Asynchronous cellular automata for pomsets*

Manfred Droste[1], Paul Gastin[2], and Dietrich Kuske[1†]

[1] Institut für Algebra, Technische Universität Dresden, D-01062 Dresden
{droste,kuske}@math.tu-dresden.de

[2] LIAFA, ERS 586, Université Paris 7, 2, place Jussieu, F - 75251 Paris Cedex 05
Paul.Gastin@liafa.jussieu.fr

Abstract

This paper extends to pomsets without auto-concurrency the fundamental notion of asynchronous cellular automata (ACA) which was originally introduced for traces by Zielonka. We generalize to pomsets the notion of asynchronous mapping introduced by Cori, Métivier and Zielonka and we show how to construct a deterministic ACA from an asynchronous mapping.

Then we investigate the relation between the expressiveness of monadic second order logic, nondeterministic ACAs and deterministic ACAs. We can generalize Büchi's theorem for finite words to a class of pomsets without auto-concurrency which satisfy a natural axiom. This axiom ensures that an asynchronous cellular automaton works on the pomset as a concurrent read and exclusive owner write machine. More precisely, in this class non-deterministic ACAs, deterministic ACAs and monadic second order logic have the same expressive power.

Then we consider a class where deterministic ACAs are strictly weaker than nondeterministic ones. But in this class nondeterministic ACAs still capture monadic second order logic. Finally it is shown that even this equivalence does not hold in the class of all pomsets since there the class of recognizable pomset languages is not closed under complementation.

1 Introduction

In a distributed system, some events may occur concurrently, meaning that they may occur in any order or simultaneously or even that their executions may overlap. This is the case in particular when two events use independent resources. On the other hand, some events may causally depend on each other. For instance, the receiving of a message must follow its sending. Therefore, a distributed behavior may be abstracted as a pomset, that is a set of events together with a partial order which describes causal dependencies of events and with a labeling function. In this paper, we mainly deal with

*Research supported by PROCOPE
†Supported by the German Research Foundation (DFG).

pomsets without auto-concurrency: concurrent events must have different labels. These pomsets are called semi-words in [Sta81, Die94]. For studies how general pomsets can be used to represent parallel processes and how they can be composed, we refer the reader e.g. to [Pra86, Gis88].

There are several ways to describe the behaviors of a system. For instance, logic formulas are suited for specification purposes. Depending on the properties we have to express, we can use various logics such as temporal logics, first order logics or (monadic) second order logics. On the other hand, transition systems are often used to give more operational descriptions. In this paper, we will concentrate on these two kinds of descriptions of systems.

When dealing with distributed systems, it is natural to look for transition systems which faithfully reflect the concurrency. For instance, Petri nets are a widely studied class of such transition systems. Asynchronous cellular automata (ACA) form another fundamental class of transition systems with built-in concurrency. They were introduced for traces by Zielonka [Zie87, Zie89]. Mazurkiewicz introduced traces in order to describe the behaviors of one-safe Petri nets [Maz77, Maz86]. A trace is a pomset where the partial order is dictated by a static dependence relation over the actions of the system.

The primary aim of this work is to generalize the notion of ACA so that they can work on pomsets without auto-concurrency. In Section 3, we define our notion of ACAs. There are two possible definitions of runs of an ACA on a pomset and for each of these definitions there are two possible criteria for acceptance. Thus, an ACA may work in four different *modes*. Section 4 starts with the proof that two of them are equivalent for nondeterministic ACAs.

Asynchronous mappings have proven to be a basic tool to construct ACAs for traces [CMZ93]. In Section 4.2, we give a definition of asynchronous mappings for general pomsets. We show that a pomset language recognized by an asynchronous mapping can be accepted by a *deterministic* ACA in any mode.

The rest of this paper is devoted to the relation between ACAs and monadic second order (MSO) logic for pomsets. We prove in Section 4.3 that from a (non deterministic) ACA one can construct a MSO formula which defines precisely the pomset language accepted by the automaton in a given mode. In Section 5, we prove the converse for the special subclass of pomsets for which the ACA works as a concurrent read and exclusive owner write (CROW) machine. These pomsets are called CROW-pomsets. More precisely, from a MSO formula we construct a *deterministic* ACA which in a given mode accepts precisely the CROW-pomsets defined by the formula. Therefore, for CROW-pomsets, we have the equivalence between (existential) MSO logic, deterministic ACAs and non deterministic ACAs (for any of the alternative modes). This result is crucial since it opens the way of model checking for distributed systems whose behaviors are described as CROW-pomsets.

In Section 6, we restrict our attention to k-pomsets and ACAs. We can show that the expressive power of non-deterministic ACAs in any mode, MSO logic and existential MSO logic coincide. Thus, in particular, any ACA running in a given mode can be simulated by a non-deterministic ACA that runs in any other mode. But this simulating automaton has to be non-deterministic since, as we show, the four modes give rise to incomparable concepts of deterministic recognizability. Thus, in the class of k-pomsets, non-deterministic ACAs are strictly more expressive than deterministic ones and deterministic ACAs are more expressive than asynchronous mappings. Furthermore, emptiness, universality and equivalence for ACAs are decidable within this class.

In Section 7, we return to the class of all pomsets. We show that in this context the modes give rise to two incomparable concepts of nondeterministic recognizability. This already implies that for pomsets MSO logic is strictly more expressive than ACAs in any mode. This negative result is sharpened in the final part where we present an example of a first-order definable pomset property that cannot be recognized by an ACA. Since the negation of this property is recognizable, we obtain that the class of recognizable pomset languages is not closed under complementation.

The proofs of the positive results are based on the classical result by Zielonka [Zie87], on a close analysis of the pomsets and automata in consideration, and on a technique developed by Thomas [Tho90] for asynchronous automata for traces. The negative results are shown by separating examples.

Preliminary versions of these results have appeared in the extended abstracts [DG96] and [Kus98].

2 Preliminaries

2.1 Pomsets

Let Σ be a finite set, called alphabet. A pomset over Σ is (an isomorphism class of) a finite labeled partial order $t = (V, \leq, \lambda)$ where V is a finite set of vertices, $\leq$ is the partial order on V and $\lambda : V \to \Sigma$ is the labeling function. The empty pomset $(\emptyset, \emptyset, \emptyset)$ will be denoted by 1. Throughout the paper we will mainly deal with pomsets without auto-concurrency, that is pomsets $t = (V, \leq, \lambda)$ such that $\lambda^{-1}(a)$ is totally ordered for all $a \in \Sigma$.

Let $s = (V_s, \leq_s, \lambda_s)$ and $t = (V_t, \leq_t, \lambda_t)$ be two pomsets. We say that s is a prefix of t, if s is (isomorphic to) a downward closed subpomset of t, that is, if V_s is a downward closed subset of V_t (i.e. $V_s \subseteq V_t$ and for all $u, v \in V_t$, $u \leq_t v$ and $v \in V_s$ imply $u \in V_s$), $\leq_s$ is the restriction of $\leq_t$ to V_s (i.e. $\leq_s = \leq_t \cap V_s \times V_s$) and λ_s is the restriction of λ_t to V_s. The prefix order relation is a partial order on the set of all pomsets (even if we allow auto-concurrency). Since we have assumed non auto-concurrency, there is a unique way to embed a prefix s of t as a downward closed subpomset of t. Hence, we will identify a downward closed subset of vertices with the corresponding prefix of the pomset. Let $s_1 = (V_{s_1}, \leq_{s_1}, \lambda_{s_1})$ and $s_2 = (V_{s_2}, \leq_{s_2}, \lambda_{s_2})$ be two prefixes of a pomset $t = (V_t, \leq_t, \lambda_t)$. Then, $V_{s_1} \cup V_{s_2}$ is a downward closed subset of V_t and the corresponding prefix of t is $s_1 \cup s_2 = (V_{s_1} \cup V_{s_2}, \leq_{s_1} \cup \leq_{s_2}, \lambda_{s_1} \cup \lambda_{s_2})$ where $\lambda_{s_1} \cup \lambda_{s_2}$ is the labeling which coincides with λ_{s_1} on V_{s_1} and with λ_{s_2} on V_{s_2} (note that λ_{s_1} and λ_{s_2} agree on $V_{s_1} \cap V_{s_2}$).

Let $t = (V, \leq, \lambda)$ be a pomset. The downward closure of a vertex v is denoted by $\downarrow v = \{u \in V \mid u \leq v\}$. The strict downward closure of a vertex v is denoted by

$\Downarrow v = \downarrow v \setminus \{v\}$. Since $\downarrow v$ and $\Downarrow v$ are downward closed subsets of V, we will identify these sets with the corresponding prefixes of t.

Let $\Sigma_1, \ldots, \Sigma_n$ be pairwise disjoint alphabets and let $\Sigma = \Sigma_1 \dot\cup \cdots \dot\cup \Sigma_n$. Intuitively, we can view $[n] = \{1, \ldots, n\}$ as a set of labels of sequential processes and $\Sigma_1, \ldots, \Sigma_n$ as the sets of actions of these sequential processes. Let $p : \Sigma \to [n]$ be the mapping which associates with each letter $a \in \Sigma$ the process $p(a) \in [n]$ which executes the letter a, i.e. $a \in \Sigma_{p(a)}$.

Let $t = (V, \leq, \lambda)$ be a pomset. We say that a vertex v covers a vertex u, denoted by $u \longrightarrow\!\!< v$, if $u < v$ and there is no vertex w such that $u < w < v$. We say that two vertices $u, v \in V$ are incomparable or concurrent, denoted by $u \parallel v$, if neither $u \leq v$ nor $u \geq v$. We may see the covering relation as the description of the interactions between the processes. More precisely, we consider that an event $v \in V$ reads the states of the processes $p \circ \lambda(\{u \mid u \longrightarrow\!\!< v\})$ and writes in the process $p \circ \lambda(v)$, which, by abuse of notation, will be abbreviated by $p(v)$. We will not allow concurrent writes, therefore two concurrent events $u \parallel v$ must write in different processes $p \circ \lambda(u) \neq p \circ \lambda(v)$. This leads to the following

Definition 2.1 *A* $(\Sigma_1, \Sigma_2, \ldots, \Sigma_n)$-pomset *or* $\vec{\Sigma}$-pomset *is a pomset* $t = (V, \leq, \lambda)$ *for which* $\lambda^{-1}(\Sigma_i)$ *is totally ordered for all* $1 \leq i \leq n$. *The set of all* $\vec{\Sigma}$*-pomsets will be denoted by* $\mathbb{P}(\vec{\Sigma})$.

Note that with this notation the set $\mathbb{P}(\Sigma)$ is the set of words over Σ. It is easily seen that $\vec{\Sigma}$-pomsets are special pomsets without auto-concurrency. If the sets $\Sigma_1, \ldots, \Sigma_n$ are all singletons $\mathbb{P}(\vec{\Sigma})$ is the set of all pomsets without auto-concurrency.

For $A \subseteq \Sigma$, we denote by $\partial_A(t) = \downarrow\lambda^{-1}(A)$ the least prefix of a pomset t which contains all elements labeled with letters from A. Note that $\partial_A(t) = \bigcup_{v \mid \lambda(v) \in A} \downarrow v$. For $a \in \Sigma$ and $i \in [n]$, we will use the following simplified notations: $\partial_a(t) = \partial_{\{a\}}(t)$ and $\partial_i(t) = \partial_{\Sigma_i}(t)$.

Note that if $\lambda^{-1}(A)$ is totally ordered then $\partial_A(t)$ is either empty or has exactly one maximal vertex. In particular, if t is a $\vec{\Sigma}$-pomset then $\partial_i(t)$ is either empty or has exactly one maximal vertex. Occasionally, we will identify $\partial_i(t)$ with its maximal vertex.

2.2 Traces

We recall now basic definitions for Mazurkiewicz traces which will be needed in this paper. The reader is referred to [DR95] for a general presentation of trace theory.

A dependence alphabet is a pair (Σ, D) where Σ is a finite alphabet and $D \subseteq \Sigma \times \Sigma$ is a reflexive and symmetric relation over Σ called the dependence relation. Intuitively, two dependent actions $(a, b) \in D$ must be executed sequentially while two independent actions $(a, b) \notin D$ may occur concurrently. More formally, one considers the congruence relation $\sim$ over the free monoid $\Sigma^\star$ generated by the relation $\{(ab, ba) \mid (a, b) \notin D\}$. A trace is simply an equivalence class of words for the congruence $\sim$. The trace monoid is then the quotient $\mathbb{M}(\Sigma, D) = \Sigma^\star / \sim$.

