. Note that the empty trace $1 \models \mathsf{G}\,\varphi$ but $1 \not\models \mathsf{F}\,\varphi$ for all $\varphi \in \mathrm{LocTL}_\Sigma$.

Our results are consequences of the following theorem.

Theorem 1 ([9]). *Let $L \subseteq \mathbb{R}$ be a first-order definable real trace language. Then there is a future formula $\varphi \in \mathrm{LocTL}_\Sigma[\mathsf{EX}, \mathsf{U}]$ such that*

$$L \cap \mathbb{R}^1 = \{t \in \mathbb{R}^1 \mid t, \min(t) \models \varphi\}.$$

By duality, Theorem 1 implies the following corollary where we let $\mathbb{R}^1$ be the set of real traces t with exactly one minimal event, denoted $\min(t)$.

Corollary 2. *Let $L \subseteq \mathbb{R}$ be a first-order definable real trace language. Then there is a past formula $\psi \in \mathrm{LocTL}_\Sigma[\mathsf{EY}, \mathsf{S}]$ such that*

$$L \cap \mathbb{P} = \{t \in \mathbb{P} \mid t, \max(t) \models \psi\}.$$

Proof. It is clear that $\mathbb{P}$ is first-order definable. Hence, if L is first-order definable, then so is $L \cap \mathbb{P}$. Now, we *reverse* all traces, i.e., we read traces from right-to-left (formally, we replace $\leq$ by $\geq$). For the reverse language of $L \cap \mathbb{P}$ (which is a first-order definable subset in $\mathbb{R}^1 \cap \mathbb{M}$) we obtain a future formula $\varphi \in \mathrm{LocTL}_\Sigma[\mathsf{EX}, \mathsf{U}]$ by Theorem 1. We obtain ψ by replacing in φ all occurrences of EX by EY and all occurrences of U by S. □

5 Local decomposition of first-order languages

In this section, we establish a *decomposition* theorem based on the local semantics of temporal logic. This decomposition theorem is in the spirit of the *separation* theorem for temporal logic over words [13]. Since the proof technique is quite similar, we also extend this *separation* theorem to the local semantics of temporal logic over traces.

Given a trace $t = [V, \leq, \lambda] \in \mathbb{R}$ and a vertex $x \in V$, we are interested in the sets of vertices that are strictly below x or strictly above x or concurrent to x:

$$\begin{aligned} \Downarrow x &= \{y \in V \mid y < x\} \\ \Uparrow x &= \{y \in V \mid x < y\} \\ \| x &= \{y \in V \mid x \not\leq y \text{ and } y \not\leq x\}. \end{aligned}$$

By slight abuse of notation, we also denote by $\Downarrow x$, $\Uparrow x$ and $\| x$ the corresponding factors of t. We have a canonical decomposition of t, depicted in Figure 1:

$$\begin{aligned} V &= \Downarrow x \cup \{x\} \cup \| x \cup \Uparrow x \\ t &= (\Downarrow x) \cdot \lambda(x) \cdot (\| x) \cdot (\Uparrow x) \end{aligned}$$

Let us introduce a new local modality CO which talks about the part $\| x$ of the trace which is concurrent to the current vertex x. By a *concurrent formula* we mean a formula of type $\mathsf{CO}\,\gamma$, where γ is any Boolean combination of F and G formulae. The semantics of a concurrent formula is given by

$$t, x \models \mathsf{CO}\,\gamma \quad \text{if} \quad \| x \models_\ell \gamma.$$

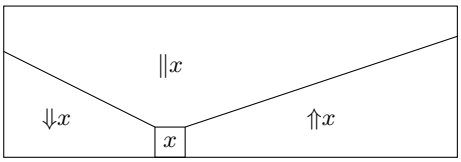

Fig. 1. Canonical decomposition of a trace

The main result of this section associates with any first-order definable language some *decomposition formula*. A decomposition formula is a disjunction

$$\delta = \bigvee_{j \in J} a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$$

where J is some finite index set, and for each $j \in J$, $a_j \in \Sigma$ is a letter, $\psi_j \in \mathrm{LocTL}_\Sigma(\mathsf{EY}, \mathsf{S})$ is a past formula, $\varphi_j \in \mathrm{LocTL}_\Sigma(\mathsf{EX}, \mathsf{U})$ is a future formula, and γ_j is an F or G formula. Note that, if $J = \emptyset$ then we get $\delta = \bot$ by convention. We can now state the *decompotion theorem*.

Theorem 3. *Let $L \subseteq \mathbb{R}$ be a first-order definable real trace language. Then there exists a* decomposition formula $\delta = \bigvee_{j \in J} a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$ *such that*

(i) $L \cup \{1\} = \mathcal{L}(\mathsf{G}\,\delta)$,
(ii) $L \setminus \{1\} = \mathcal{L}(\mathsf{F}\,\delta)$,
(iii) $\mathbb{P}\mathrm{ref}(L) = \{r \in \mathbb{P} \mid r, \max(r) \models \bigvee_{j \in J} a_j \wedge \psi_j\}$,
(iv) for each $j \in J$, the formula $a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$ is satisfiable.

Proof. The proof is by induction on the size of the alphabet. If $\Sigma = \emptyset$ then we let $J = \emptyset$ so that $\delta = \bot$. We have $\mathcal{L}(\mathsf{F}\,\delta) = \emptyset$ and $\mathcal{L}(\mathsf{G}\,\delta) = \{1\}$. Statements (i-iv) are satisfied since either $L = \emptyset$ or $L = \{1\}$ and in any cases $\mathbb{P}\mathrm{ref}(L) = \emptyset$.

Assume now that $\Sigma \neq \emptyset$. Since L is first-order, it is also aperiodic [12]. Let $h : \mathbb{M}(\Sigma, D) \to S$ be a morphism to some finite aperiodic monoid S which recognizes[3] L. Without loss of generality, we assume that h is *alphabetic*, i.e., $h(r) = h(s)$ implies $\mathrm{alph}(r) = \mathrm{alph}(s)$ for all $r, s \in \mathbb{M}(\Sigma, D)$.

