Toward a Structure Theory of Regular Infinitary Trace Languages[⋆]

Namit Chaturvedi[⋆⋆]

RWTH Aachen University, Lehrstuhl für Informatik 7, D-52056 Aachen
`chaturvedi@automata.rwth-aachen.de`

Abstract. The family of regular languages of infinite words is structured into a hierarchy where each level is characterized by a class of deterministic ω-automata – the class of deterministic Büchi automata being the most prominent among them. In this paper, we analyze the situation of regular languages of infinite Mazurkiewicz traces that model non-terminating, concurrent behaviors of distributed systems. Here, a corresponding classification is still missing. We introduce the model of "synchronization-aware asynchronous automata", which allows us to initiate a classification of regular infinitary trace languages in a form that is in nice correspondence to the case of ω-regular word languages.

1 Introduction

In the theory of ω-regular word languages, a natural classification is induced by various forms of deterministic ω-automata. The three fundamental cases are given by (a) deterministic Muller automata, capturing the class of ω-regular word languages; (b) deterministic Büchi automata, capturing recurrence properties of infinite words; and (c) weak automata, capturing reachability properties of infinite words. In this paper, we concentrate on the first two automata models, on which fundamental facts can be summarized as follows (see e.g. [8]):

1. A language is deterministically Büchi recognizable if and only if it can be expressed as $\mathsf{lim}(K) := \{\alpha \in \Sigma^\omega \mid \alpha \text{ has infinitely many prefixes in } K\}$ for some regular language $K \subseteq \Sigma^*$.
2. An ω-regular language is deterministically Büchi recognizable if and only if this language is recognized by a Muller automaton whose acceptance component is closed under supersets.
3. The class of Boolean combinations of deterministically Büchi recognizable languages coincides with the class of Muller recognizable languages.

[⋆] This is a preliminary version of the paper that appears in ICALP 2014: the 41st International Colloquium on Automata, Languages, and Programming.

[⋆⋆] Supported in part by the DFG Research Training Group-1298 AlgoSyn, and in part by the CASSTING Project funded by the European Commission's 7th Research Framework Programme.

We consider the question of defining corresponding classes in the framework of Mazurkiewicz traces [4] that model infinite, concurrent behaviors of a finite sets of interacting processes. The concept of "ω-regular trace language" can be introduced in close correspondence to the case of ω-regular word languages, for example, in terms of finite partially-commutative monoids, asynchronous automata, concurrent regular expressions, or MSO logic (cf. [4]).

However, it is remarkable that there does not yet exist a definition of Büchi automaton over traces that allows for results analogous to items 1–3 above. The objective of the present paper is to fill this gap.

Muscholl [7] took a major step toward establishing such structural results by introducing a parameterized lim operator for trace languages. She showed that the class of Boolean combinations of parameterized lim-languages is precisely the class of ω-regular trace languages, and also characterized the class of linearizations of these parameterized languages in terms of "I-diamond" Büchi (word) automata with "extended" acceptance condition. The respective family of I-diamond automata characterizing Boolean combinations of linearizations of reachability languages (where an infinite trace is in the language if it contains a certain finite prefix) is studied in [2]. However, I-diamond word automata do not offer a proper modeling of concurrency as realized over traces.

We introduce a new concept of asynchronous automata, viz. *synchronization-aware asynchronous automata* (over traces rather than their linearizations). These, when equipped with Büchi and Muller acceptance conditions, establish not only item 1, but also items 2 and 3 above. At the same time, the synchronization-aware Muller automata are equivalent in expressive power to the standard deterministic asynchronous Muller automata for infinitary trace languages. Thus we provide a new framework that prepares – at least in important parts – a structure theory for ω-regular trace languages that is compatible with that of deterministic ω-automata over words.

Synchronization-aware automata are "aware" of the fact that during a run over an infinite trace, the set of processes may be partitioned in a manner that each part is minimal and, after a finite prefix, a process belonging to one part never interacts directly or indirectly with a process belonging to another part. The processes infer this partition by observing their infinitely recurring interactions. Although infinite traces induce such partitions in all asynchronous automata, current models cannot perform such inferencing.

Another aspect of infinite runs is that while some processes may remain *live* ad infinitum, others may halt after finitely many steps. However, the set of live processes can be explicitly coded in the Büchi acceptance condition since this directly corresponds to Muscholl's parameterized lim operation mentioned above.

By combining both these aspects, we obtain the family of synchronization-aware Büchi automata corresponding to item 1 above (see Thm. 13). We also introduce synchronization-aware Muller automata recognizing precisely the class of ω-regular trace languages (see Thm. 18). Finally, Theorems 20 and 21 respectively demonstrate a characterization à la item 2 and the equivalence result of item 3. We conclude with a discussion of a number of open problems.

2 Preliminaries

2.1 Finite and Infinite Traces

Over a finite alphabet Σ, let $D \subseteq \Sigma^2$ be a binary, reflexive, and symmetric *dependence relation*. We also refer to the corresponding *independence relation* $I = \Sigma^2 \setminus D$, and to the *independence alphabet* (Σ, I). Given an independence alphabet, a *finite trace* is an isomorphism class of directed acyclic graphs $t = [V, \lessdot, \lambda]$ where V is a finite set of events; $\lambda\colon V \to \Sigma$ is a labeling function; and for events $e, e' \in V\colon \lambda(e) D \lambda(e') \Leftrightarrow e \lessdot e'$ or $e' \lessdot e$ or $e = e'$. The *concatenation* of two finite traces $t_1 = [V_1, \lessdot_1, \lambda_1]$ and $t_2 = [V_2, \lessdot_2, \lambda_2]$ is given by $t_1 \odot t_2 = [V_1 \uplus V_2, \lessdot', \lambda_1 \uplus \lambda_2]$, where $\lessdot' = \lessdot_1 \uplus \lessdot_2 \uplus \{(e_1, e_2) \in V_1 \times V_2 \mid \lambda_1(e_1) D \lambda_2(e_2)\}$. We denote the set of all finite traces over an alphabet (Σ, I) with $\mathbb{M}(\Sigma, I)$.

For convenience, we work with "simplified" traces $t = [V, \lessdot, \lambda]$ where we remove all edges that may be inferred from others, i.e. by $\lessdot$ we mean $\lessdot \setminus \lessdot^2$ (see Fig. 1a). We also refer to the partial order $<$ obtained from the transitive closure of this edge relation; and define relations $\leq$, $\gtrdot$, $\geq$, and $>$ in the natural manner. We use the abbreviation $e \in t$ to convey $t = [V, \lessdot, \lambda]$ and $e \in V$.

An *infinite trace* is a directed acyclic graph $\theta = [V, \lessdot, \lambda]$ where V is a countable set of events, and λ and $\lessdot$ are like above except $\lessdot$ satisfies an additional requirement, namely, for each $e \in \theta$, the set $\{e' \in \theta \mid e' \leq e\}$ is finite. Denote the set of all infinite traces with $\mathbb{R}(\Sigma, I)$. For traces $t \in \mathbb{M}(\Sigma, I), \theta \in \mathbb{R}(\Sigma, I)$, we refer to sets $\mathsf{alph}(t), \mathsf{alph}(\theta)$ of letters occurring in them, and to the set $\mathsf{alphinf}(\theta)$ of letters occurring infinitely often in θ.

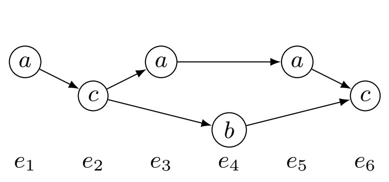
We say t_1 is a *prefix* of t_2, i.e. $t_1 \sqsubseteq t_2 :\Leftrightarrow \exists t' : t_2 = t_1 \odot t'$, and $t_1 \sqsubset t_2$ iff $t_1 \sqsubseteq t_2$ and $t_1 \neq t_2$. We also refer to prefixes t of some $\theta \in \mathbb{R}(\Sigma, I)$ in a similar way. If $E \subseteq t$ is a set of events, then $t[E] = [V', \lessdot', \lambda']$ is a prefix of t with the set $V' := \{f \in t \mid f \leq e \text{ for some } e \in E\}$ and $\lessdot'$ and λ' are obtained by restricting the corresponding entities in t to V'. The least upper bound of two traces t_1, t_2, whenever it exists, denoted $t_1 \sqcup t_2$ is the smallest trace s such that $t_1 \sqsubseteq s \wedge t_2 \sqsubseteq s$. Similarly, if it exists, the greatest lower bound of t_1 and t_2, denoted $t_1 \sqcap t_2$, is the largest trace s such that $s \sqsubseteq t_1 \wedge s \sqsubseteq t_2$.