We give now an equivalent definition of traces which is more adequate in our context. Basically, a trace can be seen as a pomset which satisfies additional requirements. More precisely, we will see that a trace over the dependence alphabet (Σ, D) is a pomset $t = (V, \leq, \lambda)$ such that for all vertices $u, v \in V$,

$$(\lambda(u), \lambda(v)) \in D \implies u \leq v \text{ or } v \leq u \tag{1}$$
$$u \longrightarrow\!\!\!< v \implies (\lambda(u), \lambda(v)) \in D \tag{2}$$

Note that a linearization of a pomset t may be identified with a word of $\Sigma^\star$. Now, let $t = (V, \leq, \lambda)$ be a pomset satisfying conditions (1) and (2). Then, the set of linearizations of t is precisely a trace, that is, an equivalence class for $\sim$. Hence, with each pomset t satisfying (1) and (2), one can associate a trace $\varphi(t)$. Conversely, a word $u \in \Sigma^\star$ defines a labeled linear order $(V_u, \leq_u, \lambda_u)$ over the occurrences of actions of u: $V_u = \{(a, i) \mid 1 \leq i \leq |u|_a\}$ ($|u|_a$ denotes the number of occurrences of a in u); $(a, i) \leq_u (b, j)$ if the i-th a occurs before the j-th b in u; and $\lambda_u((a, i)) = a$. Since two equivalent words $u \sim v$ have the same set of occurrences of actions ($V_u = V_v$), we can associate with a trace $[u]$ the pomset $\psi([u]) = (V_u, \bigcap_{v \sim u} \leq_v, \lambda_u)$. One can check that $\psi([u])$ satisfies conditions (1) and (2) and that ψ and φ are inverse bijections. This explains why the two definitions are equivalent.

We will now define recognizable trace languages. A trace automaton is a quadruple $\mathcal{A} = (Q, T, I, F)$ where Q is a finite set of states, $I \subseteq Q$ is the set of initial states, $F \subseteq Q$ is the set of final states and $T \subseteq Q \times \Sigma \times Q$ is the set of transitions which satisfies the diamond property: for all $(a, b) \in (\Sigma \times \Sigma) \setminus D$ and $q, q', q'' \in Q$, if $(q, a, q') \in T$ and $(q', b, q'') \in T$ then there exists some $\overline{q}' \in Q$ such that $(q, b, \overline{q}') \in T$ and $(\overline{q}', a, q'') \in T$. A word $w = a_1 \cdots a_n \in \Sigma^\star$ is accepted by $\mathcal{A}$ if there is a run $q_0, a_1, q_1, \ldots, a_n, q_n$ such that $q_0 \in I$, $q_n \in F$ and $(q_{i-1}, a_i, q_i) \in T$ for all $1 \leq i \leq n$. A trace $t \in \mathbb{M}(\Sigma, D)$ is accepted by $\mathcal{A}$ if some linear extension of t is accepted by $\mathcal{A}$. Note that, thanks to the diamond property of a trace automaton, if some linear extension of a trace is accepted by $\mathcal{A}$ then all linear extensions of t are accepted by $\mathcal{A}$. A trace language $L \subseteq \mathbb{M}(\Sigma, D)$ is *recognizable* if it is the set of traces accepted by some trace automaton. Equivalently, it is a recognizable language in the monoid $\mathbb{M}(\Sigma, D)$, as usual, in the sense of [Eil74].

3 Asynchronous cellular automata

Definition 3.1 *A* $(\Sigma_1, \Sigma_2, \ldots, \Sigma_n)$-asynchronous cellular automaton *(or* $\vec{\Sigma}$*-ACA) is a tuple* $\mathcal{A} = ((Q_i)_{i \in [n]}, (\delta_{a,J})_{a \in \Sigma, J \subseteq [n]}, F)$ *where*

1. *for all* $i \in [n]$*,* Q_i *is a finite set of local states for process* i*,*
2. *for all* $a \in \Sigma$ *and* $J \subseteq [n]$*,* $\delta_{a,J} : \prod_{i \in J} Q_i \to \mathcal{P}(Q_{p(a)})$ *is a (nondeterministic) transition function (where* $\mathcal{P}$ *denotes the power set operator) and*
3. $F \subseteq \bigcup_{J \subseteq [n]} \prod_{i \in J} Q_i$ *is a set of accepting states.*

The automaton is deterministic *if all the transition functions are deterministic, i.e. if* $|\delta_{a,J}((q_i)_{i\in J})| \leq 1$ *for all* $a \in \Sigma$, $J \subseteq [n]$ *and* $q_i \in Q_i$ *for* $i \in J$.

We now explain how a $\vec{\Sigma}$-ACA can accept a $\vec{\Sigma}$-pomset $t = (V, \leq, \lambda)$. The idea is that the $\vec{\Sigma}$-ACA consists of n local processes whose local states are Q_i. Then, any event $x \in V$ changes the state of its process $p \circ \lambda(x)$, only. This change depends on the local states of the processes in the read domain of this event. There are (at least) two reasonable read domains of an event x: The first one is that it reads only the local states reached at the events covered by x. In particular, in this mode it may happen that x does not read the last state of its own process $p \circ \lambda(x)$. In the second reading mode, x reads for each process that acted in the past of x the state at the last event below x on this process. Since the first reading mode is a restriction of the second one, we refer to it as the R^--mode. The second is called the R^+-mode.

To define these two kinds of runs uniformly, let $t = (V, \leq, \lambda)$ be a $\vec{\Sigma}$-pomset and $x \in V$. Then

$$\begin{aligned} \mathrm{R}^-(x) &:= p \circ \lambda\{y \in V \mid y \longrightarrow\!\!\!< x\} = p \circ \lambda(\max(\Downarrow x)) \text{ and} \\ \mathrm{R}^+(x) &:= p \circ \lambda\{y \in V \mid y < x\} = p \circ \lambda(\Downarrow x). \end{aligned}$$

Note that $\mathrm{R}^-(x) \subseteq \mathrm{R}^+(x)$. For $\alpha \in \{+, -\}$, an R^α*-run of* $\mathcal{A}$ *on* t is a function $r : V \to \bigcup_{i\in[n]} Q_i$ such that

$$r(x) \in \delta_{\lambda(x),\mathrm{R}^\alpha(x)}(r(\partial_i(\Downarrow x))_{i\in\mathrm{R}^\alpha(x)})$$

for any $x \in V$.

Note that an R^--run can be seen as a run on the Hasse-diagram of the $\vec{\Sigma}$-pomset. To compare it with an R^+-run, let $t = (V, \leq, \lambda)$ be a $\vec{\Sigma}$-pomset and define the binary relation E on V by

$$E = \{(\partial_i(\Downarrow x), x) \mid x \in V, i \in \mathrm{R}^+(x)\}.$$

Then $\longrightarrow\!\!\!<\, \subseteq E \subseteq\, \leq$. Hence $\leq$ is the transitive and reflexive closure of E. An R^+-run can be seen as a run on the directed acyclic graph (V, E, λ).

Let us consider two examples that make the difference between the two reading modes clear. In both examples, we present a deterministic $\vec{\Sigma}$-ACA and then describe the set of all $\vec{\Sigma}$-pomsets that admit an R^α-run of this automaton. Later, we will see that these sets cannot be accepted by a $\vec{\Sigma}$-ACA in the other reading mode.

Example 3.2 Let $n = 3$, $\Sigma_1 = \{a\}$, $\Sigma_2 = \{b\}$ and $\Sigma_3 = \{c\}$. We consider the automaton given by $Q_1 = Q_2 = Q_3 = \{q\}$ and

$$\delta_{x,J}((q_i)_{i\in J}) = \begin{cases} \emptyset & x = c \text{ and } 1 \in J \\ \{q\} & \text{otherwise} \end{cases}$$

for any $x \in \{a, b, c\}$. Next we describe the set of $\vec{\Sigma}$-pomsets that admit an R^--run of this automaton: Since there is only one state, for any pomset $t = (V, \leq, \lambda)$ there is only one mapping $r : V \to \{q\}$. This mapping is an R^--run iff no c-labeled event covers an a-labeled one. Indeed, if we have $x \longrightarrow\!\!< y$ with $\lambda(x) = a$ and $\lambda(y) = c$, then $\delta_{\lambda(y),\mathrm{R}^-(y)}((q_i)_{i\in\mathrm{R}^-(y)}) = \emptyset$ and therefore r does not satisfy the condition for an R^--run.

Note that a $\vec{\Sigma}$-pomset has an R^+-run of this automaton iff it does not contain an a-labeled event below some c-labeled one.

Example 3.3 Let $n = 4$, $\Sigma_1 = \{a\}$, $\Sigma_2 = \{b\}$, $\Sigma_3 = \{c\}$ and $\Sigma_4 = \{d\}$. The automaton is given by $Q_1 = \{0, 1\}$, $Q_2 = Q_3 = Q_4 = \{0\}$, and

$$\begin{aligned}
\delta_{a,J}((q_i)_{i\in J}) &= \begin{cases} \{(q_1+1) \mod 2\} & \text{if } 1 \in J \\ \{1\} & \text{otherwise} \end{cases} \\
\delta_{b,J}((q_i)_{i\in J}) &= \{0\} \\
\delta_{c,J}((q_i)_{i\in J}) &= \{0\} \\
\delta_{d,J}((q_i)_{i\in J}) &= \begin{cases} \{0\} & \text{if } q_1 = 0 \text{ or } 1 \notin J \\ \emptyset & \text{otherwise.} \end{cases}
\end{aligned}$$

This automaton is meant to run in the R^+-mode. Note that in this case the first process simply counts its events modulo 2. The second and third process do “nothing”. Now consider the fourth process. It allows a transition as long as no event from the first process occurred. Once such an a-labeled event occurred, it only allows a transition if the state on the last such event is 0. Thus, the automaton cannot proceed if some d-labeled event dominates an odd number of a-labeled events. Since this is the only case where it cannot proceed, this $\vec{\Sigma}$-ACA allows an R^+-run on a $\vec{\Sigma}$-pomset iff any d-labeled event dominates an even number of a-labeled ones.

In the R^--mode, the first process does not count the occurrences of a-labeled events modulo 2 because it restarts with 1 whenever some a-labeled event does not directly cover another such a-labeled element.

Similarly, there are two possibilities to define when a run is successful. The first one is to consider all local final states. Alternatively, we may restrict our attention to the states that correspond to the maximal elements of the pomset in consideration. More formally, let $\mathcal{A}$ be a $\vec{\Sigma}$-ACA with final states F and let t be a $\vec{\Sigma}$-pomset. Then $\mathrm{F}^+(t) := p \circ \lambda(t)$ denotes the set of local processes that perform at least one step when t is executed. The set $\mathrm{F}^-(t) := p \circ \lambda(\max(t))$ comprises those local processes that perform a maximal event. For $\alpha, \beta \in \{+, -\}$, an R^α-run r is F^β-successful iff

$$r(\partial_i(t))_{i\in\mathrm{F}^\beta(t)} \in F.$$

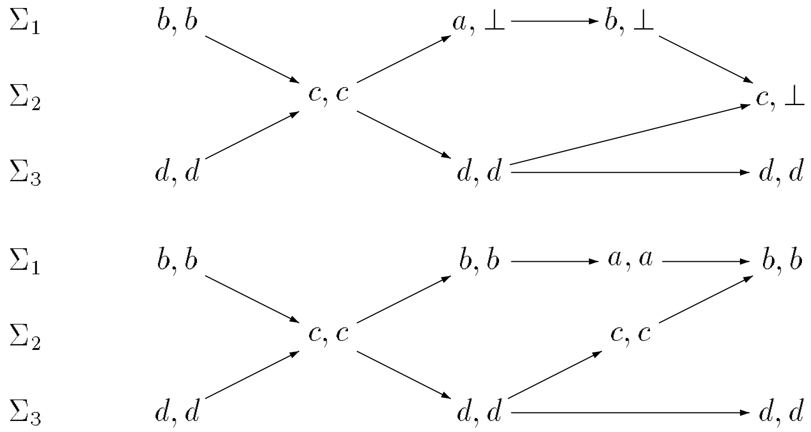
Example 3.4 Our next example is a deterministic $\vec{\Sigma}$-ACA which accepts precisely the set of $\vec{\Sigma}$-pomsets satisfying condition (2) of Section 2.2. More precisely, let (Σ, D) be a dependence alphabet with $\Sigma = \Sigma_1 \dot\cup \cdots \dot\cup \Sigma_n$ where each Σ_i is a clique of (Σ, D). We define the $\vec{\Sigma}$-ACA $\mathcal{A} = ((Q_i)_{i\in[n]}, (\delta_{a,J})_{a\in\Sigma, J\subseteq[n]}, F)$ where $Q_i = \Sigma_i \cup \{\bot\}$ for all $i \in [n]$ and

$$\delta_{a,J}((q_j)_{j\in J}) = \begin{cases} \{a\} & \text{if } q_j \neq \bot \text{ and } (a, q_j) \in D \text{ for all } j \in J \\ \{\bot\} & \text{otherwise} \end{cases}$$

for all $a \in \Sigma$ and $J \subseteq [n]$. Finally, the set of accepting states is $F = \bigcup_{J \subseteq [n]} \prod_{i \in J} \Sigma_i$.