Let $t \in L$ be a nonempty trace and let $x \in t$. The h-equivalence classes $[\Downarrow x]$, $[\Uparrow x]$ and $[\|x]$ are recognized by h, hence are first-order definable trace languages. Note also that all vertices in $\|x$ are labeled with letters independent from $\lambda(x)$. Since h is alphabetic, we deduce that the trace language $[\|x]$ is over a *strict* sub-alphabet of Σ.

We define the index set

$$J = \{(\lambda(x), [\Downarrow x], [\|x], [\Uparrow x]) \mid t \in L \setminus \{1\} \text{ and } x \in t\}$$

[3] The reader is refered to [24] for recognizability by morphisms. Here we will only use the fact that the morphism h induces an equivalence relation on $\mathbb{R}$ with finitely many classes and which saturates L ($t \in L$ implies $[t] \subseteq L$) and satisfies $[r] \cdot [s] \subseteq [r \cdot s]$ whenever the product $r \cdot s$ is defined. Moreover, each h-equivalence class is first-order definable.

which is finite since there are finitely many h-equivalence classes. Fix now some index $j = (a_j, L_p^j, L_c^j, L_f^j) \in J$. By Theorem 1 and Corollary 2 we find a future formula φ_j and a past formula ψ_j such that

$$a_j \cdot L_f^j \cap \mathbb{R}^1 = \{s \in \mathbb{R}^1 \mid s, \min(s) \models \varphi_j\}$$
$$L_p^j \cdot a_j \cap \mathbb{P} = \{r \in \mathbb{P} \mid r, \max(r) \models \psi_j\}.$$

Now, L_c^j is over a strict sub-alphabet of Σ and by induction we find a decomposition formula δ_j for this language. Depending on whether $1 \in_c^j L$ or not we let $\gamma_j = \mathsf{G}\,\delta_j$ or $\gamma_j = \mathsf{F}\,\delta_j$ so that $L_c^j = \mathcal{L}(\gamma_j)$. We claim that the decomposition formula $\delta = \bigvee_{j \in J} a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$ satisfies statements (i-iv).

First, let $t \in L \setminus \{1\}$ and let $x \in t$. Fix the index $j = (a_j, [\Downarrow x], [\|x], [\Uparrow x]) \in J$ with $a_j = \lambda(x)$. We have $\downarrow x = \Downarrow x \cdot a_j$ and $\uparrow x = a_j \cdot \Uparrow x$, hence by definition of φ_j and ψ_j we get $\downarrow x, x \models \psi_j$ and $\uparrow x, x \models \varphi_j$ and $\|x \models \gamma_j$. Since ψ_j is a past formula and φ_j is a future formula, we deduce that $t, x \models a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$, which is therefore a satisfiable formula. We have also shown $t \models \mathsf{G}\,\delta$ and $t \models \mathsf{F}\,\delta$ for all $t \in L \setminus \{1\}$ and that $r, \max(r) \models \bigvee_{j \in J} a_j \wedge \psi_j$ for all $r \in \mathbb{P}\mathrm{ref}(L)$. Hence, we have proved statement (iv) and that the left hand side is contained in the right hand side for statements (i-iii).

Conversely, assume that $t \models \mathsf{F}\,\delta$ or that $t \neq 1$ and $t \models \mathsf{G}\,\delta$. Let $x \in t$ and $j \in J$ be such that $t, x \models a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$. For clarity, we write below r, s and u for the factors $\Downarrow x$, $\Uparrow x$ and $\|x$ of t. Since ψ_j is a past formula and φ_j is a future formula, we deduce that $ra_j, x \models \psi_j$ and $a_j s, x \models \varphi_j$ and $u \models \gamma_j$ and $\lambda(x) = a_j$. Let $t' \in L \setminus \{1\}$ and $x' \in t'$ be such that $j = (\lambda(x'), [\Downarrow x'], [\|x'], [\Uparrow x'])$. As above, we write r', s' and u' for the factors $\Downarrow x'$, $\Uparrow x'$ and $\|x'$ of t'. By definition of the formulae ψ_j, φ_j and γ_j we have $r \in [r']$, $s \in [s']$ and $u \in [u']$. We deduce that $t = r \cdot a_j \cdot u \cdot s \in [t']$. Since $t' \in L$ and h recognizes L we obtain $t \in L$.

For the converse inclusion of (iii), let $j \in J$ and $ra_j \in \mathbb{P}$ be such that $ra_j, \max(ra_j) \models a_j \wedge \psi_j$. Let $t' \in L \setminus \{1\}$ and $x' \in t'$ be such that $j = (\lambda(x'), [\Downarrow x'], [\|x'], [\Uparrow x'])$. As above, we write r', s' and u' for the factors $\Downarrow x'$, $\Uparrow x'$ and $\|x'$ of t'. We obtain $r \in [r']$. Let $t = r \cdot a_j \cdot u' \cdot s'$. We have $t \in [t']$ and we get $t \in L$. Therefore, $ra_j \in \mathbb{P}\mathrm{ref}(L)$. □

A similar proof allows to generalize the separation theorem from words to traces with the local semantics of temporal logic.

Theorem 4. *Let $\varphi \in \mathrm{FO}_\Sigma(<)$ be a first-order formula with one free variable. Then there exists a* decomposition formula $\delta = \bigvee_{j \in J} a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$ *such that for all $t \in \mathbb{R} \setminus \{1\}$ and all $x \in t$ we have*

$$t \models \varphi(x) \quad \textit{iff} \quad t, x \models \delta.$$

6 Local safety properties

In this section, we investigate local safety for distributed systems. Intuitively, a *safety property* is defined by a set P of *safe partial executions*. A finite or

infinite execution is *safe* if all its partial executions are in the given set P. The difference between global safety and local safety stems from the semantics of *partial execution*. For local safety, the partial executions of a trace $t \in \mathbb{R}$ are its *prime prefixes* $\mathbb{P}\mathrm{ref}(t)$. The advantage of local safety is that it can be locally enforced by processes of the system: before executing an action, a local process makes sure that the partial execution having this action as maximal event is in the set P of safe partial executions.