2.2 Asynchronous Transition Systems

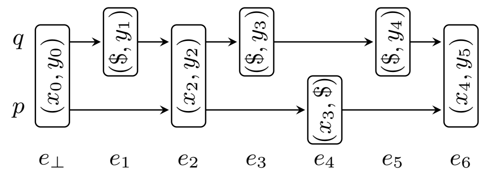
We refer to a deterministic asynchronous automaton as a pair $\mathfrak{A} = (\mathfrak{T}, \mathcal{F})$, where $\mathfrak{T}$ is a deterministic asynchronous transition system and $\mathcal{F}$ is an appropriate acceptance condition. We discuss these components separately.

For a fixed alphabet (Σ, I), an asynchronous transition system consists of a set $\mathcal{P}$ of *processes*, a mapping $\mathsf{dom} : \Sigma \to 2^{\mathcal{P}}$ assigning the *domain* of each letter such that $\bigcup_{a \in \Sigma} \mathsf{dom}(a) = \mathcal{P}$ and $a \; I \; b \Leftrightarrow \mathsf{dom}(a) \cap \mathsf{dom}(b) = \emptyset$. Naturally, for $\Sigma' \subseteq \Sigma$, we also refer to $\mathsf{dom}(\Sigma') := \bigcup_{a \in \Sigma'} \mathsf{dom}(a)$. Moreover for an event $e \in t$, we refer to $\mathsf{dom}(e)$ instead of referring to $\mathsf{dom}(\lambda(e))$. Similarly, for $E \subseteq t$.

Processes p have sets X_p of *local p-states*. Introducing a symbol $\$ \notin \bigcup_{p \in \mathcal{P}} X_p$, for a set $P \subseteq \mathcal{P}$ the set X_P of *P-states* is a defined as $X_P := \{(x_p)_{p \in \mathcal{P}} \mid x_i \in X_{p_i} \text{ if } p_i \in P, \text{ otherwise } x_i = \$\}$. We find it convenient to assume an order over



(a) Trace prefix $t = [V, \lessdot, \lambda]$.



(b) Run $\rho = [V \cup \{e_\perp\}, \lessdot', \lambda', \Lambda]$.

Fig. 1: For $\Sigma = \{a, b, c\},\ a\ I\ b$, a finite trace (prefix) $t \in \mathbb{M}(\Sigma, I)$ and the run ρ of an ATS, with $\mathsf{dom}(a) = \{q\}, \mathsf{dom}(b) = \{p\}$, and $\mathsf{dom}(c) = \{p, q\}$.

$\mathcal{P}$ and view a P-state as a tuple. So we refer to a *state* as a tuple $\pi \in X_P$ for some $P \subseteq \mathcal{P}$. A state is a *global state* if $P = \mathcal{P}$. We always distinguish between a $\{p\}$-state π and a local p-state x; and for a state π, define the *p-state in* π as $\pi_{|p} \coloneqq x_p \in X_p \cup \{\$\}$, and similarly the *$P$-state* $\pi_{|P}$ *in* π. Also, $\mathsf{dom}(\pi) \coloneqq \{p \in \mathcal{P} \mid \pi_{|p} \neq \$\}$. Finally, we denote the set of all states $X_{2^{\mathcal{P}}} \coloneqq \bigcup_{P \subseteq \mathcal{P}} X_P$.

We now define a *deterministic asynchronous transition system* (an *ATS*) as a tuple $\mathfrak{T} = ((X_p)_{p \in \mathcal{P}}, (\delta_a)_{a \in \Sigma}, \pi_0)$, where X_p are sets of local p-states; transition functions $\delta_a \colon X_{\mathsf{dom}(a)} \to X_{\mathsf{dom}(a)}$ define how processes jointly perform state transitions on input letters a; and $\pi_0 \in X_{\mathcal{P}}$ is the global initial state of $\mathfrak{T}$.

Given a trace $t = [V, \lessdot, \lambda] \in \mathbb{M}(\Sigma, I)$, or $\theta = [V, \lessdot, \lambda] \in \mathbb{R}(\Sigma, I)$, we define the corresponding *run* $\rho = [V', \lessdot', \lambda', \Lambda]$ of $\mathfrak{T}$ on the trace where $V' \coloneqq V \cup \{e_\perp\}$ contains a fictional, minimum event $e_\perp$. The relation $\lessdot'$ is identical to the edge relation $\lessdot$, except that $e_\perp$ is the unique minimum event.

During the run ρ of an ATS $\mathfrak{T}$ over a trace, each process p makes state transitions on events $e \in \mathsf{dom}^{-1}(p)$. Each such event may be called a *p-event* as well as a *P-event* where $P = \mathsf{dom}(e)$. All p-events in the run are totally ordered, and this order $<'_p$ can be defined with the help of the order $<$ of the trace. The *maximum p-event in* ρ according to the ordering $<'_p$ is denoted as $\max_p(\rho) \geq e_\perp$. If it exists, the p-predecessor f of an event e is denoted by $f \lessdot'_p e$. The labeling λ' is defined similarly except $\lambda'(e_\perp) \coloneqq \epsilon$; and $\Lambda \colon V' \to X_{2^{\mathcal{P}}}$ is defined inductively:

- $\Lambda(e_\perp) \coloneqq (\pi_0)$,
- for any $e >' e_\perp$, if 1. $a = \lambda(e)$, and 2. for $e_p \lessdot'_p e$, if $x_p = \Lambda(e_p)_{|p}$ are the most recent p-states just before e, then $\Lambda(e) \coloneqq \delta_a((y_p)_{p \in \mathcal{P}})$, where $y_p = x_p$ if $p \in \mathsf{dom}(e)$, $y_p = \$$ otherwise.

Fig. 1 shows the labeled events of a trace and the corresponding run; but λ' is omitted in ρ for readability. The processes are assumed to be lexicographically ordered, hence the representation of states as tuples. Note that, in Fig. 1b, the edges are shown as per the relations $\lessdot'_p, p \in \mathcal{P}$. Importantly, although $e_\perp <' e_2$ and $e_\perp \lessdot'_p e_2$, it is not the case that $e_\perp \lessdot' e_2$.

Analogous to trace prefixes, we refer to run prefixes, and to prefixes $\rho[e], \rho[E]$ for $e \in \rho$ and $E \subseteq \rho$ respectively. For $e \in \rho$, we also refer to the label $\Lambda(e)$ as the *state of* $\mathfrak{T}$ *at* e. Similarly, if ρ is a finite run, then the *state of* $\mathfrak{T}$ *at* ρ is given by $\Lambda(\rho) = (x_p)_{p\in\mathcal{P}}$ where $x_p \coloneqq \Lambda(\max_p(\rho))_{|p}$ is the p-state of $\mathfrak{T}$ at $\max_p(\rho)$; $x_p = \pi_{0|p}$ if $\max_p(\rho) = e_\perp$. Obviously, $\Lambda(\rho)$ is always a global state.

Finally, a *deterministic asynchronous automaton* (a *DAA*) over finite traces is a pair $\mathfrak{A} = (\mathfrak{T}, F)$, where $\mathfrak{T}$ is an ATS and $F \subseteq X_{\mathcal{P}}$ is a set of global states of $\mathfrak{T}$. A finite trace $t \in \mathbb{M}(\Sigma, I)$ is said to be *accepted* by $\mathfrak{A}$ if, for the run ρ of $\mathfrak{T}$ on t, $\Lambda(\rho) \in F$. The set $L(\mathfrak{A}) \subseteq \mathbb{M}(\Sigma, I)$ denotes the set of all finite traces accepted by the DAA $\mathfrak{A}$. A language $T \subseteq \mathbb{M}(\Sigma, I)$ is called *recognizable* or *regular* if there exists a DAA $\mathfrak{A}$ such that $T = L(\mathfrak{A})$.