In an R^--run of this automaton, each process remembers the last action performed and therefore is able to check that an event covers only dependent events.

One can easily check that $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A})$ is the set of $\vec{\Sigma}$-pomsets $(V, \leq, \lambda)$ such that for all $u, v \in V$, if $u \multimap\!\!< v$ then $(\lambda(u), \lambda(v)) \in D$. For instance, if $(\Sigma, D) = a \longrightarrow b \longrightarrow c \longrightarrow d$ with $\Sigma_1 = \{a, b\}$, $\Sigma_2 = \{c\}$ and $\Sigma_3 = \{d\}$, we give below a rejecting run and an accepting run of $\mathcal{A}$. In this picture, each vertex v is labeled by the pair $(\lambda(v), r(v))$. Note that, in order to obtain the states of minimal vertices, we apply transition functions of the form $\delta_{\lambda(v),\emptyset}$.



Example 3.5 Later, we will see that for any dependence alphabet (Σ, D), it is possible to construct a nondeterministic ACA $\mathcal{A}$ that accepts precisely the traces over (Σ, D). Here, we give such an automaton for the simple dependence alphabet $(\Sigma, D) = a \longrightarrow b \longrightarrow c$. We consider the processes $\Sigma_1 = \{a\}$, $\Sigma_2 = \{b\}$ and $\Sigma_3 = \{c\}$. The sets of states of the $(\Sigma_1, \Sigma_2, \Sigma_3)$-ACA $\mathcal{A}$ are $Q_1 = Q_2 = Q_3 = \{a, b\} \times \{b, c\}$ and all possible combinations of states are accepting. Intuitively, a state (x, y) claims that the next event labeled by a or by b is actually labeled by x (and similarly the next event labeled by c or by b is actually labeled by y). The transition functions are the following (we only give the non empty transitions):

$$\begin{aligned}
\delta_{b,\emptyset} = \delta_{b,\{1\}}((b,b)) = \delta_{b,\{2\}}((b,b)) &= \delta_{b,\{3\}}((b,b)) \\
&= \delta_{b,\{1,3\}}((b,c),(a,b)) = \{(b,b),(a,b),(b,c),(a,c)\} \\
\delta_{a,\{1\}}((a,b)) &= \delta_{a,\{2\}}((a,b)) = \{(a,b),(b,b)\} \\
\delta_{a,\{1\}}((a,c)) &= \delta_{a,\{2\}}((a,c)) = \{(a,c),(b,c)\} \\
\delta_{c,\{2\}}((b,c)) &= \delta_{c,\{3\}}((b,c)) = \{(b,c),(b,b)\} \\
\delta_{c,\{2\}}((a,c)) &= \delta_{c,\{3\}}((a,c)) = \{(a,c),(a,b)\}
\end{aligned}$$

Here is an R^--run of this automaton:

$$\begin{tikzcd}
 & a,(a,c) \arrow[r] & a,(b,c) \arrow[dr] & & & & \\
b,(a,c) \arrow[ur] \arrow[dr] & & & b,(b,b) \arrow[r] & b,(b,c) \arrow[dr] & & b,(a,b) \\
 & c,(a,b) \arrow[urr] & & & & c,(b,b) \arrow[ur] &
\end{tikzcd}$$

It is easy to see that all traces starting with b admit an F^--successful R^--run of $\mathcal{A}$. Although less trivial, the converse is also true. Therefore, in the mode $\mathrm{R}^-\mathrm{F}^-$, this automaton accepts the set of traces starting with b. By changing the initial condition of the automaton, we can recognize all traces starting with a or with c or with a and c. For instance, if we set $\delta_{b,\emptyset} = \delta_{c,\emptyset} = \emptyset$ and $\delta_{a,\emptyset} = \{(a,b),(b,b)\}$ we accept all traces starting with a.

Now let $\mathbb{P} \subseteq \mathbb{P}(\vec{\Sigma})$ be some set of $\vec{\Sigma}$-pomsets and let $\alpha, \beta \in \{+,-\}$. For a $\vec{\Sigma}$-ACA $\mathcal{A}$, the *language of pomsets from* $\mathbb{P}$ *accepted by* $\mathcal{A}$ *in the mode* $\mathrm{R}^\alpha\mathrm{F}^\beta$, denoted by $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A},\mathbb{P})$, is the set of all $\vec{\Sigma}$-pomsets $t \in \mathbb{P}$ that admit an F^β-successful R^α-run. Then $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathbb{P})$ denotes the set of all languages $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A},\mathbb{P})$ for some $\vec{\Sigma}$-ACA $\mathcal{A}$. The set $\mathrm{dR}^\alpha\mathrm{F}^\beta(\mathbb{P})$ contains all languages $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A},\mathbb{P})$ for some *deterministic* $\vec{\Sigma}$-ACA $\mathcal{A}$. Often, we will abbreviate $(\mathrm{d})\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A},\mathbb{P}(\vec{\Sigma}))$ by $(\mathrm{d})\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A})$.

We conclude this section with a few remarks. First, the covering of pomsets by the chains formed by the fixed sequential processes is crucial in the definition of asynchronous cellular automata. It allows us to use a fixed number of local states and to determine the read and write domains of the actions using the labeling and the order relation. The weakest covering is when each Σ_i is a singleton. In this case we have a set of local states per letter as in the asynchronous cellular automata for traces [Zie89, CMZ93]. Note that, even with this trivial covering, our definition is not the same as that of Zielonka for traces. Mainly, in our definition, a run of the ACA is over the Hasse diagram (over the directed acyclic graph (V,E,λ), respectively) of the pomset whereas with Zielonka's ACA for traces, a run is in fact over the dependence graph of the trace. A dependence graph is an intermediary representation of a trace between its Hasse diagram and its directed acyclic graph (V,E,λ). This intermediary representation is possible thanks to the existence of a static dependence relation over actions. More precisely, our definition of ACA for pomsets and that of Zielonka for traces differ in three respects. First, Zielonka's definition uses a global initial state which in our case is coded in the transition functions of the form $\delta_{a,\emptyset}$. Second, the read domains in our definition depend on the actual pomset whereas in Zielonka's definition a fixed set of processes is read even if the last executions of some of these processes are far below the current action. Third (especially in the acceptance mode F^-), we do not necessarily read the final states of all local processes to determine whether a run is successful whereas in Zielonka's definition the states of all processes are collected globally to decide acceptance.

4 ACAs on general pomsets - positive results

4.1 The acceptance mode

First we show that changing the acceptance mode between $-$ and $+$ preserves the expressive power of nondeterministic asynchronous cellular automata. Our constructions yield a nondeterministic automaton even if we start with a deterministic one. Later, in Proposition 6.4, we will see that in general for a deterministic ACA there is no deterministic automaton that accepts the same language in the other acceptance mode, i.e. that the following theorem does not hold for deterministic ACAs.

Theorem 4.1 *Let $\alpha \in \{+,-\}$. Then $\mathrm{R}^\alpha\mathrm{F}^-(\mathbb{P}(\vec{\Sigma})) = \mathrm{R}^\alpha\mathrm{F}^+(\mathbb{P}(\vec{\Sigma}))$, i.e. for any $\vec{\Sigma}$-ACA $\mathcal{A}$ there exist nondeterministic $\vec{\Sigma}$-ACAs $\mathcal{A}_1$ and $\mathcal{A}_2$ such that*

$$\mathrm{R}^\alpha\mathrm{F}^-(\mathcal{A},\mathbb{P}(\vec{\Sigma})) = \mathrm{R}^\alpha\mathrm{F}^+(\mathcal{A}_1,\mathbb{P}(\vec{\Sigma})) \text{ and } \mathrm{R}^\alpha\mathrm{F}^+(\mathcal{A},\mathbb{P}(\vec{\Sigma})) = \mathrm{R}^\alpha\mathrm{F}^-(\mathcal{A}_2,\mathbb{P}(\vec{\Sigma})).$$

Proof. First, we construct the $\vec{\Sigma}$-ACA $\mathcal{A}_1$. This $\vec{\Sigma}$-ACA $\mathcal{A}_1$ will simulate the run of $\mathcal{A}$ on some pomset t and additionally will guess the maximal node of t for each process. To do this, along a successful R^α-run of $\mathcal{A}_1$ on t, any local process i nondeterministically picks one and only one node x with $p \circ \lambda(x) = i$. This node sends a signal End_i upwards.

To check this guess, whenever a node y with $p \circ \lambda(y) = i$ receives the signal End_i, the automaton stops, i.e. no successor state is defined and therefore the automaton is forced to reject. Note that $i \in \mathrm{F}^+(t)$ is actually in $\mathrm{F}^-(t)$ iff there is no node z with $p \circ \lambda(z) \neq i$ that received the signal End_i. Hence, the automaton $\mathcal{A}_1$ can compute the tuple of final maximal states of the run of $\mathcal{A}$ from its tuple of final states and accept or reject the pomset accordingly.

Now we construct the $\vec{\Sigma}$-ACA $\mathcal{A}_2$. The idea is as follows: Similarly to the construction above, $\mathcal{A}_2$ simulates $\mathcal{A}$. Additionally, any local process $i \in [n]$ guesses its maximal node and sends a signal (End_i, p_i) upwards where p_i is the current state of process i. This signal is forwarded upward by all transitions. The ACA $\mathcal{A}_2$ stops if a node of process i receives the signal End_i since this means that the guessed node was not maximal in its chain. By reading the maximal states of the run of $\mathcal{A}_2$ on t we can now recover the final state of each process in $\mathrm{F}^+(t)$ and accept or reject the pomset according to the acceptance condition of $\mathcal{A}$. $\square$

4.2 Asynchronous mapping

Asynchronous mappings were introduced in [CMZ93] in order to simplify the construction of ACAs for traces. Here we generalize this notion to $\vec{\Sigma}$-pomsets. The domain of an asynchronous mapping must be a prefix closed subset of the set of $\vec{\Sigma}$-pomsets, that is a subset $\mathbb{Q}$ of $\vec{\Sigma}$-pomsets such that if some $\vec{\Sigma}$-pomset s is a prefix of $t \in \mathbb{Q}$ then $s \in \mathbb{Q}$. For instance, $\mathbb{P}(\vec{\Sigma})$ and $\mathbb{M}(\Sigma, D)$ are prefix closed sets of $\vec{\Sigma}$-pomsets.

Definition 4.2 *Let $\mathbb{Q}$ be a prefix closed set of $\vec{\Sigma}$-pomsets and let S be a finite set. A mapping $\sigma : \mathbb{Q} \to S$ is* asynchronous *if for all $t = (V, \leq, \lambda) \in \mathbb{Q}$,*

1. *for all vertices $x \in V$, the value $\sigma(\downarrow x)$ is uniquely determined by $\sigma(\Downarrow x)$ and $\lambda(x)$.*

2. *for all $A, B \subseteq \Sigma$, the value $\sigma(\partial_{A\cup B}(t))$ is uniquely determined by $\sigma(\partial_A(t))$ and $\sigma(\partial_B(t))$.*

A language $L \subseteq \mathbb{Q}$ is recognized by an asynchronous mapping *$\sigma : \mathbb{Q} \to S$ whenever $L = \sigma^{-1}(\sigma(L))$.*

Proposition 4.3 *Let $\mathbb{Q} \subseteq \mathbb{P}(\vec{\Sigma})$ be a prefix closed set of $\vec{\Sigma}$-pomsets and $\alpha, \beta \in \{+, -\}$. Let $L \subseteq \mathbb{Q}$ be a language of $\vec{\Sigma}$-pomsets recognized by some asynchronous mapping $\sigma : \mathbb{Q} \to S$. Then there exists a deterministic $\vec{\Sigma}$-asynchronous cellular automaton $\mathcal{A}$ such that $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A}, \mathbb{Q}) = L$.*

Proof. In this proof, we consider only the case $\alpha = \beta = -$. For the other modes, the proof remains essentially the same. One only has to change accordingly the definitions of the transition functions and of the final set. This is left to the reader.