Below, we first define local safety. Then we characterize local safety my means of local closure. Next, for local safety properties which are first-order definable, we give a characterization by local temporal logic formulae of the form $\mathsf{G}\,\psi$ where ψ is a past formula.

We say that a trace $t \in \mathbb{R}$ is *locally safe* with respect to some set $P \subseteq \mathbb{P}$ if $\mathbb{P}\mathrm{ref}(t) \subseteq P$. A trace language $L \subseteq \mathbb{R}$ is said to be a *local safety property*, if we can find some set $P \subseteq \mathbb{P}$ such that

$$L = \{t \in \mathbb{R} \mid \mathbb{P}\mathrm{ref}(t) \subseteq P\}.$$

Example 5. For instance, if $\Sigma = \{a, b, c\}$ and $I = \{(a, b), (b, a)\}$ then the language L of traces $t \in \mathbb{R}$ such that $t = ucrcscv$ with $|r|_c = |s|_c = 0$ implies $|r|_a + |r|_b$ is different from $|s|_a + |s|_b$ modulo 2 is a local safety property. In order to ensure locally this property, the process executing a counts the number of a's modulo 2 since the last c and similarly for the process executing b. Now, whenever a c is about to occur, these numbers are checked by the process in charge of c and depending on whether the safety property is satisfied or not, the action c is enabled or not.

A subset $K \subseteq \mathbb{R}$ is called *coherent* if for all $r, s \in K$ there is some $t \in \mathbb{R}$ such that r and s are prefixes of t. Since $\mathbb{R}$ is coherently complete [16], the least upper bound of any coherent set exists. Hence, $K \subseteq \mathbb{R}$ is coherent if and only if it is bounded from above, i.e., if all elements of K are prefixes of some $t \in \mathbb{R}$. In particular, $\mathbb{P}\mathrm{ref}(t)$ is coherent for all $t \in \mathbb{R}$.

We say that a trace language L is *locally closed* if it is closed under prime prefixes and under least upper bounds of coherent subsets: $\mathbb{P}\mathrm{ref}(L) \subseteq L$ and $\sqcup K \in L$ for any coherent set $K \subseteq L$. Note that, if L is locally closed, then it is also closed under prefixes: it $s \leq t \in L$ then $\mathbb{P}\mathrm{ref}(s) \subseteq \mathbb{P}\mathrm{ref}(t) \subseteq L$ and we get $s = \sqcup\mathbb{P}\mathrm{ref}(s) \in L$.

The *local closure* $\overline{L}^{\ell}$ of a language $L \subseteq \mathbb{R}$ is the smallest set which is locally closed and contains L. Note that $1 = \sqcup\emptyset \in \overline{L}^{\ell}$ for all $L \subseteq \mathbb{R}$. We have

$$\overline{L}^{\ell} = \{t \in \mathbb{R} \mid \mathbb{P}\mathrm{ref}(t) \subseteq \mathbb{P}\mathrm{ref}(L)\}.$$

Indeed, we have $L \subseteq L' = \{t \in \mathbb{R} \mid \mathbb{P}\mathrm{ref}(t) \subseteq \mathbb{P}\mathrm{ref}(L)\}$. Next, L' is locally closed since $s \leq t$ implies $\mathbb{P}\mathrm{ref}(s) \subseteq \mathbb{P}\mathrm{ref}(t)$ and $t = \sqcup X$ for some $X \subseteq L'$ implies $\mathbb{P}\mathrm{ref}(t) = \mathbb{P}\mathrm{ref}(X) \subseteq \mathbb{P}\mathrm{ref}(L)$. Finally, assume that $L \subseteq K$ for some locally closed set K. Let $t \in L'$. We get $\mathbb{P}\mathrm{ref}(t) \subseteq \mathbb{P}\mathrm{ref}(L) \subseteq \mathbb{P}\mathrm{ref}(K) \subseteq K$ and $t = \sqcup\mathbb{P}\mathrm{ref}(t) \in K$ since $\mathbb{P}\mathrm{ref}(t)$ is coherent. Therefore, $L' \subseteq K$.

Proposition 6. *A trace language $L \subseteq \mathbb{R}$ is a local safety property if and only if it is locally closed.*

Proof. Assume that $L = \{t \in \mathbb{R} \mid \mathbb{P}\mathrm{ref}(t) \subseteq P\}$ for some $P \subseteq \mathbb{P}$. Let $s \leq t \in L$. We have $\mathbb{P}\mathrm{ref}(s) \subseteq \mathbb{P}\mathrm{ref}(t) \subseteq P$ hence $s \in L$. Let now $t = \sqcup K$ for some $K \subseteq L$ coherent. We have $\mathbb{P}\mathrm{ref}(t) = \mathbb{P}\mathrm{ref}(K) \subseteq \mathbb{P}\mathrm{ref}(L) \subseteq P$. Hence $t \in L$.

Conversely, assume that L is locally closed. Then, $L = \overline{L}^{\ell}$ is a local safety property defined by $\mathbb{P}\mathrm{ref}(L)$. □

We turn now to the characterization of local safety by means of temporal logic formulae. A formula in $\mathrm{LocTL}_{\Sigma}[\mathsf{EX}, \mathsf{U}, \mathsf{EY}, \mathsf{S}]$ is called a *canonical local safety formula* if it can be written as $\mathsf{G}\,\psi$ where $\psi \in \mathrm{LocTL}_{\Sigma}[\mathsf{EY}, \mathsf{S}]$ is a past formula. We show that local safety properties which are first order definable can be characterized by canonical local safety formulae.

Theorem 7. *A first-order definable real trace language is a local safety property if and only if it can be expressed by a canonical local safety formula. More precisely:*

(i) Each language defined by a canonical local safety formula is locally closed.
(ii) The local closure of a first-order definable language can be expressed by a canonical local safety formula.