2.3 Regular Infinitary Languages

The definition of regular languages of infinite traces, *ω-regular trace languages*, was first provided by Gastin-Petit using monoid morphisms [5]. We use as definition, a characterization of the same family in terms of deterministic asynchronous (cellular) Muller automata [3, 7]. The notion of acceptance of an infinite trace $\theta \in \mathbb{R}(\Sigma, I)$ by an ATS $\mathfrak{T}$ is defined by referring to the *local infinity sets* $\mathsf{Inf}_p(\rho)$ of local p-states that occur infinitely often during the run ρ of $\mathfrak{T}$ over θ, with

$$\mathsf{Inf}_p(\rho) \coloneqq \begin{cases} \left\{ x \in X_p \mid \exists^\infty e \in \rho : \Lambda(e)_{|p} = x \right\} & \text{if } p \in \mathsf{dom}(\mathsf{alphinf}(\theta)), \\ \left\{ x \in X_p \,\middle|\, \begin{array}{l} \exists e \in \rho : e = \max_p(\rho) \\ \text{and } \Lambda(e)_{|p} = x \end{array} \right\} & \text{otherwise.} \end{cases}$$

Let $\mathcal{F} = \{F_1, F_2, \dots\}$ be a table where each $F_i = (F_i^p)_{p\in\mathcal{P}}$ is a tuple of sets of local states of the processes. A *deterministic asynchronous Büchi automaton* (a *DABA*) is a pair $\mathfrak{A} = (\mathfrak{T}, \mathcal{F})$. A DABA is said to accept a trace $\theta \in \mathbb{R}(\Sigma, I)$ if, on the run ρ of $\mathfrak{A}$ on θ, there exists a tuple $F_i \in \mathcal{F}$ such that for each process p, $F_i^p \subseteq \mathsf{Inf}_p(\rho)$ [5, 3]. A *deterministic asynchronous Muller automaton* (a *DAMA*) is a pair $\mathfrak{A} = (\mathfrak{T}, \mathcal{F})$, and is said to accept a trace θ if there exists a tuple $F_i \in \mathcal{F}$ such that for each process p, $F_i^p = \mathsf{Inf}_p(\rho)$ [3].

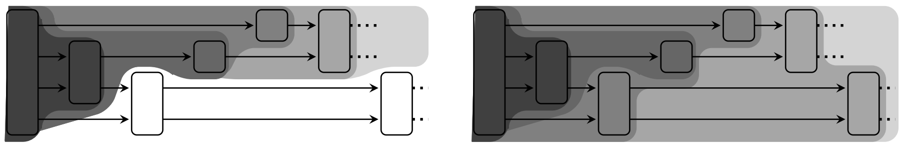
Definition 1. *A language $\Theta \subseteq \mathbb{R}(\Sigma, I)$ is said to be a* regular infinitary language *(or an ω-*regular trace language*) if it is recognized by a DAMA.*

Definition 2 ([3]). *For a language $T \subseteq \mathbb{M}(\Sigma, I)$ finite traces, the* infinitary limit *of T, denoted $\mathsf{lim}(T)$, is the language containing traces $\theta \in \mathbb{R}(\Sigma, I)$ such that there exists a sequence $(t_i)_{i\in\mathbb{N}}, t_i \in T$ satisfying $t_i \sqsubset t_{i+1}$ and $\bigsqcup_{i\in\mathbb{N}} t_i = \theta$.*

Fig. 2 illustrates the definition of $\mathsf{lim}(T)$ with the help of an infinite run of an asynchronous automaton recognizing T. Fig. 2a illustrates an induced run if the trace $\theta \notin \mathsf{lim}(T)$, whereas Fig. 2b illustrates the contrary.

Muscholl studies infinitary limits that are parameterized by a set of letters. This set governs which letters from the alphabet must occur infinitely often in the traces, and which letters may not.

Definition 3 ([7]). *For $T \subseteq \mathbb{M}(\Sigma, I)$ and some $A \subseteq \Sigma$, the A-*infinitary limit *of T is defined as $\mathsf{lim}_A(T) \coloneqq \{\theta \in \mathsf{lim}(T) \mid D(\mathsf{alphinf}(\theta)) = D(A)\}$.*



(a) $t_i \in T$ and $t_i \sqsubset t_{i+1}$, but $\theta \notin \mathsf{lim}(T)$ since $\bigsqcup_{i \in \mathbb{N}} t_i \neq \theta$.

(b) $\theta \in \mathsf{lim}(T)$ since each event is eventually covered by an accepting prefix.

Fig. 2: Illustrating Def. 2. Shaded regions constitute sequences of accepting runs.

Definition 4 ([7]). *An ω-regular trace language is called a* deterministic trace language *if it can be expressed as a finite union $\bigcup_i \mathsf{lim}_{A_i}(T_i)$ for regular trace languages $T_i \subseteq \mathbb{M}(\Sigma, I)$ and sets $A_i \subseteq \Sigma$.*

Clearly, the language $\mathsf{lim}(T)$ is a deterministic trace language since $\mathsf{lim}(T) = \bigcup_{A \subseteq \Sigma} \mathsf{lim}_A(T)$. However, not every deterministic trace language can be expressed in the form $\lim(T)$ for any T.

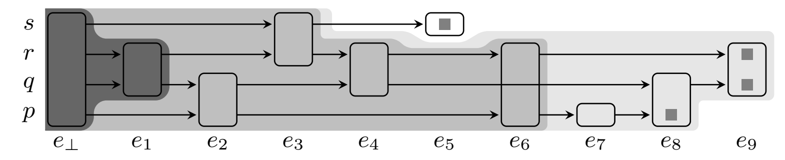
It is still open whether there exists a DABA recognizing the language $\mathsf{lim}(T)$ for any given regular trace language $T \subseteq \mathbb{M}(\Sigma, I)$. Furthermore, there exist deterministic trace languages that are not accepted by any DABA [7]. In this regard the term "deterministic trace language" [7] is not well founded, since it has no equivalent in any of the classes of deterministic asynchronous ω-automata known so far. The results of this paper justify this term by providing a matching class of deterministic, "synchronization-aware" Büchi automata.

2.4 Secondaries and Frontiers

During a run ρ of an ATS, the processes can be thought of as "possessing and updating information" regarding other processes [6]. If ρ is finite and $p, q \in \mathcal{P}$, the *first-hand information* that p has about q at ρ, denoted by $\mathsf{latest}_{p \to q}(\rho)$, is the maximal q-event in the prefix $\rho[\max_p(\rho)]$. Trivially, $\mathsf{latest}_{p \to p}(\rho) = \max_p(\rho)$. Similarly, for $p, q, r \in \mathcal{P}$, the *second-hand information* that p has about r via q at ρ, denoted by $\mathsf{latest}_{p \to q \to r}(\rho)$, is the maximal r-event in the prefix $\rho[\mathsf{latest}_{p \to q}(\rho)]$. Trivially, $\mathsf{latest}_{p \to p \to q}(\rho) = \mathsf{latest}_{p \to q}(\rho)$.

The *primary information* of p at ρ is defined as the ordered set $\mathsf{Pri}_p(\rho) \coloneqq \{\mathsf{latest}_{p \to q}(\rho) \mid q \in \mathcal{P}\}$. The *secondary information* of p at ρ is given by the set $\mathsf{Sec}_p(\rho) \coloneqq \{\mathsf{latest}_{p \to q \to r}(\rho) \mid q, r \in \mathcal{P}\}$. It is easy to see that on the one hand $\mathsf{Pri}_p(\rho) \subseteq \mathsf{Sec}_p(\rho)$, on the other hand the events of $\mathsf{Sec}_p(\rho)$ may be ordered as per the partial order $<$ of ρ. This gives us a view of the *secondary graph* of p at ρ, which we identify with secondary information itself. In this paper, we are mainly interested in secondary information of the form $\mathsf{Sec}_p(\rho[e])$ for $p \in \mathsf{dom}(e)$. Since, $\mathsf{Sec}_p(\rho[e]) = \mathsf{Sec}_q(\rho[e])$ for all $p, q \in \mathsf{dom}(e)$, for convenience we denote this information simply as $\mathsf{Sec}(e)$.

There exists a distributed *gossip algorithm* that enables processes to update their secondary graphs at the points of synchronization (cf. [6]). When processes synchronize at an event e, the gossip algorithm takes the secondary sets



Partial frontiers for ρ: $\{e_5\}$, $\{e_9\}$, $\{e_5, e_9\}$, $\{e_8, e_9\}$, and $\{e_5, e_8, e_9\}$.
At e_4, $\rho_\sqcap = \rho[e_1]$; and at e_9, $\rho_\sqcap = \rho[e_6]$. Note that $e_5 \notin \rho[e_9]$.

Fig. 3: Partial frontiers (see below); and illustration of Lemma 6 (see Ex. 7).

$\mathsf{Sec}(f_p), f_p \lessdot_p e$, for each $p \in \mathsf{dom}(e)$, and outputs the updated secondary set $\mathsf{Sec}(e)$ reflecting the consistent, most recent information available within $\mathsf{dom}(e)$.