Our proof follows the same ideas as the corresponding one for traces. Assume that $\sigma : \mathbb{Q} \to S$ recognizes the language $L \subseteq \mathbb{Q}$. We define a deterministic $\vec{\Sigma}$-ACA $\mathcal{A}$ as follows:

1. For all $i \in [n]$, let $Q_i = S$,

2. For all $a \in \Sigma$, $J \subseteq [n]$ and $(q_i)_{i\in J} \in S^J$, let

$$\begin{aligned} \delta_{a,J}\left((q_i)_{i\in J}\right) &= \{\sigma(t) \mid t \in \mathbb{Q}, t = \downarrow x \text{ for some } x \text{ such that} \\ &\qquad \lambda(x) = a, \mathrm{R}^-(x) = J \text{ and } \sigma(\partial_{\Sigma_i}(\Downarrow x)) = q_i \text{ for all } i \in J\} \end{aligned}$$

3. $F = \{\sigma(\partial_{\Sigma_i}(t))_{i\in \mathrm{F}^-(t)} \mid t \in L\}$.

Claim 1. $\mathcal{A}$ is deterministic.

Indeed, let $a \in \Sigma$, $J \subseteq [n]$ and $(q_i)_{i\in J} \in S^J$. Choose $t = \downarrow x$ and $t' = \downarrow x'$ in $\mathbb{Q}$ with $\lambda(x) = \lambda(x')$, $\mathrm{R}^-(x) = J = \mathrm{R}^-(x')$ and $\sigma(\partial_{\Sigma_i}(\Downarrow x)) = q_i = \sigma(\partial_{\Sigma_i}(\Downarrow x'))$ for all $i \in J$. We have $\Downarrow x = \partial_{(\cup_{i\in J}\Sigma_i)}(\Downarrow x)$ and $\Downarrow x' = \partial_{(\cup_{i\in J}\Sigma_i)}(\Downarrow x')$. Hence, using the definition of asynchronous mappings, we deduce $\sigma(\Downarrow x) = \sigma(\Downarrow x')$ and since $\lambda(x) = \lambda(x')$ it follows that $\sigma(t) = \sigma(\downarrow x) = \sigma(\downarrow x') = \sigma(t')$ which proves the claim.

Claim 2. Let $t = (V, \leq, \lambda) \in \mathbb{Q}$ be a $\vec{\Sigma}$-pomset. Then, the mapping $r : V \to \bigcup_{i\in[n]} Q_i$ defined by $r(x) = \sigma(\downarrow x)$ is the R^--run of $\mathcal{A}$ on t.

One only has to check that $r(x) \in \delta_{\lambda(x),\mathrm{R}^-(x)}(r(\partial_i(\Downarrow x))_{i\in\mathrm{R}^-(x)})$ for any $x \in V$. But this follows directly from the definition of the transition functions of $\mathcal{A}$.

Claim 3. $\mathrm{R}^-\mathrm{F}^-(\mathcal{A},\mathbb{Q}) = L$.

Note first that $t = \partial_{(\cup_{i\in \mathrm{F}^-(t)}\Sigma_i)}(t)$ for all $t \in \mathbb{Q}$. Now assume that $t \in \mathrm{R}^-\mathrm{F}^-(\mathcal{A})$. Then for the unique run r of $\mathcal{A}$ on t, we have $r(\partial_i(t))_{i\in \mathrm{F}^-(t)} \in F$ and there exists $t' \in L$ such that $\mathrm{F}^-(t) = \mathrm{F}^-(t')$ and $\sigma(\partial_{\Sigma_i}(t)) = \sigma(\partial_{\Sigma_i}(t'))$ for all $i \in \mathrm{F}^-(t) = \mathrm{F}^-(t')$. Since σ is asynchronous, it follows that $\sigma(t) = \sigma(t')$. Therefore, $t \in \sigma^{-1}(\sigma(L)) = L$ which proves one inclusion. The converse is trivial. $\square$

Note that, for trace languages, the converse of Proposition 4.3 is also true implying that all alternative modes of ACAs are equivalent for traces. Indeed, it is easy to show that a trace language accepted by an ACA is a recognizable trace language, whatever mode is chosen for accepting runs. Moreover, if L is a recognizable trace language, the existence of an asynchronous mapping which recognizes L was proven in [CMZ93]. Finally, by Proposition 4.3, from this asynchronous mapping one can easily get for each mode an ACA which accepts L.

The equivalence between alternative definitions of accepting runs will be extended to a more general class of pomsets in Section 5. Section 6 will deal with a class of pomsets where the equivalence for nondeterministic ACAs remains true while it will not hold for deterministic ACAs. In Section 7 we will show that even the equivalence for nondeterministic ACAs does not hold for the class of all $\vec{\Sigma}$-pomsets. In addition, in the general setting of $\mathbb{P}(\vec{\Sigma})$, the converse of Proposition 4.3 is false (cf. discussion after Proposition 6.4).

4.3 From ACA to MSO

In this section, we will define monadic second order (MSO) formulas and their interpretations over pomsets. We will then prove that for all ACAs $\mathcal{A}$ (deterministic or not), there exists an MSO formula which defines the language accepted by $\mathcal{A}$.

Let Σ be a finite alphabet. Formulas of the MSO language over Σ that we consider involve first order variables $x, y, z \ldots$ for vertices and monadic second order variables $X, Y, Z, \ldots$ for sets of vertices. They are built up from the atomic formulas $\lambda(x) = a$ for $a \in \Sigma$, $x \leq y$, and $x \in X$ by means of the boolean connectives $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ and quantifiers $\exists, \forall$ (both for first order and for second order variables). If we use the quantifiers $\exists$ and $\forall$ only for first-order variables, we obtain formulas of first-order logic. Formulas without free variables are called sentences. For instance, the following formulas are first order and monadic second order sentences respectively.

$$\begin{aligned}
\varphi_1 \; &::= \; \exists x(\lambda(x) = a \wedge \forall y(x \leq y \rightarrow \neg\lambda(y) = b)) \\
\varphi_2 \; &::= \; \exists X \exists Y(\forall x(x \in X \vee x \in Y) \wedge \exists x\ (x \in X) \wedge \exists y\ (y \in Y) \\
&\qquad \wedge \forall x \forall y(x \in X \wedge y \in Y \rightarrow \neg x \leq y \wedge \neg y \leq x))
\end{aligned}$$

The satisfaction relation $\models$ between pomsets $t = (V, \leq, \lambda)$ and a sentence φ of the monadic second order logic is defined canonically with the understanding that first order variables range over the vertices of V and second order variables over subsets of V. The set of pomsets which satisfy a sentence φ is denoted by $L(\varphi)$. For instance, $L(\varphi_1)$ is the set of pomsets which have a vertex labeled by a with no vertex labeled by b above and $L(\varphi_2)$ is the set of non connected pomsets.

In order to make the formulas more readable, we will use several abbreviations which can be easily translated in our MSO language. For instance, we will write

$$\begin{aligned}
x < y &\quad \text{for} \quad x \leq y \wedge \neg y \leq x \\
x \longrightarrow\!\!\!< y &\quad \text{for} \quad x < y \wedge \neg \exists z (x < z \wedge z < y) \\
\lambda(x) \in A &\quad \text{for} \quad \bigvee_{a \in A} \lambda(x) = a \\
p \circ \lambda(x) = p \circ \lambda(y) &\quad \text{for} \quad \bigvee_{1 \leq i \leq n} (\lambda(x) \in \Sigma_i \wedge \lambda(y) \in \Sigma_i) \\
X \cap Y = \emptyset &\quad \text{for} \quad \neg \exists x (x \in X \wedge x \in Y)
\end{aligned}$$

Note that the language defined by a formula can contain pomsets with auto-concurrency (concurrent vertices with the same label). We do not need to put restrictions on the pomsets defined by a formula because all restrictions we need can be expressed by MSO (or even first-order) formulas. For instance the set $\mathbb{P}(\vec{\Sigma})$ of $\vec{\Sigma}$-pomsets is defined by the formula

$$\varphi_{\vec{\Sigma}} ::= \forall x \forall y (p \circ \lambda(x) = p \circ \lambda(y) \rightarrow (x \leq y \vee y \leq x))$$

and the set $\mathbb{M}(\Sigma, D)$ of traces over a dependence alphabet is defined by the formula

$$\forall x \forall y \left([(\lambda(x), \lambda(y)) \in D \rightarrow (x \leq y \vee y \leq x)] \wedge [x \longrightarrow\!\!\!< y \rightarrow (\lambda(x), \lambda(y)) \in D] \right)$$

where $(\lambda(x), \lambda(y)) \in D$ stands for the formula $\bigvee_{(a,b) \in D} (\lambda(x) = a \wedge \lambda(y) = b)$.

We are now ready to state

Theorem 4.4 *Let $\mathcal{A}$ be a possibly nondeterministic $\vec{\Sigma}$-ACA and let $\alpha, \beta \in \{+, -\}$. There exists an existential monadic second order sentence φ over $\vec{\Sigma}$ such that*

$$L(\varphi) = \mathrm{R}^\alpha \mathrm{F}^\beta (\mathcal{A}, \mathbb{P}(\vec{\Sigma})).$$

Proof. Let $\mathcal{A} = ((Q_i)_{i \in [n]}, (\delta_{a,J})_{a \in \Sigma, J \subseteq [n]}, F)$ be a $\vec{\Sigma}$-ACA. We will construct an MSO sentence which will be satisfied exactly by those $\vec{\Sigma}$-pomsets accepted by $\mathcal{A}$ in the mode $\mathrm{R}^-\mathrm{F}^-$. For the other modes, the proof is essentially the same: one only has to change the formulas *transition* and *accepted* accordingly. Let k be the number of states in $\bigcup_{i \in [n]} Q_i$. We may assume that $\bigcup_{i \in [n]} Q_i = [k] = \{1, \ldots, k\}$. The following formula claims the existence of an F^--successful R^--run of the automaton.

$$\psi ::= \exists X_1 \ldots \exists X_k \left(\text{partition}(X_1, \ldots, X_k) \wedge (\forall x \; \text{transition}(x)) \wedge \text{accepted} \right)$$

We will now explain this formula and give the sub-formulas *partition*, *transition* and *accepted*. An R^--run over a $\vec{\Sigma}$-pomset $t = (V, \leq, \lambda)$ is coded by the MSO variables

$X_1, \ldots, X_k$. More precisely, X_i stands for the set of vertices mapped on the state i by the R^--run. The formula partition$(X_1, \ldots, X_k)$ makes sure that the MSO variables $X_1, \ldots, X_k$ describe a mapping from V to $\bigcup_{i\in[n]} Q_i$.

$$\text{partition}(X_1, \ldots, X_k) ::= \left(\forall x \bigvee_{i\in[k]} x \in X_i\right) \wedge \left(\bigwedge_{1\leq i<j\leq k} X_i \cap X_j = \emptyset\right)$$

Then, we have to claim that this labeling of vertices by states agrees with the transition functions of the automaton.

$$\begin{aligned}\text{transition}(x) ::= \bigvee_{q\in\delta_{a,J}((q_i)_{i\in J})} \Big(&\lambda(x) = a \wedge x \in X_q \wedge \forall y\ (y \longrightarrow\!\!\!< x \rightarrow p\circ\lambda(y) \in J)\\ &\wedge \bigwedge_{i\in J} \exists y\ (y \longrightarrow\!\!\!< x \wedge p\circ\lambda(y) = i \wedge y \in X_{q_i})\Big)\end{aligned}$$

where the disjunction ranges over all letters $a \in \Sigma$, states $q \in Q_{p(a)}$, subsets $J \subseteq [n]$ and tuples $(q_i)_{i\in J} \in \prod_{i\in J} Q_i$ such that $q \in \delta_{a,J}((q_i)_{i\in J})$.