Proof. (i) Let $L = \mathcal{L}(\mathsf{G}\,\psi)$ where ψ is a past formula. Let $t \in \mathbb{R}$ with $\mathbb{P}\mathrm{ref}(t) \subseteq \mathbb{P}\mathrm{ref}(L)$. For all $x \in t$ we have $r = {\downarrow}x \in \mathbb{P}\mathrm{ref}(t)$ hence we find $s \in L$ such that $r \in \mathbb{P}\mathrm{ref}(s)$. Since ψ is a past formula, we have $t, x \models \psi$ if and only if $r, x \models \psi$ if and only if $s, x \models \psi$, which holds since $s \in L$. Therefore, $t \models \mathsf{G}\,\psi$ and $t \in L$. Therefore, $L = \overline{L}^{\ell}$ is locally closed.

(ii) Let $L \subseteq \mathbb{R}$ be a first-order definable language. Consider a decomposition formula $\delta = \bigvee_{j \in J} a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$ for L as given by Theorem 3 and let $\psi = \bigvee_{j \in J} a_j \wedge \psi_j$. For all $r \in \mathbb{P}$, we have $r \in \mathbb{P}\mathrm{ref}(L)$ if and only if $r, \max(r) \models \psi$. We show that $\overline{L}^{\ell} = \mathcal{L}(\mathsf{G}\,\psi)$.

First, let $t \in \mathcal{L}(\mathsf{G}\,\psi) \setminus \{1\}$. Let $r = {\downarrow}x$ with $x \in t$ be a prime prefix of t. We have $t, x \models \psi$ and since ψ is a past formula we deduce $r, \max(r) \models \psi$ and then $r \in \mathbb{P}\mathrm{ref}(L)$. Therefore, $\mathbb{P}\mathrm{ref}(t) \subseteq \mathbb{P}\mathrm{ref}(L)$ and we obtain $t \in \overline{L}^{\ell}$.

Conversely, let $t \in \mathbb{R}$ with $\mathbb{P}\mathrm{ref}(t) \subseteq \mathbb{P}\mathrm{ref}(L)$. For all $r \in \mathbb{P}\mathrm{ref}(t)$, we get $r, \max(r) \models \psi$ by Theorem 3(iii). Since ψ is a past formula, we deduce that $t, x \models \psi$ for all $x \in t$. Therefore, $t \in \mathcal{L}(\mathsf{G}\,\psi)$. □

Example 8. The language L defined in Example 5 is a local safety property but is not first-order definable. On the other hand, with the same dependence alphabet, the language L' of traces $t \in \mathbb{R}$ such that $t = ucrcv$ with $|r|_c = 0$ implies $|r|_a \leq 2$ and $|r|_b \leq 2$ is a local safety property which is first-order definable. It is defined by the canonical local safety formula

$$\mathsf{G}\big(c \wedge \mathsf{EY}(\top\,\mathsf{S}\,c) \longrightarrow \neg\,\mathsf{EY}(a \wedge \mathsf{EY}(a \wedge \mathsf{EY}\,a)) \wedge \neg\,\mathsf{EY}(b \wedge \mathsf{EY}(b \wedge \mathsf{EY}\,b))\big).$$

We conclude this section by a comparison between global safety and local safety. With the global semantics, a partial execution is an arbitrary prefix, not necessarily a prime. Hence, a trace $t \in \mathbb{R}$ is *globally safe* with respect to some set $M \subseteq \mathbb{M}$ if $\mathrm{Pref}(t) \subseteq M$. Moreover, a language $L \subseteq \mathbb{R}$ is a *global safety property* if $L = \{t \in \mathbb{R} \mid \mathrm{Pref}(t) \subseteq M\}$ for some set $M \subseteq \mathbb{M}$. It was shown in [8] that a language $L \subseteq \mathbb{R}$ is a global safety property if and only if it is *Scott closed*, i.e., closed under prefixes and under least upper bounds of directed sets. The Scott closure of a set $L \subseteq \mathbb{R}$ is $\overline{L}^{\sigma} = \{t \in \mathbb{R} \mid \mathrm{Pref}(t) \subseteq \mathrm{Pref}(L)\}$. It follows immediately that $\overline{L}^{\sigma} \subseteq \overline{L}^{\ell}$ and if L is locally closed then it is also Scott closed. Therefore, any local safety property is also a global safety property.

The complement of Scott closed sets are called Scott open sets and they form a topology. In particular, the union of two Scott closed sets is also Scott closed. But a union of two locally closed sets needs not be locally closed. Indeed, let $\Sigma = \{a, b\}$ with $(a, b) \in I$ and consider the set $L = a^{\infty} \cup b^{\infty}$. This set is Scott-closed, but its local closure is $\mathbb{R}$. Therefore the global safety property $a^{\infty} \cup b^{\infty}$ is not necessarily a local safety property: locally safety is a stronger requirement than global safety.

As another example, let $\Sigma = \{a, b, c\}$ with $I = \{(a, b), (b, a)\}$ and consider the set L of traces $t \in \mathbb{R}$ such that $t = ucrcv$ with $|r|_c = 0$ implies $|r|_a + |r|_b \leq 3$. Then L is a first-order definable global safety property but not a local safety property. In order to enforce the global safety property L we need a synchronization between the processes executing a and b, which is not possible in a distributed system since a and b are independent. In other words, any asynchronous (cellular) automaton [5, 31] for L has *deadlocks*: it is not possible to have a *safe (without deadlocks)* distributed implementation for L. On the other hand, any first-order definable local safety property can be implemented by a deterministic asynchronous cellular automaton without deadlocks.

7 Local liveness property

We turn now our attention to local liveness for distributed systems. A partial execution is *live* with respect to a set L of desired behaviors if it can be extended to some element in L. Again, the difference between local liveness and global liveness stems from the semantics of *partial execution*. In our local paradigm, a partial execution is a prime trace r and it is live with respect to L if $r \in \mathbb{P}\mathrm{ref}(L)$.