While referring to finite runs ρ over finite traces, or over finite prefixes of infinite traces, it is useful to refer to their maximum p-events as a set. Define *frontier of* ρ as $H_\rho \coloneqq \{e \in \rho \mid \exists p \in \mathcal{P}, e = \max_p(\rho)\}$. Any upward closed subset $H \subseteq H_\rho$ is called a *partial frontier*. E.g., the set $\{e_5, e_8\}$ in Fig. 3 is not a partial frontier of ρ since it is not an upward closed subset of the frontier $\{e_5, e_8, e_9\}$.

Finally, for event $e \in \rho$, define the *top of* e *in* ρ as $\top_\rho(e) \coloneqq \{f \in \rho \mid e \leq f \wedge \exists p \in \mathcal{P} \colon f = \max_p(\rho)\}$. Of course for any $e_1, \ldots, e_n \in \rho$, $\bigcup_{i=1}^{n} \top_\rho(e_i)$ is a partial frontier of ρ. If $\Lambda(\rho)$ is the global state of an automaton, and if H is a (partial) frontier of ρ, then we define $\Lambda(H) \coloneqq \Lambda(\rho)_{|\mathsf{dom}(H)}$. Roughly speaking, identifying a reasonable set of partial frontiers is necessary and sufficient for computing the global state at the end of a finite run.

3 A New Model of Asynchronous Automata

Any infinite run ρ of an ATS $\mathfrak{T}$ over a trace $\theta \in \mathbb{R}(\Sigma, I)$ yields a partition $\Psi = (P_1, \ldots, P_n)$ of set $\mathcal{P}$ of processes such that each part $P_i \subseteq \mathcal{P}$ is minimal, and after finite prefixes $\rho_i \sqsubset \rho$, the processes $p \in P_i$ no longer interact directly or indirectly with another process $p' \in P_j, i \neq j$. We wish to obtain a family of ATS's where each process can infer during a run the part to which it belongs. Owing to space restrictions, we present a concise discussion here, and refer the reader to [1] for details and for proofs of all the claims made in this section.

3.1 Degrees of Synchronization

For an ATS $\mathfrak{T}$ and a run ρ of $\mathfrak{T}$ over any trace, we associate with each event $e \in \rho$ a measure of how much information is exchanged among the processes in $\mathsf{dom}(e)$. We use sets $P \subseteq \mathcal{P}$ of processes as the gauge for this measure.

Definition 5. *For a run* ρ *of an ATS and an event* $e \in \rho$, *let the* secondary update at e *be the set* $\mathcal{U}_e \coloneqq \{g \in \rho[e] \mid \exists p, q, r \in \mathcal{P}, \exists f_p \lessdot_p e : g = \mathsf{latest}_{p \to q \to r}(f_p) \neq \mathsf{latest}_{p \to q \to r}(e)\}$. *Then, the* degree of synchronization at e *is defined as as the set* $\mathsf{ds}(e) \coloneqq \bigcup_{g \in \mathcal{U}_e} \mathsf{dom}(\top_{\rho[e]}(g))$. *By default,* $\mathsf{ds}(e_\perp) \coloneqq \mathcal{P}$.

The set $\mathsf{ds}(e)$ implies that there must exist prefixes $\rho' \sqsubseteq \rho[e]$ with partial frontiers H, $\mathsf{dom}(H) = \mathsf{ds}(e)$, such that for some process $p \in \mathsf{dom}(e)$ with a predecessor $f_p \lessdot_p e$, $H \not\subseteq \rho[f_p]$. The following lemma illustrates this point, and demonstrates the importance of the set $\mathcal{U}_e$.

Lemma 6. *For $e \in \rho$, $e > e_\bot$, let $\rho_\sqcap := \bigsqcap_{f_p \lessdot_p e} \rho[f_p]$ be the greatest lower bound of all its p-prefixes. For every prefix $\rho' \sqsubseteq \rho[e]$ with $\rho' \not\sqsubseteq \rho_\sqcap$, there exist $H \subseteq \rho'$ and $U \subseteq \mathcal{U}_e$ such that 1. H is a partial frontier in ρ' with $\mathsf{dom}(H) = \mathsf{ds}(e)$; and 2. $\bigcup_{g \in U} \top_{\rho'}(g) = H$.*

Example 7. Referring to Fig. 3, at e_4, we have $e_2 \lessdot_q e_4$ and $e_3 \lessdot_r e_4$. Then, $\mathsf{ds}(e_4) = \mathcal{P}$ because $\mathcal{U}_{e_4} = \{e_\bot, e_1, e_2, e_3\}$. For instance $e_\bot = \mathsf{latest}_{q \to r \to s}(e_2) \neq \mathsf{latest}_{q \to r \to s}(e_4)$. Since $\rho_\sqcap = \rho[e_1]$, we have four possibilities of ρ', viz. $\rho'_1 = \rho[e_4]$, $\rho'_2 = \rho[e_2, e_3]$, $\rho'_3 = \rho[e_3]$, and $\rho'_4 = \rho[e_2]$. For ρ'_4, $H = \{e_\bot, e_1, e_2\}$ and we can choose $U = e_\bot \subseteq \mathcal{U}_{e_4}$. Symmetrically for ρ'_3. Also verify that, for ρ'_2, $H = U = \{e_2, e_3\}$; and for ρ'_1, $H = \{e_2, e_3, e_4\}$ and $U = \{e_\bot\}$.

Considering e_9 next, we have $e_8 \lessdot_q e_9$, $e_6 \lessdot_r e_9$, and $\mathcal{U}_{e_9} = \{e_2, e_4, e_6, e_8\}$. For instance, $e_2 = \mathsf{latest}_{r \to q \to p}(e_6) \neq \mathsf{latest}_{r \to q \to p}(e_9) = e_8$. Clearly, $\mathsf{ds}(e_9) = \{p, q, r\}$. And since $\rho_\sqcap = \rho[e_6]$, we have three possibilities of $\rho' \sqsubseteq \rho[e_9]$ s.t. $\rho' \not\sqsubseteq \rho_\sqcap$, the most interesting one being $\rho' = \rho[e_7]$. Now $H = \{e_4, e_6, e_7\}$ is the partial frontier of $\rho[e_7]$ with $\mathsf{dom}(H) = \mathsf{ds}(e_9)$, so we choose $U = \{e_2\} \subseteq \mathcal{U}_{e_9}$. $\boxtimes$

Remark 8. If $\mathcal{M}_e$ is the set of the (mutually concurrent) minimal events of $\mathcal{U}_e$, then it suffices to always consider $U = \mathcal{M}_e$ in Lemma 6.

Why we are interested in precisely these frontiers will be clear from Lemma 9 and Remark 10 below. Presently, with respect to the partial frontiers H that are revealed by Lemma 6 at an event e, we refer to the set Y_e of states $\Lambda(H)$ as the *yield at e*. Clearly, for each $\pi_1, \pi_2 \in Y_e$: $\mathsf{dom}(\pi_1) = \mathsf{dom}(\pi_2) = \mathsf{ds}(e)$. We say that the yield Y_e is *bigger* than yield Y_f if $\mathsf{ds}(f) \subsetneq \mathsf{ds}(e)$.

Lemma 9. *For an infinite run ρ and $p \in \mathcal{P}$, if $p \in \mathsf{dom}(\mathsf{alphinf}(\rho))$ then there exists a unique maximal $P \subseteq \mathcal{P}$ such that $\exists^\infty e \in \rho : p \in \mathsf{dom}(e) \wedge \mathsf{ds}(e) = P$.*

We call the set P from Lemma 9 the *max-degree of p-synchronizations in ρ*, denoted by $\lceil \mathsf{ds}_p(\rho) \rceil$. For processes $p \notin \mathsf{dom}(\mathsf{alphinf}(\rho))$ that eventually halt, we define $\lceil \mathsf{ds}_p(\rho) \rceil := \{p\}$ regardless of the value of $\mathsf{ds}(\max_p(\rho))$ The following remark follows immediately from Lemma 9, and demonstrates the "symmetric" nature of max-degree of synchronizations.

Remark 10. For an infinite run ρ and $p, q \in \mathcal{P}$, either $\lceil \mathsf{ds}_p(\rho) \rceil = \lceil \mathsf{ds}_q(\rho) \rceil$ or $\lceil \mathsf{ds}_p(\rho) \rceil \cap \lceil \mathsf{ds}_q(\rho) \rceil = \emptyset$.