It remains to state that the R^--run reaches a final state of the automaton.

$$\begin{aligned}\text{accepted} ::= \bigvee_{(f_i)_{i\in J}\in F} \Big(&\forall x\ ((\neg\exists y\ x < y) \rightarrow p\circ\lambda(x) \in J)\\ &\wedge \bigwedge_{i\in J} \exists x\big((\neg\exists y\ x<y) \wedge p\circ\lambda(x) = i \wedge x \in X_{f_i}.\big)\Big)\end{aligned}$$

In fact, the formula ψ describes an F^--accepting R^--run of the automaton only for $\vec{\Sigma}$-pomsets. Therefore, we need in addition the formula $\varphi_{\vec{\Sigma}}$ described above. Finally, the theorem follows from

$$L(\varphi_{\vec{\Sigma}} \wedge \psi) = \mathrm{R}^-\mathrm{F}^-(\mathcal{A}).$$

□

The converse of the theorem above does not hold as we will see by Theorem 7.3 and Proposition 7.4.

5 CROW-pomsets

In this section, we prove that the converse of Theorem 4.4 holds for the special subclass of $\vec{\Sigma}$-pomsets which satisfy the CROW axiom defined below.

Definition 5.1 *A $\vec{\Sigma}$-pomset $t = (V, \leq, \lambda)$ satisfies the* Concurrent Read and Exclusive Owner Write (CROW) *axiom if for all $x, y, z \in V$,*

$$x \longrightarrow\!\!\!< y, x < z \text{ and } y \parallel z \quad\Longrightarrow\quad p\circ\lambda(x) \neq p\circ\lambda(z).$$

The set of $\vec{\Sigma}$-pomsets which satisfy the CROW axiom is denoted by $\mathbb{CROW}(\vec{\Sigma})$.

A possible interpretation of this axiom is to think of the ACA as a Concurrent Read and Exclusive Owner Write (CROW) machine. More precisely, we consider n processes whose sets of actions are $\Sigma_1, \ldots, \Sigma_n$ respectively. Each process has a memory which can be read by all actions but can be written by its own actions only (Owner Write). We allow concurrent reads of memories but no concurrent writes. As mentioned in Section 2.1, this restriction is already enforced by the very definition of $\vec{\Sigma}$-pomsets. Without further restrictions, two concurrent events may respectively read from and write to the same location. This is the case when there exist two concurrent events $y \parallel z$ such that z writes in the memory of some process i ($p \circ \lambda(z) = i$) and y reads the memory of this process i ($p \circ \lambda(x) = i$ for some $x \longrightarrow\!\!\!< y$). This is precisely the situation which is forbidden by the CROW axiom.

Theorem 5.2 *Let φ be an MSO sentence over $\vec{\Sigma}$ and let $\beta \in \{+, -\}$. There exists a deterministic $\vec{\Sigma}$-ACA $\mathcal{A}$ such that*

$$L(\varphi) \cap \mathbb{CROW}(\vec{\Sigma}) = \mathrm{R}^{-}\mathrm{F}^{\beta}(\mathcal{A}, \mathbb{CROW}(\vec{\Sigma})).$$

In order to prove this theorem, one can use an induction on the structure of the formula. Disjunction and existential quantification are easily dealt with when nondeterministic ACAs are allowed. On the other hand, complement is easy for deterministic ACAs. Whence the core of such an approach is the determinization of ACAs. For this problem, starting from a nondeterministic ACA $\mathcal{A}$, one can directly construct an asynchronous mapping which accepts the language $\mathrm{R}^{-}\mathrm{F}^{\beta}(\mathcal{A}, \mathbb{CROW}(\vec{\Sigma}))$ and then use Proposition 4.3. This construction is similar to that of [Mus96] and uses the asynchronous time stamping ν of Cori, Métivier & Zielonka [CMZ93] but the proofs are more involved. In particular, it is known that for traces the mapping ν is asynchronous by itself [CMZ93, DM95] but this is not the case for CROW-pomsets. Here we give a simpler proof which uses Zielonka's theorem. For this, we first map CROW-pomsets into traces by simply changing the labeling.

Let $\Sigma' = \Sigma \times \mathcal{P}([n])$ be a new set of labels and for all $i \in [n]$, let $\Sigma'_i = \Sigma_i \times \mathcal{P}([n])$ be the associated new processes. By a slight abuse of notation, let p again be the mapping that associates with any $(a, M) \in \Sigma'$ the process i with $(a, M) \in \Sigma'_i$. This is justified since $(a, M) \in \Sigma'_i$ iff $a \in \Sigma_i$, i.e. $p(a, M) = p(a)$. Intuitively, the second component of a label in Σ' stands for the read domain of the action. We define an embedding g from $\mathbb{P}(\vec{\Sigma})$ into $\mathbb{P}(\vec{\Sigma'})$ by $g(V, \leq, \lambda) = (V, \leq, \lambda')$ where for all $x \in V$, $\lambda'(x) = (\lambda(x), \mathrm{R}^{-}(x))$. Note that g is well defined, since for all $i \in [n]$, $\lambda'^{-1}(\Sigma'_i) = \lambda^{-1}(\Sigma_i)$ is totally ordered. Let D' be the dependence relation defined on Σ' by

$$D' = \{((a, A), (b, B)) \mid p(a) = p(b) \vee p(a) \in B \vee p(b) \in A\}.$$

Hence, two actions are dependent if either they both write in the same process, or one reads the process written by the other.

Proposition 5.3

$$\mathbb{CROW}(\vec{\Sigma}) = g^{-1}(\mathbb{M}(\Sigma', D'))$$

Proof. We first prove that $\mathbb{CROW}(\vec{\Sigma}) \subseteq g^{-1}(\mathbb{M}(\Sigma', D'))$. Let $t = (V, \leq, \lambda)$ be a CROW-pomset from $\mathbb{CROW}(\vec{\Sigma})$ and let $g(t) = (V, \leq, \lambda')$. Let $x, y \in V$ and assume that $x -\!\!< y$. Then, $p \circ \lambda(x) \in \mathrm{R}^-(y)$ and it follows that $(\lambda'(x), \lambda'(y)) \in D'$. Now, let $y, z \in V$ and assume that $(\lambda'(y), \lambda'(z)) \in D'$. If $p \circ \lambda(y) = p \circ \lambda(z)$ then $y \not\parallel z$ since t is a $\vec{\Sigma}$-pomset. If $p \circ \lambda(y) \neq p \circ \lambda(z)$, we have for instance $p \circ \lambda(z) \in \mathrm{R}^-(y)$. Hence, there exists $x \in V$ such that $x -\!\!< y$ and $p \circ \lambda(x) = p \circ \lambda(z)$. Therefore, x and z must be ordered. Since $x < z$ would contradict the CROW-axiom, it follows $z \leq x$, whence $z < y$. Therefore, $g(t) \in \mathbb{M}(\Sigma', D')$.

Conversely, let $t = (V, \leq, \lambda) \in g^{-1}(\mathbb{M}(\Sigma', D'))$ and let $g(t) = (V, \leq, \lambda')$. Let $x, y, z \in V$ be such that $x -\!\!< y$, $x < z$ and $y \parallel z$. By definition, $p \circ \lambda(x) \in \mathrm{R}^-(y)$ and $(\lambda'(y), \lambda'(z)) \notin D'$. Therefore, $p \circ \lambda(z) \notin \mathrm{R}^-(y)$ and it follows that $p \circ \lambda(x) \neq p \circ \lambda(z)$. $\square$

Proposition 5.4 *Let φ be an MSO sentence over Σ. There exists an MSO sentence φ' over Σ' such that*

$$L(\varphi) \cap \mathbb{CROW}(\vec{\Sigma}) = g^{-1}(L(\varphi') \cap \mathbb{M}(\Sigma', D'))$$

Proof. Let φ' be the MSO sentence over Σ' obtained from φ by substituting for atomic formulas of the form $\lambda(x) = a$ the disjunction $\bigvee_{J \subseteq [n]} \lambda'(x) = (a, J)$:

$$\varphi' = \varphi\left[\bigvee_{J \subseteq [n]} \lambda'(x) = (a, J) \Big/ \lambda(x) = a\right].$$

Let $t = (V, \leq, \lambda) \in L(\varphi) \cap \mathbb{CROW}(\vec{\Sigma})$. We have $g(t) = (V, \leq, \lambda') \in \mathbb{M}(\Sigma', D')$ by Proposition 5.3 and it remains to show that $g(t) \models \varphi'$. This is clear since $\lambda(x) = a$ if and only if $\lambda'(x) = (a, J)$ for some $J \subseteq [n]$. The converse can be shown similarly. $\square$

Proposition 5.5 *Let $\mathcal{A}'$ be a (deterministic) $\vec{\Sigma}'$-ACA and $\beta \in \{+, -\}$. There exists a (deterministic) $\vec{\Sigma}$-ACA $\mathcal{A}$ such that $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma})) = g^{-1}(\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}', \mathbb{P}(\vec{\Sigma}')))$.*

Proof. Let $\mathcal{A}' = ((Q_i)_{i \in [n]}, (\delta'_{a',J})_{a' \in \Sigma', J \subseteq [n]}, F)$ be a $\vec{\Sigma}'$-ACA. For all $a \in \Sigma$ and $J \subseteq [n]$, let $\delta_{a,J} = \delta'_{(a,J),J}$. We claim that the automaton $\mathcal{A} = ((Q_i)_{i \in [n]}, (\delta_{a,J})_{a \in \Sigma, J \subseteq [n]}, F)$ is the required $\vec{\Sigma}$-ACA. Note that if $\mathcal{A}'$ is deterministic then so is $\mathcal{A}$.

We first show that in order to accept a pomset in $g(\mathbb{P}(\vec{\Sigma}))$ the ACA $\mathcal{A}'$ only uses transition functions of the form $\delta'_{(a,J),J}$. Indeed, let $t = (V, \leq, \lambda) \in \mathbb{P}(\vec{\Sigma})$ and let $g(t) = (V, \leq, \lambda')$. Then $\mathrm{R}^-(x) = p \circ \lambda'(\{y \in V \mid y -\!\!< x\}) = p \circ \lambda(\{y \in V \mid y -\!\!< x\})$ for all $x \in V$. Therefore, in a run of $\mathcal{A}'$ on $g(t)$ the transition functions used are of the form $\delta'_{\lambda'(x),\mathrm{R}^-(x)} = \delta'_{(\lambda(x),\mathrm{R}^-(x)),\mathrm{R}^-(x)} = \delta_{\lambda(x),\mathrm{R}^-(x)}$.

It follows that a mapping $r : V \to \bigcup_{i \in [n]} Q_i$ is an F^β-successful R^--run of $\mathcal{A}'$ on $g(t)$ if and only if it is an F^β-successful R^--run of $\mathcal{A}$ on t, that is,

$$t \in \mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma})) \iff g(t) \in \mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}', \mathbb{P}(\vec{\Sigma}')) \iff t \in g^{-1}(\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}', \mathbb{P}(\vec{\Sigma}'))).$$

The proposition follows. $\square$

Proof of Theorem 5.2 Let φ be an MSO sentence over Σ. By Proposition 5.4, there exists an MSO sentence φ' over Σ' such that

$$L(\varphi) \cap \mathbb{CROW}(\vec{\Sigma}) = g^{-1}(L(\varphi') \cap \mathbb{M}(\Sigma', D')).$$

The language $L(\varphi') \cap \mathbb{M}(\Sigma', D')$ is a recognizable trace language [Tho90]. Hence by [CMZ93], there exists an asynchronous mapping σ from $\mathbb{M}(\Sigma', D')$ into a finite set which recognizes $L(\varphi') \cap \mathbb{M}(\Sigma', D')$. By Proposition 4.3, there exists a deterministic $\vec{\Sigma'}$-ACA $\mathcal{A}'$ such that $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}', \mathbb{M}(\Sigma', D')) = L(\varphi') \cap \mathbb{M}(\Sigma', D')$. It follows by Proposition 5.5 that there exists a deterministic $\vec{\Sigma}$-ACA $\mathcal{A}$ such that $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma})) = g^{-1}(\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}', \mathbb{P}(\vec{\Sigma'})))$. Finally, applying Proposition 5.3 we obtain

$$\begin{aligned}
\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{CROW}(\vec{\Sigma})) &= \mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma})) \cap \mathbb{CROW}(\vec{\Sigma}) \\
&= g^{-1}(\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}', \mathbb{P}(\vec{\Sigma'}))) \cap g^{-1}(\mathbb{M}(\Sigma', D')) \\
&= g^{-1}(\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}', \mathbb{M}(\Sigma', D'))) \\
&= g^{-1}(L(\varphi') \cap \mathbb{M}(\Sigma', D')) \\
&= L(\varphi) \cap \mathbb{CROW}(\vec{\Sigma}).
\end{aligned}$$

$\square$

As a corollary of Theorems 4.4 and 5.2 we obtain that (existential) MSO sentences, nondeterministic $\vec{\Sigma}$-ACAs in the reading mode R^- and deterministic $\vec{\Sigma}$-ACAs in the reading mode R^- have the same expressive power for $\mathbb{CROW}(\vec{\Sigma})$-pomsets.