Below, we formally define local liveness and we show that it is characterized by local density. We establish a natural characterization by local temporal logic formulae using the decomposition formulae introduced in Section 5. Then we define and characterize *strong local liveness* where the possibility to *extend* a partial execution to a desired one is strongly limited.

A language $L \subseteq \mathbb{R}$ is a *local liveness property* if each partial execution can be extended to some trace in L, i.e., if $\mathbb{P}\mathrm{ref}(L) = \mathbb{P}$.

With the global semantics, a partial execution is simply a finite trace and not necessarily a prime. Hence a language $L \subseteq \mathbb{R}$ is a *global liveness property* if $\mathbb{M} = \mathrm{Pref}(L)$. Global liveness implies local liveness since $\mathbb{P}\mathrm{ref}(L) = \mathrm{Pref}(L) \cap \mathbb{P}$.

Example 9. Let $\Sigma = \{a, b\}$ with $(a, b) \in I$. Then the language $L = \{a^\omega, b^\omega\}$ is a local liveness property since we have $\mathbb{P} = a^+ \cup b^+ = \mathbb{P}\mathrm{ref}(L)$. But L is not a global liveness property since $\mathrm{Pref}(L) = \mathbb{P}\mathrm{ref}(L) \neq \mathbb{M}$. On the other hand, the language $L = \{(ab)^\omega\}$ is a global liveness property, hence also a local liveness property.

Let us call a language $L \subseteq \mathbb{R}$ *locally dense* if all traces are in the local closure of L, i.e., if $\overline{L}^\ell = \mathbb{R}$.

Proposition 10. *A trace language $L \subseteq \mathbb{R}$ is a local liveness property if and only if it is locally dense.*

Proof. Assume first that L is a local liveness property and let $t \in \mathbb{R}$. We have $\mathbb{P}\mathrm{ref}(t) \subseteq \mathbb{P} = \mathbb{P}\mathrm{ref}(L)$, hence $t \in \overline{L}^\ell$. Conversely, assume that L is locally dense. Let $r \in \mathbb{P} \subseteq \overline{L}^\ell$. We have $r \in \mathbb{P}\mathrm{ref}(r) \subseteq \mathbb{P}\mathrm{ref}(L)$. Hence $\mathbb{P} = \mathbb{P}\mathrm{ref}(L)$. □

It follows by some purely formal argument that every language $L \subseteq \mathbb{R}$ is the intersection of a locally closed language and a locally dense one. Indeed, $\overline{L}^\ell$ is locally closed, $L \cup \mathbb{R} \setminus \overline{L}^\ell$ is locally dense and we have

$$L = \overline{L}^\ell \cap (L \cup \mathbb{R} \setminus \overline{L}^\ell).$$

We deduce that every trace language is the intersection of a *local safety property* with a *local liveness property*, which extends a classical result on words.

We turn now to the characterization of first-order definable local liveness properties by means of local temporal logic formulae. This will be based on the decomposition formulae introduced in Section 5.

Consider a first-order language $L \subseteq \mathbb{R}$ and a decomposition formula

$$\delta = \bigvee_{j \in J} a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\, \gamma_j$$

for L as given by Theorem 3. In particular, with $\psi = \bigvee_{j \in J} a_j \wedge \psi_j$ we have

$$\mathbb{P}\mathrm{ref}(L) = \{r \in \mathbb{P} \mid r, \max(r) \models \psi\}$$

and we deduce that a partial execution $r \in \mathbb{P}$ is *live* with respect to L if $r, \max(r) \models \psi$. Note that this can be checked *locally* by a deterministic distributed automaton. More precisely, there is a deterministic asynchronous cellular automaton [5, 31] such that whenever executing an event e labeled a, the local process in charge of a knows whether the partial execution $r = \downarrow e$ with e as maximal event is live, i.e., satisfies the past formula ψ.

The second consequence is that the language L is a local liveness property if and only if the formula ψ is *valid*, i.e., if for all non empty trace $t \in \mathbb{R}$ and all $x \in t$ we have $t, x \models \psi$. Indeed, if ψ is valid then $r, \max(r) \models \psi$ for all $r \in \mathbb{P}$ and we deduce that $\mathbb{P}\mathrm{ref}(L) = \mathbb{P}$. Conversely, if L is a local liveness property then

for all $t \in \mathbb{R} \setminus \{1\}$ and all $x \in t$ we must have $\downarrow x, x \models \psi$ since $\downarrow x$ is prime. We deduce that $t, x \models \psi$ since ψ is a past formula.

This motivates the following definition. A *canonical local liveness formula* is an F formula $\mathsf{F}\,\delta$ where

$$\delta = \bigvee_{j \in J} a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$$

is a decompotion formula such that $\psi = \bigvee_{j \in J} a_j \wedge \psi_j$ is *valid* and $a_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$ is satisfiable for all $j \in J$.

Proposition 11. *Let $\mathsf{F}\,\delta$ be a canonical local liveness formula. Then the language $L = \mathcal{L}(\mathsf{F}\,\delta)$ is a local liveness property.*

Proof. We use the notations above for δ and ψ. Let $r \in \mathbb{P}$. Since ψ is valid we have $r, \max(r) \models a_j \wedge \psi_j$ for some $j \in J$. Now, $a_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$ is satisfiable and we find $t \in \mathbb{R} \setminus \{1\}$ and $x \in t$ such that $t, x \models a_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$. For clarity, we write s and u for the factors of t corresponding to $\|x$ and $\Uparrow x$. Let $t' = r \cdot s \cdot u$ and $y = \max(r) \in t'$. We know that $s \models \mathsf{CO}\,\gamma_j$ and $s = \|y$ in t'. Since φ_j is a future formula we deduce from $t, x \models a_j \wedge \varphi_j$ that $a_j u, y \models a_j \wedge \varphi_j$ and also $t', y \models a_j \wedge \varphi_j$. Finally, ψ_j being a past formula we get $t', y \models \psi_j$. Therefore, $t', y \models a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$ and we obtain $t' \in L = \mathcal{L}(\mathsf{F}\,\delta)$. Hence, $r \in \mathbb{P}\mathrm{ref}(L)$ and we have shown that $\mathbb{P} = \mathbb{P}\mathrm{ref}(L)$ as desired. □

We have already explained above that, conversely, any first-order definable local liveness property $L \subseteq \mathbb{R} \setminus \{1\}$ can be described by a canonical local liveness formula. The following theorem is more precise.