In particular, for each part $P_i \in \Psi$: $q \in P_i \Leftrightarrow \lceil \mathsf{ds}_q(\rho) \rceil = P_i$. This concretizes our observation that every run ρ induces a partition Ψ of the set of states, where each part is minimal.

Definition 11. *A* synchronization-aware transition system *(an* SATS*) is a pair* $(\mathfrak{T}, \mathcal{D})$ *where* $\mathfrak{T} = ((X_p)_{p\in\mathcal{P}}, (\delta_a)_{a\in\Sigma}, \pi_0)$ *is an ATS and* $\mathcal{D} = (\mathcal{D}_p)_{p\in\mathcal{P}}$ *is a collection of mappings* $\mathcal{D}_p\colon X_p \to 2^{\mathcal{P}}$ *such that 1.* $\mathcal{D}_p(\pi_{0|p}) = \mathcal{P}$*, and 2. for every run* ρ *of* $\mathfrak{T}$ *and every event* $e \in \rho$*, if* $\Lambda(e) = \pi$ *and* $p \in \mathsf{dom}(e)$ *then* $\mathsf{ds}(e) = P \Leftrightarrow \mathcal{D}_p(\pi_{|p}) = P$.

This definition implies that the local p-states of an SATS always match the degrees of synchronization of events where they occur. It is easy to see that property 2 therein is in fact decidable, whence the definition is "syntactic".

3.2 Synchronization-aware Asynchronous Büchi Automata

A set $X \subseteq X_p$ of local p-states is called *homosynchronous* if for all local p-states $x, y \in X\colon \mathcal{D}_p(x) = \mathcal{D}_p(y)$. For an infinite run ρ of an SATS, we define the homosynchronous *maximal local infinity sets* $\lceil \mathsf{Inf}_p(\rho) \rceil$ as follows.

$$\lceil \mathsf{Inf}_p(\rho) \rceil := \begin{cases} \left\{ x \in X_p \;\middle|\; \begin{array}{l} \mathcal{D}_p(x) = \lceil \mathsf{ds}_p(\rho) \rceil \text{ and} \\ \exists^\infty e \in \rho : \Lambda(e)_{|p} = x \end{array} \right\} & \text{if } p \in \mathsf{dom}(\mathsf{alphinf}(\theta)), \\ \left\{ x \in X_p \;\middle|\; \begin{array}{l} \exists e \in \rho : e = \max_p(\rho) \\ \text{and } \Lambda(e)_{|p} = x \end{array} \right\} & \text{otherwise.} \end{cases}$$

Definition 12. *A* deterministic, synchronization-aware asynchronous Büchi automaton *(a* D-SABA*) is a tuple* $\mathfrak{A} = (\mathfrak{T}, \mathcal{D}, \mathcal{F})$*, where* $(\mathfrak{T}, \mathcal{D})$ *is an SATS, and the acceptance table* $\mathcal{F} = \{(Q_1, F_1), \ldots (Q_k, F_k)\}$ *is such that each* $Q_i \subseteq \mathcal{P}$ *and* $F_i = (F_i^p)_{p\in\mathcal{P}}$ *is a tuple of homosynchronous sets* F_i^p*. A D-SABA* $\mathfrak{A}$ accepts a trace $\theta \in \mathbb{R}(\Sigma, I)$ *if, for the run* ρ *of* $\mathfrak{A}$ *on* θ*, there exists a pair* $(Q_i, F_i) \in \mathcal{F}$ *s.t.* $\mathsf{dom}(\mathsf{alphinf}(\theta)) = Q_i$ *and for each process* $p \in \mathcal{P}$: $F_i^p \cap \lceil \mathsf{Inf}_p(\rho) \rceil \neq \emptyset$.

The above definition essentially requires that processes p ignore all of their infinitely occurring local p-states except those whose image under $\mathcal{D}_p$ matches the maximal degree of p-synchronizations. One of our main results is as follows.

Theorem 13. *A language* $\Theta \subseteq \mathbb{R}(\Sigma, I)$ *is recognized by a D-SABA iff* Θ *is a deterministic trace language, i.e.* Θ *can be expressed as a finite union of languages of the form* $\mathsf{lim}_A(T)$ *for regular languages* $T \subseteq \mathbb{M}(\Sigma, I)$ *and sets* $A \subseteq \Sigma$.

We prove this claim by breaking it up into Lemmas 14 and 15, and Prop. 16.

Lemma 14. *Given a regular trace language* $T \subseteq \mathbb{M}(\Sigma, I)$ *and a set* $A \subseteq \Sigma$*, there exists a D-SABA accepting* $\Theta = \mathsf{lim}_A(T)$.

To prove Lemma 14, we start with a DAA $\mathfrak{A} = (\mathfrak{T}, F)$ recognizing T and construct a D-SABA $\mathfrak{A}' = (\mathfrak{T}', \mathcal{D}, \mathcal{F})$ with such an SATS $(\mathfrak{T}', \mathcal{D})$ that (a) it mimics the run ρ of $\mathfrak{T}$ on every trace; and (b) at each event e in its own run ρ', it computes the yield Y_e for the corresponding event e in the run ρ of $\mathfrak{T}$.

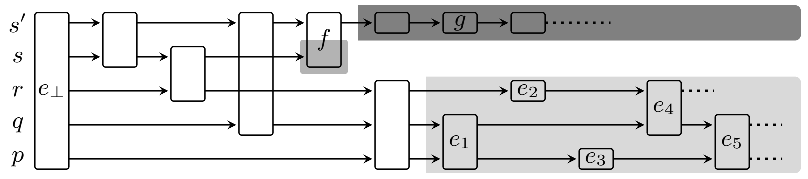


Fig. 4: Processes eventually halt, or settle in maximally interacting sets.

Fig. 4 illustrates a run ρ induced by a trace $\theta \in \mathsf{lim}_A(T)$ on $\mathfrak{A}$. The shaded regions represent the partition Ψ of $\mathcal{P}$ induced by ρ. Note that $f = \max_s(\rho)$ and $\lceil \mathsf{ds}_s(\rho) \rceil = \{s\}$ even though $\mathsf{ds}(f) = \mathcal{P}$. It is easy to see here that all partial frontiers H' in the top region are concurrent to all partial frontiers H'' in the bottom region. This means that $H' \cup H'' \cup \{f\}$ are partial frontiers of some prefixes $t \sqsubset \rho$. In particular, if $\mathsf{dom}(H'), \mathsf{dom}(H'') \in \Psi$ then $H \coloneqq H' \cup H'' \cup \{f\}$ is a frontier, and $\Lambda(H)$ is the global state at t.

Lemma 6 helps in retroactively computing partial frontiers. One can verify that $\mathsf{ds}(e_5) = \{p, q, r\}$ and $H'' = \{e_1, e_2, e_3\}$ is one of the partial frontiers computed at e_5. Then $\Lambda(H'')$ belongs to the yield Y_{e_5} at e_5. Similarly, at g we have $Y_g = \{\Lambda(g)\}$. Lastly, if $\pi_s = \Lambda(f)_{|\{s\}}$ is the $\{s\}$-state at f, then by "joining" the yields Y_{e_5} and Y_g with π_s, we obtain a set Π of global states which contains the state $\Lambda(H)$ at prefix $\rho[H] \sqsubset \rho$, for $H = \{e_1, e_2, e_3, f, g\}$.

However, such computations of global states are only required at the "end" of the infinite run ρ. By joining π_s with the maximal yields that occur infinitely often (as guaranteed by Lemma 9 and Remark 10), $\mathfrak{T}'$ can compute precisely the set of global states occurring infinitely often in the run ρ of $\mathfrak{A}$.

Consequently, a local p-state of the SATS $\mathfrak{T}'$ is of the form $\overline{x} = (x, \overline{\mathsf{Sec}}, Y)$, where x is a local p-state of $\mathfrak{T}$, $\overline{\mathsf{Sec}}$ is a finite data structure to help compute the yields, and Y is a yield. $\mathfrak{T}'$ ensures that for each $e \in \rho'$ of $\mathfrak{T}'$, $\Lambda(e)_{|p} = (x, \overline{\mathsf{Sec}}, Y)$ <u>iff</u> for the corresponding $e \in \rho$ of $\mathfrak{T}$, $\Lambda(e)_p = x$ and $Y = Y_e$ is the yield at e.