Theorem 5.6 *Let $L \subseteq \mathbb{CROW}(\vec{\Sigma})$ and $\beta \in \{+, -\}$. The following are equivalent:*

1. *L is definable by a monadic second order sentence,*
2. *L is definable by an existential monadic second order sentence,*
3. *there exists a nondeterministic $\vec{\Sigma}$-ACA $\mathcal{A}$ such that $L = \mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{CROW}(\vec{\Sigma}))$,*
4. *there exists a deterministic $\vec{\Sigma}$-ACA $\mathcal{A}$ such that $L = \mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{CROW}(\vec{\Sigma}))$.*

In the remainder of this section, we prove an analogous result for the mode $\mathrm{R}^+\mathrm{F}^\beta$. We do this by constructing a deterministic ACA $\mathcal{A}'$ from a deterministic ACA $\mathcal{A}$ such that $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{CROW}(\vec{\Sigma})) = \mathrm{R}^+\mathrm{F}^\beta(\mathcal{A}', \mathbb{CROW}(\vec{\Sigma}))$. Then Theorem 4.4 together with Theorem 5.6 gives the desired result. The main task of this construction is to provide an ACA with the ability to distinguish immediate predecessors among all predecessors in an R^+-run. We show that this is possible in the next proposition using the following lemma.

Lemma 5.7 ([CMZ93]) *For any trace monoid* $\mathbb{M}$*, there exists a finite set* S *and an asynchronous mapping* $\tau : \mathbb{M} \to S$ *such that* $\tau(t)$ *uniquely determines the labels of the maximal elements of* t*, i.e. the set* $\lambda(\max(t))$*, for each trace* $t \in \mathbb{M}$*.*

Note that the mere mapping $t \mapsto \lambda(\max(t))$ is not asynchronous but it is easy to obtain an asynchronous mapping τ satisfying the condition of the lemma above by using the asynchronous time stamping introduced in [CMZ93].

Notation. Let $t = (V, \leq, \lambda)$ be a $\vec{\Sigma}$-pomset. Then $t' := (V, \leq, \lambda \times \mathrm{R}^-)$ is a trace in $\mathbb{M}(\Sigma', D')$ with $g(t) = t'$. We write (t, R^-) as an abbreviation for $g(t) = (V, \leq, \lambda \times \mathrm{R}^-)$. In addition, let $(\downarrow x, \mathrm{R}^-)$ denote $(\downarrow x, \leq \cap (\downarrow x \times \downarrow x), (\lambda \times \mathrm{R}^-) \restriction_{\downarrow x})$ and similarly for $(\Downarrow x, \mathrm{R}^-)$ whenever $x \in V$.

Proposition 5.8 *There exists a deterministic* $\vec{\Sigma}$*-ACA* $\mathcal{A}'' = ((Q'')_{i \in [n]}, (\delta''_{a,J})_{a \in \Sigma, J \subseteq [n]}, F'')$ *(note that all processes have the same set* Q'' *of local states) and a mapping* $\eta : Q'' \to 2^{[n]}$ *such that*

(i) $\mathrm{R}^+\mathrm{F}^\beta(\mathcal{A}'', \mathbb{CROW}(\vec{\Sigma})) = \mathbb{CROW}(\vec{\Sigma})$.

(ii) Let $t = (V, \leq, \lambda)$ *be a CROW-pomset and let* r *be the* R^-*-run of* $\mathcal{A}''$ *on* t*. Then* $\eta \circ r = \mathrm{R}^-$*, that is,* $\mathrm{R}^-(x) = \eta(r(x))$ *for all* $x \in V$*.*

Proof. Let $\tau : \mathbb{M}(\Sigma', D') \to S$ be the asynchronous mapping given by Lemma 5.7. The common local state space of the ACA $\mathcal{A}''$ is given by $Q'' = 2^{[n]} \times S$. For $a \in \Sigma_i$, let $\delta''_{a,\emptyset} = \{(\emptyset, \tau(a, \emptyset))\}$. Now let $\emptyset \neq J \subseteq [n]$, $(M_j, s_j) \in Q''$ for $j \in J$ and $a \in \Sigma_i$. Then $\delta''_{a,J}((M_j, s_j)_{j \in J})$ consists of all pairs $(M, s) \in Q''$ such that there exists a trace $t \in \mathbb{M}(\Sigma', D')$ with

(1) $J = p(t)$, $\tau\partial_j(t) = s_j$ for $j \in J$,

(2) $M = p(\max(t))$, and $s = \tau(t \cdot (a, M))$.

Finally, all tuples of local states are accepting.

First we show that the transition functions are indeed deterministic: So let $(M_j, s_j) \in Q''$ for $j \in J \subseteq [n]$, and $a \in \Sigma_i$. Let $t, t' \in \mathbb{M}(\Sigma', D')$ be traces such that $J = p(t) = p(t')$ and $s_j = \tau\partial_j(t) = \tau\partial_j(t')$ for any $j \in p(t)$. Clearly, $t = \bigvee_{j \in p(t)} \partial_j(t)$ and similarly for t'. Since τ is an asynchronous mapping, $\tau(t) = \tau(t')$ follows from $\tau\partial_j(t) = \tau\partial_j(t')$ for $j \in p(t) = p(t')$. Then $p \circ \max(t) = p \circ \max(t') =: M$ by the choice of the asynchronous mapping τ. Now let $y \in \max(t)$. Then $p(y) \in M$ implying that $\lambda'(y)$ and (a, M) are dependent. Since this holds for all $y \in \max(t)$, the trace $t \cdot (a, M)$ is prime. Similarly, the trace $t' \cdot (a, M)$ is prime. Since, as we saw above, $\tau(t) = \tau(t')$, the asynchronicity of τ implies $\tau(t \cdot (a, M)) = \tau(t' \cdot (a, M)) =: s$. Thus, we showed that (M, s) is the only element of $\delta''_{a,J}((M_j, s_j)_{j \in J})$, i.e. the automaton $\mathcal{A}''$ is deterministic.

To show the first statement of Proposition 5.8, it is sufficient to prove that any CROW-pomset allows a run of the ACA $\mathcal{A}''$. Therefore, let $t = (V, \leq, \lambda) \in \mathbb{CROW}(\vec{\Sigma})$.

Define $r : V \to 2^{[n]} \times S$ by $r(x) := (\mathrm{R}^-(x), \tau(\downarrow x, \mathrm{R}^-))$. Let $x \in V$, $r(x) = (M, s)$ and $r\partial_j(\Downarrow x) = (M_j, s_j)$ for $j \in \mathrm{R}^+(x)$. We have to show that

$$(M, s) \in \delta''_{\lambda(x),\mathrm{R}^+(x)}((M_j, s_j)_{j \in \mathrm{R}^+(x)}).$$

Then $t := (\Downarrow x, \mathrm{R}^-)$ is a trace from $\mathbb{M}(\Sigma', D')$ with $\mathrm{R}^+(x) = p(\Downarrow x) = p(t)$ and $s_j = \tau\partial_j(\Downarrow x, \mathrm{R}^-) = \tau(\partial_j(\Downarrow x), \mathrm{R}^-)$ for all $j \in \mathrm{R}^+(x)$. Thus, (1) holds. Clearly, $M = \mathrm{R}^-(x) = p \circ \max(\Downarrow x) = p \circ \max(t)$ and $s = \tau(\downarrow x, \mathrm{R}^-) = \tau((\Downarrow x, \mathrm{R}^-) \cdot (\lambda(x), M))$ which proves (2). Thus, r is indeed an R^+-run of $\mathcal{A}''$ on t.

Since $\mathcal{A}''$ is deterministic, r is the only possible run of $\mathcal{A}''$ on t. Defining η to be the first projection from Q'' to $2^{[n]}$, the second statement is obvious. $\square$

Now we can easily construct an ACA that simulates an R^--run of a given ACA in the mode R^+ as follows: Let $\mathcal{A} = ((Q_i)_{i \in [n]}, (\delta_{a,J})_{a \in \Sigma, J \subseteq [n]}, F)$ be a $\vec{\Sigma}$-ACA and let $\mathcal{A}'' = ((Q'')_{i \in [n]}, (\delta''_{a,J})_{a \in \Sigma, J \subseteq [n]}, F'')$ be the $\vec{\Sigma}$-ACA from Proposition 5.8. Then define $Q'_i := Q_i \times Q''$ and $F' := \{(q_j, q''_j)_{j \in J} \mid (q_j)_{j \in J} \in F\}$. For $a \in \Sigma$, $J \subseteq [n]$ and $(q_j, q''_j) \in Q'_j$ for $j \in J$, let $\delta'_{a,J}((q_j, q''_j)_{j \in J})$ consist of all tuples (q, q'') with $q'' \in \delta''_{a,J}((q''_j)_{j \in J})$ and $q \in \delta_{a,\eta(q'')}((q_j)_{j \in \eta(q'')})$. Note that, since $\mathcal{A}''$ is deterministic, $\mathcal{A}'$ is deterministic whenever $\mathcal{A}$ is.

Now let $t = (V, \leq, \lambda)$ be a CROW-pomset and let r' be an F^β-successful R^+-run of $\mathcal{A}'$ on t. Then, by Proposition 5.8 (ii), $\pi_1 \circ r'$ is an R^--run of $\mathcal{A}$ on t. By the definition of the accepting states of $\mathcal{A}'$, it is F^β-successful. Hence $\mathrm{R}^+\mathrm{F}^\beta(\mathcal{A}') \subseteq \mathrm{R}^-\mathrm{F}^\beta(\mathcal{A})$. To show the other inclusion, let r be an F^β-successful R^--run of $\mathcal{A}$ on t. By Proposition 5.8 (i), there is an R^+-run r'' of $\mathcal{A}''$ on t. Let $r' = r \times r''$. We show that this is an R^+-run of $\mathcal{A}'$ on t. Let $x \in V$. By Proposition 5.8 (ii), $\mathrm{R}^-(x) = \eta(r''(x))$. Since r is an R^--run of $\mathcal{A}$, we get that $r(x) \in \delta_{a,\eta(r''(x))}(r\partial_j(\Downarrow x)_{j \in \eta(r''(x))})$. Hence $r \times r''$ is an R^+-run on t that is F^β-successful since the second component does not influence the acceptance. Thus $\mathrm{R}^+\mathrm{F}^\beta(\mathcal{A}') = \mathrm{R}^-\mathrm{F}^\beta(\mathcal{A})$. Hence, using Theorems 4.4 and 5.6 we get

Theorem 5.9 *Let $L \subseteq \mathbb{CROW}(\vec{\Sigma})$ and $\alpha, \beta \in \{+, -\}$. The following are equivalent:*

1. *L is definable by a monadic second order sentence,*
2. *L is definable by an existential monadic second order sentence,*
3. *there exists a nondeterministic $\vec{\Sigma}$-ACA $\mathcal{A}$ such that $L = \mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A}, \mathbb{CROW}(\vec{\Sigma}))$,*
4. *there exists a deterministic $\vec{\Sigma}$-ACA $\mathcal{A}$ such that $L = \mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A}, \mathbb{CROW}(\vec{\Sigma}))$.*

6 k-pomsets

Let $t = (V, \leq, \lambda)$ be a $\vec{\Sigma}$-pomset. Furthermore, let k be a positive integer and $C_\ell \subseteq V$ for $1 \leq \ell \leq k$. We call the tuple $(C_1, C_2, \ldots, C_k)$ a *k-chain covering* of t if

1. C_ℓ is a chain for $\ell = 1, 2, \ldots, k$,

2. $V = \bigcup_{\ell \in [k]} C_\ell$ and

3. for any $x, y \in V$ with $x \longrightarrow\!\!< y$ there exists $\ell \in [k]$ with $x, y \in C_\ell$.

The $\vec{\Sigma}$-pomset t is a *k-pomset* if it has a k-chain covering. Let $\mathcal{P}_k$ denote the set of all k-pomsets over $\vec{\Sigma}$.