Theorem 12. *Let $L \subseteq \mathbb{R}$ be a first-order definable real trace language and let*

$$\delta = \bigvee_{j \in J} a_j \wedge \psi_j \wedge \varphi_j \wedge \mathsf{CO}\,\gamma_j$$

be a decomposition formula for L given by Theorem 3. Let $\psi = \bigvee_{j \in J} a_j \wedge \psi_j$. Then we have

(i) $\overline{L}^{\ell} = \mathcal{L}(\mathsf{G}\,\psi)$.

(ii) If L is a local liveness property, then ψ is valid and $L \setminus \{1\} = \mathcal{L}(\mathsf{F}\,\delta)$ is defined by a canonical local liveness formula.

(iii) $\mathsf{F}(\neg\psi \vee \delta)$ is a canonical local liveness formula. Hence $\widetilde{L} = \mathcal{L}(\mathsf{F}(\neg\psi \vee \delta))$ is a local liveness property. Moreover, $\widetilde{L} = (L \setminus \{1\}) \cup (\mathbb{R} \setminus \overline{L}^{\ell})$ and $\widetilde{L}$ is the largest set K such that $L \setminus \{1\} = \overline{L}^{\ell} \cap K$.

Proof. (i) This was already shown in the proof of Theorem 7(ii).

(ii) We have seen above that if L is a local liveness property then ψ is valid. We know that $L \setminus \{1\} = \mathcal{L}(\mathsf{F}\,\delta)$ by Theorem 3(ii).

(iii) Note that $\neg\psi = \bigvee_{a \in \Sigma} a \wedge \neg\psi \wedge \top \wedge \mathsf{CO}\,\mathsf{G}\,\top$, therefore, $\delta' = \neg\psi \vee \delta$ is a decomposition formula. Also, $\neg\psi \vee \psi$ is valid, hence $\mathsf{F}\,\delta'$ is a canonical local liveness formula. We deduce that $\widetilde{L}$ is a local liveness property by Proposition 11.

By Theorem 3 we have $\mathcal{L}(\mathsf{F}\,\delta) = L \setminus \{1\}$. Next, $\mathcal{L}(\mathsf{F}\,\neg\psi) = \mathbb{R} \setminus \overline{L}^{\ell}$ by (i). Therefore, $\widetilde{L} = \mathcal{L}(\mathsf{F}(\delta \vee \neg\psi)) = (L \setminus \{1\}) \cup (\mathbb{R} \setminus \overline{L}^{\ell})$. It follows that $\widetilde{L} \cap \overline{L}^{\ell} = L \setminus \{1\}$. Conversely, if $\overline{L}^{\ell} \cap K = L \setminus \{1\}$ then $K \subseteq (L \setminus \{1\}) \cup (\mathbb{R} \setminus \overline{L}^{\ell}) = \widetilde{L}$. □

We introduce now a stronger notion of local liveness. We first motivate why a stronger notion may be interesting. If L is a local liveness property then each partial execution which is prime, i.e., corresponding to the local view of some process, may be extended to a behavior in L. But based on its local view a process does not know whether the current global execution can be extended to some behavior in L. For instance, if $\Sigma = \{a, b\}$ with $(a, b) \in I$, the language $L = \{a^\omega, b^\omega\}$ is a local liveness property. Assume that we have two processes, one executing a and the other executing b. After the execution of ab the local views of the two processes are a and b respectively. Based on its local view, each process may think that the current execution is live although it is not possible to extend it to some trace in L.

Alternatively, we may think that a computation is locally live with respect to some language L if each process has the possibility to locally initiate a computation reaching the language L whatever the current local states of concurrent processes are. This is a much stronger notion of local liveness that can be formalized as follows.

A language $L \subseteq \mathbb{R}$ is a *strong local liveness property* (SLLP) if it is a local liveness property (LLP) such that for all $t = raus \in \mathbb{R} \setminus \{1\}$ with $ra \in \mathbb{P}$, $a \in \Sigma$, $as \in \mathbb{R}^1$ and $\mathrm{alph}(u) \subseteq I(a)$ we have $raus \in L$ if and only if $ras \in L$. Intuitively, ra is the local view of some process, as is the computation that this process may initiate in order to reach L and $raus$ is a possible resulting behavior including u which is the part executed independently by the other processes. The additional condition makes sure that reaching L does not depend on what the concurrent processes may have already performed.

Note that if L is a strong local liveness property then it is also a global liveness property (GLP). Indeed, any nonempty finite trace may be written rau with $ra \in \mathbb{P}$, $a \in \Sigma$ and $\mathrm{alph}(u) \subseteq I(a)$. Since L is a local liveness property, we find $vs \in \mathbb{R}$ such that $ravs \in L$, $\mathrm{alph}(v) \subseteq I(a)$ and $as \in \mathbb{R}^1$. Since L is a SLLP we deduce that $ras \in L$ and then $raus \in L$. Therefore, any finite trace may be extended to some trace in L which is the definition of a GLP. But not all GLP are SLLP. Consider again $\Sigma = \{a, b\}$ with $(a, b) \in I$. If L is a SLLP over this dependence alphabet then $L \cap a^\infty \neq \emptyset$. Indeed, $a \in \mathbb{P}$ and L is a LLP hence we find $us \in \mathbb{R}$ with $aus \in L$, $as \in \mathbb{R}^1$ and $\mathrm{alph}(u) \subseteq I(a)$. Since L is a SLLP we deduce that $as \in L$. But since $as \in \mathbb{R}^1$ we get $\mathrm{alph}(as) = \{a\}$. Therefore, the singleton $\{(ab)^\omega\}$ is not a SLLP although it is a GLP. Recall also that any GLP is also a LLP but not all LLP are GLP:

$$\mathrm{SLLP} \subsetneq \mathrm{GLP} \subsetneq \mathrm{LLP}.$$

We conclude by giving a temporal logic characterization of first-order definable SLLP.