Since the set $Q \coloneqq \mathsf{dom}(A)$ of "live" processes is given, $\mathfrak{T}'$ can distinguish between the cases, e.g., that $p \in \mathsf{dom}(\mathsf{alphinf}(\theta))$ and $s \notin \mathsf{dom}(\mathsf{alphinf}(\theta))$ as shown in Fig. 4. By observing the sets $\lceil \mathsf{Inf}_p(\rho') \rceil, p \in \mathcal{P}$ in its run ρ', $\mathfrak{T}'$ can extract (a) the infinitely recurring maximal yields Y_p of $\mathfrak{T}$, from infinitely recurring maximal p-states $\overline{x}$ of live processes p; and (b) the final $\{p\}$-states π_p of $\mathfrak{T}$, from the final p-states $\overline{x}$ for processes p that halt.

Thus, $\mathfrak{T}'$ computes the set Π of global states occurring infinitely often in the run ρ of $\mathfrak{A}$. The run ρ' of $\mathfrak{T}'$ is accepting if Π has a non-empty intersection with the acceptance set F of $\mathfrak{A}$. The Büchi acceptance table $\mathcal{F} = \{(Q, F_1), \ldots (Q, F_k)\}$ is defined accordingly. For precise construction and proofs, see [1].

Lemma 15. *If $\mathfrak{A} = (\mathfrak{T}, \mathcal{D}, \mathcal{F})$ is a D-SABA with $|\mathcal{F}| = 1$ and $L(\mathfrak{A}) = \Theta$, then there exists a set $A \subseteq \Sigma$ and $T \subseteq \mathbb{M}(\Sigma, I)$ regular such that $\Theta = \mathsf{lim}_A(T)$.*

The proof of this lemma relies on constructing a non-deterministic asynchronous automaton recognizing the language T such that if $\mathcal{F} = \{(Q, F)\}$ then for $A \coloneqq \mathsf{dom}^{-1}(Q) \setminus \mathsf{dom}^{-1}(\mathcal{P} \setminus Q)$ it holds that $\Theta = \mathsf{lim}_A(T)$ (cf. [1]).

Proposition 16. *The family of D-SABA-recognizable languages is closed under finite unions.*

Hence, Thm. 13 follows. Lastly, following the result established for the class of deterministic trace languages in [7], one obtains that the family of D-SABA-recognizable languages is also closed under finite intersections.

3.3 Synchronization-aware Asynchronous Muller Automata

We now define the class of synchronization-aware asynchronous Muller automata that accept precisely the ω-regular trace languages.

Definition 17. *A* deterministic synchronization-aware asynchronous Muller automaton *(a D-SAMA) is a tuple $\mathfrak{A} = (\mathfrak{T}, \mathcal{D}, \mathcal{F})$, where $(\mathfrak{T}, \mathcal{D})$ is an SATS and the acceptance table $\mathcal{F} = \{F_1, \dots F_k\}$ is s.t. $F_i = (F_i^p)_{p\in\mathcal{P}}$ are tuples of homosynchronous sets F_i^p. A D-SAMA $\mathfrak{A}$* accepts a trace $\theta \in \mathbb{R}(\Sigma, I)$ *if, for the run ρ of $\mathfrak{A}$ on θ, there exists a tuple $F_i \in \mathcal{F}$ s.t. for each process $p \in \mathcal{P}$: $\lceil \mathsf{Inf}_p(\rho) \rceil = F_i^p$.*

Theorem 18. *Any language $\Theta \subseteq \mathbb{R}(\Sigma, I)$ of infinite traces is recognized by a D-SAMA if and only if Θ is recognized by a DAMA.*

The proofs of this theorem and of the result that the family of D-SAMAs is closed under Boolean operations may be found in [1].

4 Characterization of Deterministic Büchi Recognizability

A prominent result on ω-regular word languages, due to Landweber [8], states that a language $L \subseteq \Sigma^\omega$ is deterministically Büchi recognizable iff for some (in fact, for each) deterministic Muller automaton recognizing L the acceptance component is closed under supersets. The stronger (bracketed) version supplies a decision procedure for Büchi recognizability of ω-regular languages. Here we present a weaker existential characterization over infinite traces (see appendix for proofs). We define supersets in a manner that retains the essence of acceptance tables. Consider $F_1 = (F_1^p)_{p\in\mathcal{P}}$ and $F_2 = (F_2^p)_{p\in\mathcal{P}}$ from $\mathcal{F}$ where both F_1 and F_2 are tuples of homosynchronous sets F_1^p and $F_2^p, p \in \mathcal{P}$. We say that F_1 *is a superset of* F_2 denoted $F_1 \supseteq F_2$ if for each $p \in \mathcal{P}$, $F_1^p \supseteq F_2^p$. A table $\mathcal{F}$ is said to be *closed under supersets* if $\big((F \in \mathcal{F}) \wedge (F' \supseteq F)\big) \Rightarrow (F' \in \mathcal{F})$.

While discussing the closure under supersets, we must exempt the acceptance tuples that guarantee the halting of some processes. Let $F \in \mathcal{F}$ be a realizable acceptance tuple with $F^p = \{x\} \subseteq X_p$ for some $p \in \mathcal{P}$. Process p is guaranteed to halt during any run ρ that is accepted by referring to F only if it is the case that during two successive visits to x, p must visit another state $y \in X_p$ such that $\mathcal{D}_p(y) \subsetneq \mathcal{D}_p(x)$. Then p must halt because otherwise, either $\lceil \mathsf{ds}_p(\rho) \rceil \supsetneq \mathcal{D}_p(x)$ or $\lceil \mathsf{Inf}_p(\rho) \rceil \supsetneq \{x\}$. Such a singleton F^p is referred to as a *finitary acceptance set.*

Definition 19. *A Muller acceptance table $\mathcal{F}$ is said to be* closed under supersets modulo finitary acceptance sets *if (a) whenever $F \in \mathcal{F}$ does not contain any finitary acceptance sets and $F' \supseteq F$, then $F' \in \mathcal{F}$; and (b) whenever $F \in \mathcal{F}$ contains a finitary acceptance set F^p and $F' \supseteq F$ with $F'^p = F^p$, then $F' \in \mathcal{F}$.*

Theorem 20. *A language Θ is recognized by a D-SABA $\mathfrak{B} = (\mathfrak{T}', \mathcal{D}', \mathcal{F}')$ if and only if Θ is recognized by a D-SAMA $\mathfrak{A} = (\mathfrak{T}, \mathcal{D}, \mathcal{F})$ whose acceptance table $\mathcal{F}$ is closed under supersets modulo finitary acceptance sets.*

As mentioned previously, every ω-regular trace language can be written as a finite Boolean combination of A-infinitary limit languages [3]. Our results allow us to state an equivalent claim by referring to classes of automata.

Theorem 21. *For any language $\Theta \subseteq \mathbb{R}(\Sigma, I)$ of infinite traces, Θ is D-SAMA recognizable if and only if Θ can be expressed as a finite Boolean combination of D-SABA recognizable languages.*

5 Conclusion

We introduced synchronization-aware asynchronous transition systems that allow us to define for the first time the family of deterministic Büchi automata that matches the expressive power of the lim operator for trace languages. Not only is this definition a generalization of that for the word case but, more importantly, the corresponding languages are closed under finite unions and intersections – analogous to the deterministically Büchi recognizable word languages. In this sense, our results have further justified Muscholl's definition of "deterministic trace languages" as finite unions of parameterized lim-languages. Finally, we have also characterized deterministically Büchi recognizable trace languages in terms of recognition via a special subset of deterministic Muller automata.

The results of this paper uncover a clear path for completing a structure theory of regular infinitary trace languages. In ongoing work, we address the issue of weak recognizability, leading to a definition of weak D-SAMA's recognizing the languages that can be expressed as Boolean combinations of reachability trace languages. A next step is concerned with conceivable characterization of these weak trace languages as those that are recognized by both D-SABA's and D-SAcBA's (the latter equipped with the co-Büchi acceptance condition). Finally, it would be interesting to establish decidability of membership in each of these subclasses, for instance, by showing a strong Landweber theorem as indicated at the beginning of Section 4.

Acknowledgement I am grateful to Wolfgang Thomas for encouragement and numerous suggestions for improving this paper, to Marcus Gelderie and Christof Löding for many fruitful discussions. I also thank anonymous referees of a previous conference to which a prior version of this paper was submitted.