Remark 6.1 *Let $t = (V, \leq, \lambda) \in \mathbb{CROW}(\vec{\Sigma})$. For $i, j \in [n]$ and $i \neq j$ define $C_{i,j} := \lambda^{-1}(\Sigma_i) \cup \{y \in \lambda^{-1}(\Sigma_j) \mid \exists x \in \lambda^{-1}(\Sigma_i) : x \longrightarrow\!\!< y\}$. Then the set $\{C_{i,j} \mid i, j \in [n], i \neq j\}$ satisfies properties 2 and 3 given above. Furthermore, $C_{i,j}$ is a chain since t is a CROW-pomset. Thus, $\mathbb{CROW}(\vec{\Sigma}) \subseteq \mathcal{P}_{n(n-1)}$.*

6.1 Separating the deterministic classes for k-pomsets

Example 6.2 Let $n = 3$, $k = 2$, $\Sigma_1 = \{a\}$, $\Sigma_2 = \{b\}$ and $\Sigma_3 = \{c\}$. Furthermore, let L be the set of all k-pomsets $(V, \leq, \lambda)$ over $(\Sigma_1, \Sigma_2, \Sigma_3)$ such that $\lambda^{-1}(a)$ is even, $\lambda^{-1}(b)$ and $\lambda^{-1}(c)$ are nonempty and no a-labeled element dominates some b- or c-labeled one. Then L is in $\mathrm{dR}^\alpha\mathrm{F}^+(\mathcal{P}_k)$, but not in $\mathrm{dR}^\alpha\mathrm{F}^-(\mathcal{P}_k)$ for $\alpha \in \{+, -\}$.

Proof. Let $Q_1 = \{0, 1\}$ and $Q_2 = Q_3 = \{0\}$. Furthermore, we define transition functions as follows:

$$\delta_{a,J}((q_i)_{i \in J}) = \begin{cases} \emptyset & 2 \in J \text{ or } 3 \in J \\ \{(q_1 + 1) \mod 2\} & J = \{1\} \\ \{1\} & J = \emptyset \end{cases}$$

and $\delta_{b,J}((q_i)_{i \in J}) = \delta_{c,J}((q_i)_{i \in J}) = \{0\}$ for any J and q_i. With $F = \{(0, 0, 0)\}$, we get a deterministic $\vec{\Sigma}$-ACA $\mathcal{A} = ((Q_i)_{i \in [3]}, (\delta_{d,J}), F)$ such that $\mathrm{R}^-\mathrm{F}^+(\mathcal{A}, \mathcal{P}_k) = \mathrm{R}^+\mathrm{F}^+(\mathcal{A}, \mathcal{P}_k) = L$ witnessing $L \in \mathrm{dR}^\alpha\mathrm{F}^+(\mathcal{P}_k)$.

Suppose, $\mathcal{A}$ is a deterministic $\vec{\Sigma}$-ACA such that $L = \mathrm{R}^\alpha\mathrm{F}^-(\mathcal{A}, \mathbb{P}(\vec{\Sigma}))$ for $\alpha = +$ or $\alpha = -$. Furthermore, let $\ell = |Q_2 \times Q_3| + 1$. To derive a contradiction, let t_m (for $m \in \mathbb{N}$) denote the $\vec{\Sigma}$-pomset depicted in Figure 1. The labeling is defined canonically by $\lambda(a_i) = a$, $\lambda(b_1) = b$ and $\lambda(c_1) = c$. Since $t_m \in L$, there is an F^--successful R^α-run r_m of $\mathcal{A}$ on t_m. Since ℓ is larger than the number of tuples from $Q_2 \times Q_3$, there are $i < j \leq \ell$ such that $r_i(b_1) = r_j(b_1)$ and $r_i(c_1) = r_j(c_1)$.

Now consider the $\vec{\Sigma}$-pomset $t = (V, \leq, \lambda)$ on Figure 1 that does not belong to L.

The labeling on t is defined canonically. Let $r : V \to Q_1 \cup Q_2 \cup Q_3$ be the restriction of r_j, i.e. $r = r_j \restriction_V$. Since $\mathcal{A}$ is deterministic, we get $r \restriction \{a_1, a_2, \ldots, a_{2i+2}\} = r_i \restriction \{a_1, a_2, \ldots, a_{2i+2}\}$, in particular $r(a_{2i+2}) = r_i(a_{2i+2})$. Thus, r is an R^α-run of $\mathcal{A}$ on t. Since $\mathrm{F}^-(t) = \{2, 3\} = \mathrm{F}^-(t_j)$, the run r is F^--successful contradicting $L = \mathrm{R}^\alpha\mathrm{F}^-(\mathcal{A}, \mathbb{P}(\vec{\Sigma}))$. □

Example 6.3 Let $n = k = 2$, $\Sigma_1 = \{a\}$ and $\Sigma_2 = \{b\}$. Furthermore, let L consist of all k-pomsets $(V, \leq, \lambda)$ over (Σ_1, Σ_2) that have a largest element x such that $\lambda(x) = b$. Then L is in $\mathrm{dR}^\alpha\mathrm{F}^-(\mathcal{P}_k)$, but not in $\mathrm{dR}^\alpha\mathrm{F}^+(\mathcal{P}_k)$ for any $\alpha \in \{+, -\}$.

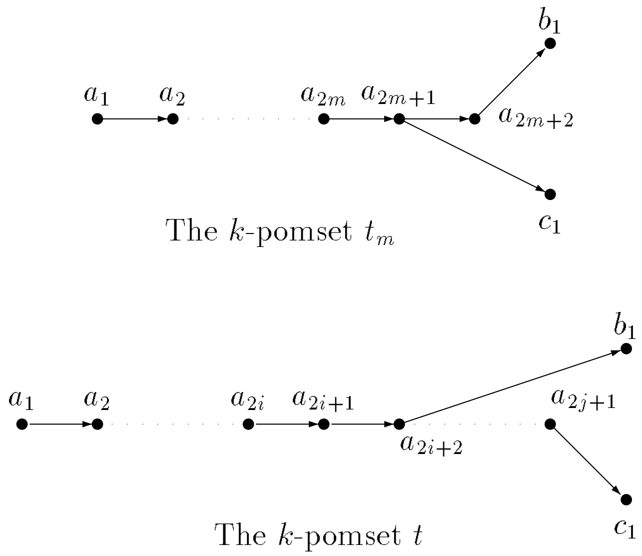


Figure 1: cf. Proof of Example 6.2

Proof. Clearly, there is a deterministic $\vec{\Sigma}$-ACA that allows an R^α-run on any $\vec{\Sigma}$-pomset. Let F consist of all tuples $(q_i)_{i\in J}$ of local states with $J=\{2\}$. Then this $\vec{\Sigma}$-ACA $\mathcal{A}$ accepts L in the mode $\mathrm{R}^\alpha\mathrm{F}^-$, i.e. $\mathrm{R}^\alpha\mathrm{F}^-(\mathcal{A},\mathcal{P}_k)=L$ witnessing $L\in \mathrm{dR}^\alpha\mathrm{F}^-(\mathcal{P}_k)$.

We want to show that there is no deterministic $\vec{\Sigma}$-ACA $\mathcal{A}$ such that $\mathrm{R}^\alpha\mathrm{F}^+(\mathcal{A},\mathcal{P}_k)=L$. By contradiction, assume $\mathcal{A}$ is such a $\vec{\Sigma}$-ACA. Let $\ell=|Q_1|+2$ and consider the k-pomset $t=(V,\le,\lambda)$ with $V=\{a_i\mid i=1,2,\dots,\ell\}\cup\{b_1\}$, $a_1<a_2\cdots<a_\ell<b_1$ and with the canonical labeling λ. Then $t\in L$. Hence there is an F^+-successful R^α-run r of $\mathcal{A}$ on t. Since $\ell>|Q_1|+1$, there are $i<j<\ell$ such that $r(a_i)=r(a_j)$. Now consider the k-pomsets t_1 and t_2 with $V_1=V_2=\{a_\ell\mid \ell=1,2,\dots,j\}\cup\{b_1\}$ and the canonical labeling. The order relations are defined by $a_1<_1 a_2<_1 a_3\cdots<_1 a_j<_1 b_1$ (i.e. t_1 is a linear ordering with maximal element b_1) and $a_1<_2 a_2<_2 a_3\cdots<_2 a_j$ and $a_i<_2 b_1$ (i.e. in t_2, the a-labeled elements are linearly ordered, but the maximal element b_1 covers a_i). Since $t_1\in L$, there is an F^+-successful R^α-run r_1 of $\mathcal{A}$ on t_1. Since $\mathcal{A}$ is deterministic, we have $r_1(a_\ell)=r(a_\ell)$ for $\ell\le j$. This implies $r_1(a_i)=r_1(a_j)$ since the equality holds for the run r. Hence r_1 is an R^α-run on t_2, too. This implies that r_1 is an F^+-successful R^α-run on t_2, contradicting $L=\mathrm{R}^\alpha\mathrm{F}^+(\mathcal{A},\mathcal{P}_k)$. $\square$

The two examples above can be generalized to prove the following

Proposition 6.4 *Let $n,k\ge 3$. Then the classes $\mathrm{dR}^+\mathrm{F}^+(\mathcal{P}_k)$, $\mathrm{dR}^-\mathrm{F}^+(\mathcal{P}_k)$, $\mathrm{dR}^+\mathrm{F}^-(\mathcal{P}_k)$, and $\mathrm{dR}^-\mathrm{F}^-(\mathcal{P}_k)$ are pairwise incomparable. Hence, also the classes $\mathrm{dR}^+\mathrm{F}^+(\mathbb{P}(\vec{\Sigma}))$, $\mathrm{dR}^-\mathrm{F}^+(\mathbb{P}(\vec{\Sigma}))$, $\mathrm{dR}^+\mathrm{F}^-(\mathbb{P}(\vec{\Sigma}))$, and $\mathrm{dR}^-\mathrm{F}^-(\mathbb{P}(\vec{\Sigma}))$ are pairwise incomparable.*

Proof. The incomparability of the classes from $\{\mathrm{dR}^+\mathrm{F}^+(\mathcal{P}_k), \mathrm{dR}^-\mathrm{F}^+(\mathcal{P}_k)\}$ and those from $\{\mathrm{dR}^+\mathrm{F}^-(\mathcal{P}_k), \mathrm{dR}^-\mathrm{F}^-(\mathcal{P}_k)\}$ is witnessed by the two examples above. A language that is deterministically acceptable in the mode $\mathrm{R}^+\mathrm{F}^\beta$, but not in the mode $\mathrm{R}^-\mathrm{F}^\beta$ is easily obtained from the language in Example 6.2 as follows: L consists of all k-pomsets such that *any b-labeled element dominates an even number of a-labeled elements.* This language is in $\mathrm{dR}^+\mathrm{F}^\beta(\mathcal{P}_k)$ for $\beta \in \{+,-\}$. To show that it is not in $\mathrm{dR}^-\mathrm{F}^\beta(\mathcal{P}_k)$, one adds an additional maximal b-labeled element b_2 to the k-pomsets t_m and t (Figure 2). Then, the reading domain of this additional element in mode R^- is precisely the acceptance domain of the original pomset on accepting mode F^-. Therefore, the proof goes through as before.

Similarly, one can adopt the idea from Example 6.3 to obtain a language that is in $\mathrm{dR}^-\mathrm{F}^\beta(\mathcal{P}_k)$ but not in $\mathrm{dR}^+\mathrm{F}^\beta(\mathcal{P}_k)$.