Theorem 13. *A language $L \subseteq \mathbb{R}$ is a first-order definable strong local liveness property if and only if there is a decomposition formula*

$$\delta = \bigvee_{j \in I} a_j \wedge \psi_j \wedge \varphi_j$$

with J finite, $\psi = \bigvee_{j \in J} a_j \wedge \psi_j$ valid, and for all $j \in J$, $a_j \in \Sigma$ is a letter, $\psi_j \in \mathrm{LocTL}_\Sigma(\mathsf{EY}, \mathsf{S})$ is a past formula, $\varphi_j \in \mathrm{LocTL}_\Sigma(\mathsf{EX}, \mathsf{U})$ is a future formula, $a_j \wedge \psi_j \wedge \varphi_j$ is satisfiable and such that

$$L \setminus \{1\} = \mathcal{L}(\mathsf{F}\,\delta) \qquad \text{and} \qquad L \cup \{1\} = \mathcal{L}(\mathsf{G}\,\delta).$$

Proof. Assume first that L is a first-order definable SLLP. The proof is similar to that of Theorem 3. Let $h : \mathbb{M}(\Sigma, D) \to S$ be a morphism to some finite aperiodic monoid S which recognizes L. We define the index set

$$J = \{(a, [r], [s]) \mid ras \in L \text{ with } ra \in \mathbb{P}, a \in \Sigma \text{ and } as \in \mathbb{R}^1\}$$

which is finite since there are finitely many h-equivalence classes. Fix now some index $j = (a_j, L_p^j, L_f^j) \in J$. By Theorem 1 and Corollary 2 we find a future formula φ_j and a past formula ψ_j such that

$$a_j \cdot L_f^j \cap \mathbb{R}^1 = \{s \in \mathbb{R}^1 \mid s, \min(s) \models \varphi_j\}$$
$$L_p^j \cdot a_j \cap \mathbb{P} = \{r \in \mathbb{P} \mid r, \max(r) \models \psi_j\}.$$

We claim that the decomposition formula $\delta = \bigvee_{j \in J} a_j \wedge \psi_j \wedge \varphi_j$ satisfies the requirements.

By definition of J each formula $a_j \wedge \psi_j \wedge \varphi_j$ is satisfiable. Let now $t \in \mathbb{R} \setminus \{1\}$ and $x \in t$. We write r for the factor $\Downarrow x$ in t and we let a be the label of x. Then $ra \in \mathbb{P}$. Since L is a LLP we find $us \in \mathbb{R}$ such that $raus \in L$ with $\mathrm{alph}(u) \subseteq I(a)$ and $as \in \mathbb{R}^1$. We deduce that $ras \in L$ since this language is a SLLP. With $j = (a, [r], [s]) \in J$ we obtain $ra, \max(ra) \models a_j \wedge \psi_j$ and since ψ_j is a past formula it follows $t, x \models a_j \wedge \psi_j$. Therefore, ψ is valid.

Next, let $t \in L \setminus \{1\}$ and let $x \in t$. We write r, u and s the factors of t corresponding to $\Downarrow x$, $\| x$ and $\Uparrow x$ and let a be the label of x. We have $ra \in \mathbb{P}$, $\mathrm{alph}(u) \subseteq I(a)$ and $as \in \mathbb{R}^1$. Since L is a SLLP and $t = raus \in L$ we obtain $ras \in L$. Hence we can consider the index $j = (a, [r], [s]) \in J$. It is easy to check that $raus, x \models a_j \wedge \psi_j \wedge \varphi_j$. We deduce that $t \models \mathsf{G}\,\delta$ and $t \models \mathsf{F}\,\delta$. Therefore, $L \setminus \{1\} \subseteq \mathcal{L}(\mathsf{F}\,\delta)$ and $L \cup \{1\} \subseteq \mathcal{L}(\mathsf{G}\,\delta)$.

Conversely, assume that $t \models \mathsf{F}\,\delta$ or that $t \neq 1$ and $t \models \mathsf{G}\,\delta$. Let $x \in t$ and $j \in J$ be such that $t, x \models a_j \wedge \psi_j \wedge \varphi_j$. We write r, u and s for the factors $\Downarrow x$, $\| x$ and $\Uparrow x$ of t. Since ψ_j is a past formula and φ_j is a future formula, we deduce that $ra_j, x \models \psi_j$ and $a_j s, x \models \varphi_j$ and $\lambda(x) = a_j$. Let $r'as' \in L$ with $r'a \in \mathbb{P}$, $a = a_j$ and $as' \in \mathbb{R}^1$ be such that $j = (a, [r'], [s'])$. By definition of the formulae ψ_j and φ_j we have $r \in [r']$ and $s \in [s']$. We deduce that $ras \in [r'as']$ and since $r'as' \in L$ and h recognizes L we obtain $ras \in L$. Finally, since L is a SLLP we get $t = raus \in L$. Therefore, $L \setminus \{1\} \supseteq \mathcal{L}(\mathsf{F}\,\delta)$ and $L \cup \{1\} \supseteq \mathcal{L}(\mathsf{G}\,\delta)$.

We turn now to the proof of the "if" part of Theorem 13. So let δ be a decomposition formula with the notations and properties stated in the theorem. Since $\top = \mathsf{CO}\,\mathsf{G}\,\top$, the formula $\mathsf{F}\,\delta$ is a canonical local liveness formula and we deduce from Proposition 11 that $L \setminus \{1\} = \mathcal{L}(\mathsf{F}\,\delta)$ is a LLP. Hence, L is also a LLP.