References

1. Namit Chaturvedi. Languages of infinite traces and deterministic asynchronous automata. Technical Report AIB-2014-04, RWTH Aachen University, Feb 2014.
2. Namit Chaturvedi and Marcus Gelderie. Weak ω-Regular Trace Languages. arXiv.org, Feb 2014. arXiv:1402.3199 [cs.FL].
3. Volker Diekert and Anca Muscholl. Deterministic asynchronous automata for infinite traces. *Acta Informatica*, 31(4):379–397, 1994.
4. Volker Diekert and Grzegorz Rozenberg, editors. *The Book of Traces*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1995.
5. Paul Gastin and Antoine Petit. Asynchronous cellular automata for infinite traces. In W. Kuich, editor, *Automata, Languages and Programming*, volume 623 of *Lecture Notes in Computer Science*, pages 583–594. Springer, 1992.
6. Mukund Madhavan. Automata on distributed alphabets. In Deepak D'Souza and Preeti Shankar, editors, *Modern Applications of Automata Theory*, volume 2 of *IISc Research Monographs Series*, pages 257–288. World Scientific, May 2012.
7. Anca Muscholl. *Über die Erkennbarkeit unendlicher Spuren*. PhD thesis, 1994.
8. Dominique Perrin and Jean-Éric Pin. *Infinite Words: Automata, Semigroups, Logic and Games*, volume 141 of *Pure and Applied Mathematics*, chapter Automata and Infinite Words. Elsevier, 2004.

A Proofs from Section 4

A.1 Discussion and proof of Theorem 20

Theorem. *A language Θ is recognized by a D-SABA $\mathfrak{B} = (\mathfrak{T}', \mathcal{D}', \mathcal{F}')$ if and only if Θ is recognized by a D-SAMA $\mathfrak{A} = (\mathfrak{T}, \mathcal{D}, \mathcal{F})$ whose acceptance table $\mathcal{F}$ is closed under supersets modulo finitary acceptance sets.*

For the proof of this theorem, we would like to recall the data structure called *latest appearance record*. Over a finite set $X = \{x_1, \ldots x_N\}$, we define the latest appearance record $\mathsf{LAR} \coloneqq X! \times [1, N]$. Any $M = \big((x_{i_1} x_{i_2} \ldots x_{i_N}), m\big) \in \mathsf{LAR}$ is a pair containing a permutation $(x_{i_1} x_{i_2} \ldots x_{i_N})$ of X and $1 \leq m \leq N$. The number m is usually called the *hit value*, and the set $\{x_{i_1}, \ldots x_{i_m}\}$ of the first m elements in the permutation is referred to as the *hit set* of M. We also refer to an update function $\upsilon \colon \mathsf{LAR} \times X \to \mathsf{LAR}$ is given by $\upsilon \colon \big((x_{i_1}, \ldots x_{i_N}), m\big), x \mapsto \big((x_{i_\ell}, x_{i_1}, \ldots x_{i_{\ell-1}}, x_{i_{\ell+1}}, \ldots x_{i_N}), \ell\big)$ where $x = x_{i_\ell}$.

Now, for the set X_p of local p-states of $\mathfrak{T}$, let $X_{p,Q_1}, X_{p,Q_2}, \ldots X_{p,Q_{n_p}}$ be the maximal homosynchronous subsets of X_p. That is, $\forall x \in X_p \colon x \in X_{p,Q_i} \Leftrightarrow \mathcal{D}_p(x) = Q_i$. For each $i, 1 \leq i \leq n_p$, we now define the latest appearance record $\mathsf{LAR}_{p,Q_i} \coloneqq (X_{p,Q_i})! \times [1, N_i]$ where $N_i \coloneqq |X_{p,Q_i}|$. Note that it may be the case that $n_p \neq n_q$ for $p, q \in \mathcal{P}$.

Proof (Theorem 20). For one direction of the theorem, let $\mathfrak{A} = (\mathfrak{T}, \mathcal{D}, \mathcal{F})$ be a D-SAMA such that $\mathcal{F}$ is closed under supersets modulo finitary acceptance sets.

Construct a D-SABA $\mathfrak{B} = (\mathfrak{T}', \mathcal{D}', \mathcal{F}')$ where the local p-state sets X'_p of $\mathfrak{T}'$ are given by $X'_p \coloneqq X_p \times \mathsf{LAR}_{p,Q_1} \times \mathsf{LAR}_{p,Q_2} \times \ldots \times \mathsf{LAR}_{p,Q_{n_p}}$. The initial local p-state in $\mathfrak{T}'$ is $(\pi_{0|p}, M_{p,1}, \ldots, M_{p,n_p})$ where π_0 is the global initial state of $\mathfrak{T}$ and $M_{p,i}$ are arbitrarily chosen initial records. The transition functions are given by $\delta'_a \colon \big((x_p, M_{p,1}, \ldots, M_{p,n_p})\big)_{p \in \mathsf{dom}(a)} \mapsto \big((y_p, L_{p,1}, \ldots, L_{p,n_p})\big)_{p \in \mathsf{dom}(a)}$ where:

– $(y_p)_{p \in \mathsf{dom}(a)} = \delta_a\big((x_p)_{p \in \mathsf{dom}(a)}\big)$, and
– if $\mathcal{D}_p(y_p) = Q_{i_p}$ then $L_{p,j} \coloneqq \begin{cases} M_{p,j} & \text{if } j \neq i_p \\ \upsilon(M_{p,j}, y_p) & \text{otherwise.} \end{cases}$

The mapping $\mathcal{D}'_p$ for $\mathfrak{T}'$ is defined as $\mathcal{D}'_p\big((x_p, \ldots, M_{p,i}, \ldots)\big) \coloneqq \mathcal{D}_p(x_p)$.

In order to define the Büchi acceptance table, consider any Muller acceptance tuple $F \in \mathcal{F}$ and a set $R \subsetneq \mathcal{P}$ such that if $r \in R$ then $|F^r| = 1$, and if F^r is a finitary acceptance set then $r \in R$. Intuitively, the processes in R are earmarked as precisely the ones that will halt. Create a Büchi acceptance tuple $(\mathcal{P} \setminus R, F'_R)$ where for each $p \in \mathcal{P}$, assuming $\mathcal{D}_p(F^p) = Q_{k}$,:

– if $p \notin R$ then $F'^p_R \coloneqq \{(x, M_{p,1}, \ldots, M_{p,k}, \ldots, M_{p,n_p}) \in X'_p \mid \mathcal{D}_p(x) = Q_k$ <u>and</u> the hit set $H_{p,k}$ of $M_{p,k}$ is a superset of $F^p\}$; otherwise
– if $p \in R$ then $F'^p_R \coloneqq \{(x, M_{p,1}, \ldots M_{p,n_p}) \in X'_p \mid \mathcal{D}_p(x) = Q_k$ <u>and</u> $x \in F^p\}$.

Note that for a given Muller acceptance tuple $F \in \mathcal{F}$, we may obtain multiple Büchi acceptance tuples $(\mathcal{P} \setminus R_1, F'_{R_1}), (\mathcal{P} \setminus R_2, F'_{R_2}), \ldots$ where each R_i contains (some of) the processes r for which the corresponding Muller set F^r is singleton. R_i may be empty only if there are no finitary acceptance sets in F.

Now, a trace $\theta \in \mathbb{R}(\Sigma, I)$ is accepted by the D-SABA $\mathfrak{B}$:

iff for the run ρ' of $\mathfrak{A}'$ on θ there exists $(Q, F') \in \mathcal{F}'$ such that for each $p \in \mathcal{P}$, $F'^p \cap \lceil \mathsf{Inf}_p(\rho') \rceil \neq \emptyset$; and this holds

iff for each processes $p \in \mathcal{P}$
- if $p \in Q$, then p witnesses a state $(x, X_1^p, \ldots, X_{i_p}^p, \ldots) \in F'^p$ infinitely often, where $\lceil \mathsf{ds}_p(\rho') \rceil = \mathcal{D}'_p\big((x, X_1^p, \ldots, X_{i_p}^p, \ldots)\big) = \mathcal{D}_p(X_{i_p}^p \cup \{x\})$; and
- if $p \notin Q$, then p halts at a state $(x, X_1^p, \ldots) \in F'^p$ – and in this case, let $X_{i_p}^p := \{x\}$;

and, by construction, this holds

iff $(X_{i_p}^p)_{p \in \mathcal{P}} \supseteq F$ for some Muller tuple $F \in \mathcal{F}$, satisfying $\forall p \notin Q$: $F^p = X_{i_p}^p$; and this holds

iff $(X_{i_p}^p)_{p \in \mathcal{P}} = G$ for some $G \in \mathcal{F}$ satisfying $\forall p \notin Q$: $G^p = F^p$ (since $\mathcal{F}$ is closed under supersets modulo finitary acceptance sets); and this holds

iff θ induces a run ρ on the D-SAMA $\mathfrak{A}$ such that for each process $p \in \mathcal{P}, G^p = \lceil \mathsf{Inf}_p(\rho) \rceil$; and this holds

iff the D-SAMA $\mathfrak{A}$ accepts θ.

For the other direction of the theorem, let us assume that Θ is recognized by a D-SABA $\mathfrak{B} = (\mathfrak{T}', \mathcal{D}', \mathcal{F}')$. We define a D-SAMA $\mathfrak{A} = (\mathfrak{T}, \mathcal{D}, \mathcal{F})$ whose acceptance table $\mathcal{F}$ is closed under supersets modulo finitary acceptance sets as follows:

– for each $p \in \mathcal{P}$, define $X_p := X'_p \times \{0,1\}^{(2^{|\mathcal{P}|}-1)}$;
– for each $(x, \mathbb{B}) \in X_p$, $\mathcal{D}_p((x, \mathbb{B})) := \mathcal{D}'_p(x)$;
– for $a \in \Sigma$ and $\pi \in X'_{\mathsf{dom}(a)}$, if $\delta'_a(\pi) = \pi'$ then define the new mapping $\delta_a\big(((\pi_{|p}, \mathbb{B}_p))_{p \in \mathsf{dom}(a)}\big) := \big((\pi'_{|p}, \mathbb{B}'_p)\big)_{p \in \mathsf{dom}(a)}$, where the new bit-vector is assigned[1] as $\mathbb{B}'_p[Q] := \begin{cases} 1 - \mathbb{B}_p[Q] & \text{if } Q = \mathcal{D}'_p(\pi_{|p}) \\ \mathbb{B}_p[Q] & \text{otherwise} \end{cases}$;
– if π_0 is the initial state of $\mathfrak{B}$, then the initial state of $\mathfrak{A}$ is $((\pi_{0|p}, \mathbb{B}_0))_{p \in \mathcal{P}}$, where $\mathbb{B}_0 = (0, \ldots 0)$.

Every time a process p moves out of a local p-state $(x, \mathbb{B})$ with $\mathcal{D}'_p(x) = Q$ and arrives in a new local state $(y, \mathbb{B}')$, it ensures that $\mathbb{B}[Q] \neq \mathbb{B}'[Q]$. By this construction, during a run ρ of $\mathfrak{A}$, a process p halts after finitely many transitions if and only if $\lceil \mathsf{Inf}_p(\rho) \rceil$ is a singleton. In particular, if a processes p loops infinitely often on the same state x in $\mathfrak{B}$, then in $\mathfrak{A}$ it will alternate between $(x, \mathbb{B})$ and $(x, \mathbb{B}')$ ad infinitum where $\mathbb{B}$ and $\mathbb{B}'$ must differ at index $\mathcal{D}'_p(x)$. This immediately provides a mechanism for defining the Muller acceptance table $\mathcal{F}$ of $\mathfrak{A}$.

For each acceptance pair $(Q, F') \in \mathcal{F}'$ of $\mathfrak{B}$, we define a number of Muller acceptance tuples $F \in \mathcal{F}$ for $\mathfrak{A}$ such that

[1] Since the number of bits in a bit-vector $\mathbb{B}$ is equal to the number of non-empty subsets of $\mathcal{P}$, we refer to the bit corresponding to a subset Q as $\mathbb{B}[Q]$.

– for each $p \notin Q$, $F^p = \{(x, \mathbb{B})\}$ for some $x \in F'^p$ and $\mathbb{B} \in \{0,1\}^{(2^{|\mathcal{P}|}-1)}$; and
– for each $p \in Q$, $|F^p| \geq 2$ and there exists a state $(x, \mathbb{B}) \in F^p$ such that $x \in F'^p$, and there exists at least one pair of states $(y, \mathbb{B}_1), (z, \mathbb{B}_2) \in F^p$ such that $\mathbb{B}_1$ and $\mathbb{B}_2$ differ (at the least) at index $\mathcal{D}'_p(x)$.

Clearly, the acceptance table $\mathcal{F}$ is closed under supersets modulo singletons, that is, it is closed under supersets modulo finitary acceptance sets.

Now it is trivial to show that $L(\mathfrak{A}) = L(\mathfrak{B})$. ■

A.2 Proof of Theorem 21

Theorem. *For any language $\Theta \subseteq \mathbb{R}(\Sigma, I)$ of infinite traces, Θ is D-SAMA recognizable if and only if Θ can be expressed as a finite Boolean combination of D-SABA recognizable languages.*

Proof. One direction of this theorem follows trivially from the facts that for each D-SABA there exists a D-SAMA (cf. Theorem 20) and that the family of D-SAMAs is closed under finite Boolean operations (cf. [1]).

We only need to show how a language recognized by a D-SAMA $\mathfrak{A} = (\mathfrak{T}, \mathcal{D}, \mathcal{F})$ can be expressed as a finite Boolean combination of D-SABA recognizable languages. For any $F_i \in \mathcal{F}$, with $F_i = (F_i^p)_{p \in \mathcal{P}}$,

– let $Y_i^p \subseteq X_p$ consist of all the p-states y such that for $x \in F_i^p$, $\mathcal{D}_p(x) = \mathcal{D}_p(y)$;
– let $\Pi_i \subseteq X_{\mathcal{P}}$ be a set of global states satisfying $\forall p \in \mathcal{P}\colon\ \bigcup_{\pi \in \Pi_i} \pi_{|p} = F_i^p$;
– let $Q_i \subseteq \mathcal{P}$ consist of all processes p such that $|F_i^p| = 1$.

For each $\pi \in \Pi_i$ define a D-SABA $\mathfrak{A}_{i,\pi} \coloneqq (\mathfrak{T}, \mathcal{D}, \mathcal{F}_{i,\pi})$ whose acceptance table $\mathcal{F}_{i,\pi} \coloneqq \bigcup_{R \subseteq Q_i} \{(\mathcal{P} \setminus R, (\{\pi_{|p}\})_{p \in \mathcal{P}})\}$. If a trace θ is accepted by $\mathfrak{A}_{i,\pi}$ then there exists some $R \subseteq Q_i$ that processes $q \notin R$ visit local q-states $\pi_{|q}$ infinitely often process $q \in R$ eventually stall at states $\pi_{|q}$. It is nowhere required that the global state π ever occurs. We only use π as an easy reference to local states that are drawn from the Muller acceptance tuple F_i.

Referring to languages $L(\mathfrak{A}_{i,\pi})$, we define the language $L_i^+ \coloneqq \bigcap_{\pi \in \Pi_i} L(\mathfrak{A}_{i,\pi})$ of traces inducing runs that, for some $R \subseteq Q_i$, result in processes $p \in R$ halting in p-states $x \in F_i^p$, and processes $p \notin R$ visiting all the states (and maybe more) from sets F_i^p.

Next, we define languages that will prevent the processes from visiting any more than their respective acceptance sets F_i^p. For each $p \in \mathcal{P}$, we define a D-SABA $\mathfrak{A}_{i,p} \coloneqq (\mathfrak{T}, \mathcal{D}, \mathcal{F}_{i,p})$ with $\mathcal{F}_{i,p} \coloneqq \bigcup_{Q \subseteq \mathcal{P}} \{(Q, (F_{i,p}^q)_{q \in \mathcal{P}})\}$ where the sets $F_{i,p}^q \coloneqq \begin{cases} Y_i^q \setminus F_i^q & \text{if } q = p \\ Y_i^q & \text{otherwise} \end{cases}$. The language $L(\mathfrak{A}_{i,p})$ consists of traces which, irrespective of the set Q of live processes, ensure that either p halts in a state outside of F_i^p (i.e. if $p \notin Q$) or p infinitely often visits states outside of F_i^p (i.e. if $p \in Q$). In this sense, $L(\mathfrak{A}_{i,p})$ consists of all traces on whose runs at least process p "misbehaves". Thus, the language $L_i^- \coloneqq \bigcup_{p \in \mathcal{P}} L(\mathfrak{A}_{i,p})$, comprises of traces on whose runs at least one process misbehaves.

Finally, we can express $L(\mathfrak{A}) = \bigcup_{F_i \in \mathcal{F}} \left(L_i^+ \cap \overline{L_i^-}\right)$. ■