Let now $t = raus \in \mathbb{R} \setminus \{1\}$ with $ra \in \mathbb{P}$, $a \in \Sigma$, $as \in \mathbb{R}^1$ and $\mathrm{alph}(u) \subseteq I(a)$. For all $j \in J$ we have $raus, \max(ra) \models a_j \wedge \psi_j \wedge \varphi_j$ if and only if $ras, \max(ra) \models a_j \wedge \psi_j \wedge \varphi_j$ since ψ_j is a past formula and φ_j is a future formula. Therefore, $raus \in L \subseteq \mathcal{L}(\mathsf{G}\,\delta)$ implies $ras \in \mathcal{L}(\mathsf{F}\,\delta) \subseteq L$ and conversely, $ras \in L \subseteq \mathcal{L}(\mathsf{G}\,\delta)$ implies $raus \in \mathcal{L}(\mathsf{F}\,\delta) \subseteq L$. □

Remark 14. We have seen two notions of local liveness: LLP and SLLP. We have chosen LLP as the standard notion since it is equivalent with local density. Note that, if we wish that every language is the intersection of a local safety property and a local liveness property then each locally dense language must be called locally live. Indeed, assume that $L = K_1 \cap K_2$ where L is locally dense and K_1 is a local safety property. Then $\mathbb{R} = \overline{L}^{\ell} \subseteq \overline{K_1}^{\ell} = K_1$ and we deduce $K_2 = L$.

References

1. B. Alpern and F.B. Schneider. Defining liveness. *Information Processing Letters*, 21:181–185, 1985.
2. B. Alpern and F.B. Schneider. Recognizing safety and liveness. *Distributed Computing*, 2:117–126, 1987.
3. R. Alur, D. Peled, and W. Penczek. Model-checking of causality properties. In *Proc. of LICS'95*, pages 90–100. IEEE Computer Society Press, 1995.
4. E. Chang, Z. Manna, and A. Pnueli. Characterization of temporal property classes. In W. Kuich, editor, *Proc. of ICALP'92*, number 623 in LNCS, pages 474–486. Springer Verlag, 1992.
5. R. Cori, Y. Métivier, and W. Zielonka. Asynchronous mappings and asynchronous cellular automata. *Information and Computation*, 106:159–202, 1993.
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. A preliminary version appeared at ICALP'00, LNCS 1853, pages 211-222, Springer Verlag.
8. V. Diekert and P. Gastin. Safety and liveness properties for real traces and a direct translation from LTL to monoids. In *Formal and Natural Computing — Essays Dedicated to Grzegorz Rozenberg*, number 2300 in LNCS, pages 26–38. Springer Verlag, 2002.
9. V. Diekert and P. Gastin. Pure future local temporal logics are expressively complete for Mazurkiewicz traces. *Information and Computation*, 204:1597–1619, 2006. A preliminary version appeared at LATIN'04, LNCS 2976, pages 232–241, Springer Verlag.
10. V. Diekert and G. Rozenberg, editors. *The Book of Traces*. World Scientific, Singapore, 1995.

11. W. Ebinger. *Charakterisierung von Sprachklassen unendlicher Spuren durch Logiken*. Dissertation, Institut für Informatik, Universität Stuttgart, 1994.
12. W. Ebinger and A. Muscholl. Logical definability on infinite traces. *Theoretical Computer Science*, 154:67–84, 1996.
13. D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Proc. of PoPL'80*, pages 163–173, 1980.
14. 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.
15. P. Gastin and D. Kuske. Uniform satisfiability in PSPACE for local temporal logics over Mazurkiewicz traces. *Fundamenta Informaticae*, 80(1-3):169–197, November 2007. A preliminary version appeared at CONCUR'05, LNCS 3653, pages 533–547, Springer Verlag.
16. P. Gastin and A. Petit. Infinite traces. In V. Diekert and G. Rozenberg, editors, *The Book of Traces*, chapter 11, pages 393–486. World Scientific, Singapore, 1995.
17. P.W. Hoogers, H.C.M. Kleijn, and P.S. Thiagarajan. A trace semantics for petri nets. In W. Kuich, editor, *Proc. of ICALP'92*, number 623 in LNCS, pages 595–604. Springer Verlag, 1992.
18. S. Katz and D. Peled. Interleaving set temporal logic. *Theoretical Computer Science*, 75:21–43, 1991.
19. Z. Manna and A. Pnueli. *The temporal logic of reactive and concurent systems: Specification*. Springer Verlag, 1992.
20. A. Mazurkiewicz. Concurrent program schemes and their interpretations. DAIMI Rep. PB 78, Aarhus University, Aarhus, 1977.
21. A. Mazurkiewicz. Traces, histories, graphs: Instances of a process monoid. In M.P. Chytil et al., editors, *Proc. of MFCS'84*, number 176 in LNCS, pages 115–133. Springer Verlag, 1984.
22. 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.
23. W. Penczek. On undecidability of temporal logics on trace systems. *Information Processing Letters*, 43:147–153, 1992.
24. D. Perrin and J.-E. Pin. *Infinite words*, volume 141 of *Pure and Applied Mathematics*. Elsevier, 2004.
25. G. Rozenberg and P.S. Thiagarajan. Petri nets: Basic notions, structure and behaviour. In *Current Trends in Concurrency*, number 224 in LNCS, pages 585–668. Springer Verlag, 1986.
26. B. Rozoy and P.S. Thiagarajan. Event structures and trace monoids. *Theoretical Computer Science*, 91(2):285–313, 1991.
27. P.S. Thiagarajan. Elementary net systems. In W. Brauer, editor, *Petri nets: central models and their properties; advances in Petri nets Vol. 1*, number 254 in LNCS, pages 26–59. Springer Verlag, 1986.
28. 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.
29. I. Walukiewicz. Difficult configurations – on the complexity of LTrL. In *Proc. of ICALP'98*, number 1443 in LNCS, pages 140–151. Springer Verlag, 1998.
30. G. Winskel. Event structures. In W. Brauer, editor, *Petri nets: central models and their properties; advances in Petri nets Vol. 2*, number 255 in LNCS, pages 325–392. Springer Verlag, 1986.
31. W. Zielonka. Notes on finite asynchronous automata. *R.A.I.R.O. — Informatique Théorique et Applications*, 21:99–135, 1987.