To show that the classes $\mathrm{dR}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ are mutually incomparable, let $\alpha, \beta, \alpha', \beta' \in \{+,-\}$ with $\mathrm{dR}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma})) \subseteq \mathrm{dR}^{\alpha'}\mathrm{F}^{\beta'}(\mathbb{P}(\vec{\Sigma}))$. Now let $\mathcal{A}$ be a deterministic $\vec{\Sigma}$-ACA. By our assumption $\mathrm{dR}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma})) \subseteq \mathrm{dR}^{\alpha'}\mathrm{F}^{\beta'}(\mathbb{P}(\vec{\Sigma}))$, there is a deterministic $\vec{\Sigma}$-ACA $\mathcal{A}'$ such that $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma})) = \mathrm{R}^{\alpha'}\mathrm{F}^{\beta'}(\mathcal{A}', \mathbb{P}(\vec{\Sigma}))$. Hence in particular $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A}, \mathcal{P}_k) = \mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma})) \cap \mathcal{P}_k = \mathrm{R}^{\alpha'}\mathrm{F}^{\beta'}(\mathcal{A}', \mathbb{P}(\vec{\Sigma})) \cap \mathcal{P}_k = \mathrm{R}^{\alpha'}\mathrm{F}^{\beta'}(\mathcal{A}', \mathcal{P}_k)$ and therefore $\alpha = \alpha'$ and $\beta = \beta'$ by what we showed above. $\square$

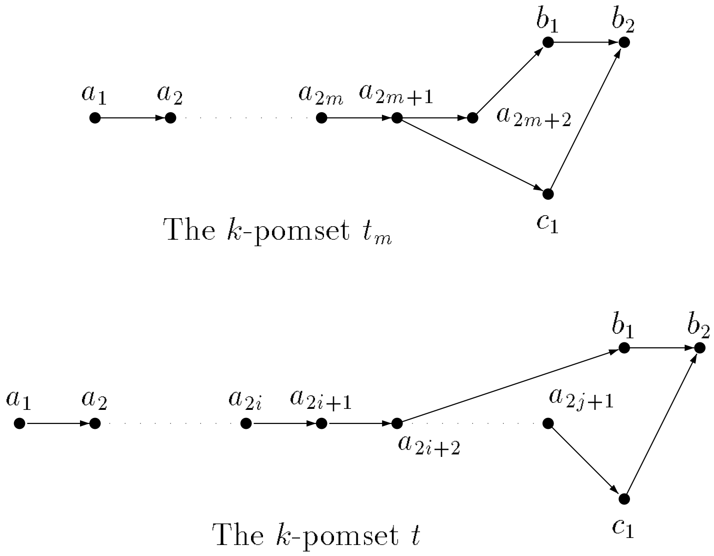


Figure 2: cf. Proof of Proposition 6.4

Note that the set of k-pomsets forms a prefix closed class of $\vec{\Sigma}$-pomsets. Hence Proposition 4.3 can be applied. Therefore, any language of k-pomsets recognizable by an asynchronous mapping is in the intersection of all classes $\mathrm{dR}^\alpha\mathrm{F}^\beta(\mathcal{P}_k)$. Since these classes are incomparable, the languages recognizable by an asynchronous mapping form a proper subclass of each of them. Hence we showed that the converse of Proposition 4.3 does not hold.

6.2 The mode R^-

Due to Proposition 6.4, the expressive power of deterministic asynchronous cellular automata does not capture that of monadic second order logic on k-pomsets. It is the aim of the remaining section to show that, on the other side, nondeterministic asynchronous cellular automata do the job. More precisely, we saw that the reading and the accepting mode of deterministic automata influence the expressive power relative to the class of k-pomsets. Here, we will see that this is not the case for nondeterministic automata. This subsection deals with the proof that nondeterministic automata in the mode R^- have the same expressive power as monadic second order logic. In the following subsection we will simulate an R^--automaton by an R^+-ACA.

Definition 6.5 *Let $t = (V, \leq, \lambda)$ be a $\vec{\Sigma}$-pomset, $k \in \mathbb{N}$ and $\Lambda : V \to (2^{[k]} \setminus \{\emptyset\})$. The function Λ is a* k-chain mapping *if*

1. *for all minimal vertices $x, y \in V$, if $x \neq y$ then $\Lambda(x) \cap \Lambda(y) = \emptyset$,*
2. *for all non minimal vertices $x \in V$ and $\ell \in \Lambda(x)$, there exists $y \in V$ with $y \prec\!\!\cdot\, x$ and $\ell \in \Lambda(y)$.*
3. *for all non maximal vertices $x \in V$ and $\ell \in \Lambda(x)$, there exists at most one $y \in V$ with $x \prec\!\!\cdot\, y$ and $\ell \in \Lambda(y)$.*
4. *For all $x, y \in V$, if $x \prec\!\!\cdot\, y$ then $\Lambda(x) \cap \Lambda(y) \neq \emptyset$.*

The following lemma relates k-chain mappings and k-chain coverings thereby justifying the name k-chain mapping.

Lemma 6.6 *Let $t = (V, \leq, \lambda)$ be a $\vec{\Sigma}$-pomset. Then $t \in \mathcal{P}_k$ iff there exists a k-chain mapping. In particular, if Λ is a k-chain mapping of t and $\ell \in [k]$, then the set $\Lambda^{-1}(\ell) = \{x \in V \mid \ell \in \Lambda(x)\}$ is a chain.*

Proof. Let $t \in \mathcal{P}_k$. Then there exists a k-chain covering $(C_\ell)_{\ell \in [k]}$ of t. We may assume that each of the chains C_ℓ is a maximal chain. Now define $\Lambda(x) := \{\ell \in [k] \mid x \in C_\ell\}$. Then $\Lambda : V \to (2^{[k]} \setminus \{\emptyset\})$ since $V = \bigcup_{\ell \in [k]} C_\ell$. Since C_ℓ is a chain for each $\ell \in [k]$, any two different minimal elements of t belong to disjoint sets of chains. Hence the first property is satisfied. Now let $x \in V$ be non minimal and $\ell \in \Lambda(x)$. Since the chain C_ℓ is maximal, there exists $y \in V$ with $y \prec\!\!\cdot\, x$ and $\ell \in \Lambda(y)$. Thus, the second requirement is satisfied. Since the upper neighbors of x are mutually incomparable, the third clause holds as well. If $x \prec\!\!\cdot\, y$, then there exists $\ell \in [k]$ such that $x, y \in C_\ell$. Hence $\ell \in \Lambda(x) \cap \Lambda(y)$ proving the last statement.

Conversely, let t be a $\vec{\Sigma}$-pomset and let Λ be a k-chain mapping. For $\ell \in [k]$, define $C_\ell := \{x \in V \mid \ell \in \Lambda(x)\}$. Since $\Lambda(x) \neq \emptyset$ for all $x \in V$, we get $V = \bigcup_{\ell \in [k]} C_\ell$. By the last property for Λ, for any $x \longrightarrow\!\!\!< y$ there exists $\ell \in [k]$ with $x, y \in C_\ell$. It remains to show that C_ℓ is a chain for any ℓ: Let $x, y \in C_\ell$. By the second property of Λ, there exist chains $x_0 \longrightarrow\!\!\!< x_1 \cdots \longrightarrow\!\!\!< x_a = x$ and $y_0 \longrightarrow\!\!\!< y_1 \cdots \longrightarrow\!\!\!< y_b = y$ with x_0, y_0 minimal in t, and $x_i, y_j \in C_\ell$ for $0 \leq i \leq a$, $0 \leq j \leq b$, $a \leq b$, say. By the first property of Λ, $x_0 = y_0$. Let $0 \leq i < a$ such that $x_i = y_i$. This element is covered by x_{i+1} and by y_{i+1}. By the third property of Λ, $x_{i+1}, y_{i+1} \in C_\ell$ implies $x_{i+1} = y_{i+1}$. This shows that $x \leq y$. ☐

Next we show that $\mathcal{P}_k \in \mathrm{R^-F^-}(\mathbb{P}(\vec{\Sigma}))$ by the construction of a recognizing automaton $\mathcal{A}_k^-$. This ACA will be used later to relabel k-pomsets into traces.

Let $\mathrm{part}(k, n)$ denote the set of *partial* functions g from $[k]$ to $[n]$ with $\mathrm{dom}(g) \neq \emptyset$. For a partial function $f \in \mathrm{part}(k, n)$, we first define an ACA $\mathcal{A}_k^-(f)$ whose local states are partial functions in $\mathrm{part}(k, n)$. Intuitively, a node x of some k-pomset t will be labeled by the partial function g in some run of $\mathcal{A}_k^-(f)$ if $\mathrm{dom}(g)$ is the set of (maximal) chains going through x and for all $\ell \in \mathrm{dom}(g)$, $g(\ell)$ is the next process for the chain ℓ. The partial mapping f is in some sense the initial state of the automaton $\mathcal{A}_k^-(f)$: $f(\ell) = i$ iff the chain ℓ starts in process i. As we will see, runs of this automaton correspond to k-chain mappings.

More precisely, the ACA $\mathcal{A}_k^-(f)$ is defined as follows: The set of local states (common for all processes) is $Q = \mathrm{part}(k, n)$. For $a \in \Sigma_i$, let $\delta_{a,\emptyset}$ consist of all partial functions $g \in Q$ with $\mathrm{dom}(g) = f^{-1}(i)$. For $\emptyset \neq J \subseteq [n]$ and $g_j \in Q$ for $j \in J$, we let $\delta_{a,J}((g_j)_{j \in J})$ be the set of all partial functions $g \in Q$ such that

1. $\forall j \in J, \exists \ell \in \mathrm{dom}(g)$ such that $g_j(\ell) = i$ and

2. $\forall \ell \in \mathrm{dom}(g), \exists j \in J$ such that $g_j(\ell) = i$.

Finally, all tuples of states are accepting. Let $\mathcal{A}_k^-$ denote the disjoint union of the automata $\mathcal{A}_k^-(f)$ for all partial functions $f \in \mathrm{part}(k, n)$. Note that not all runs of $\mathcal{A}_k^-$ are successful, only those that lie completely inside $\mathcal{A}_k^-(f)$ for some $f \in \mathrm{part}(k, n)$ are. This can be easily checked with either acceptance condition F^β with $\beta \in \{+, -\}$.

The following lemma shows that the k-chain mappings Λ on a $\vec{\Sigma}$-pomset t coincide precisely with the mappings $\mathrm{dom} \circ r : V \to \mathcal{P}([k])$ where r is an F^β-successful R^--run of the automaton $\mathcal{A}_k^-$ constructed above.

Lemma 6.7 *For $k \in \mathbb{N}$, $\beta \in \{+, -\}$ and $t = (V, \leq, \lambda) \in \mathbb{P}(\vec{\Sigma})$, we have:*

1. *for any F^β-successful R^--run r of $\mathcal{A}_k^-$ on t, the mapping $\mathrm{dom} \circ r : V \to 2^{[k]} \setminus \{\emptyset\}$ is a k-chain mapping.*

2. *For any k-chain mapping Λ on t, there exists an F^β-successful R^--run r of $\mathcal{A}_k^-$ on t such that $\Lambda = \mathrm{dom} \circ r$.*

Proof. 1. Let $r : V \to \mathrm{part}(k, n)$ be an F^β-successful R^--run of $\mathcal{A}_k^-$ on t and let $\Lambda = \mathrm{dom} \circ r$. There exists a partial function $f \in \mathrm{part}(k, n)$ such that r is an R^--run of $\mathcal{A}_k^-(f)$. Now let $x, y \in V$ be minimal and different. Then $r(x) \in \delta_{\lambda(x),\emptyset}$, and therefore $\mathrm{dom} \circ r(x) = f^{-1}(p(x))$. Similarly, $\mathrm{dom} \circ r(y) = f^{-1}(p(y))$. Since x and y are incomparable, $p(x) \neq p(y)$. Hence $\Lambda(x)$ and $\Lambda(y)$ are disjoint. Thus we showed the first condition of Definition 6.5.

Now, let $x \in V$ be non minimal. For all $j \in \mathrm{R}^-(x)$, let $x_j \in V$ be such that $x_j \multimap x$ and $p(x_j) = j$. Let also $g_j = r(x_j)$ and $g = r(x)$. From $g \in \delta_{\lambda(x),\mathrm{R}^-(x)}((g_j)_{j \in \mathrm{R}^-(x)})$ we deduce

1. $\forall j \in \mathrm{R}^-(x)$, $\exists \ell \in \mathrm{dom}(g) \cap \mathrm{dom}(g_j) = \Lambda(x) \cap \Lambda(x_j)$ showing Definition 6.5 (4),
2. $\forall \ell \in \Lambda(x) = \mathrm{dom}(g)$, $\exists j \in \mathrm{R}^-(x)$ with $\ell \in \mathrm{dom}(g_j) = \Lambda(x_j)$ showing Definition 6.5 (2).

Finally, let $x \in V$ be non maximal. The third condition in Definition 6.5 is a direct consequence of the following claim: For all $x, y \in V$, if $x \multimap y$ and $\ell \in \Lambda(x) \cap \Lambda(y)$ then $p(y) = r(x)(\ell)$. To prove this claim, let $x, y \in V$ be such that $x \multimap y$ and let $\ell \in \Lambda(x) \cap \Lambda(y)$. By Definition 6.5 (2) and (4) shown above, we find

- $y_0 \multimap y_1 \cdots \multimap y_i = y$ with y_0 minimal and $r(y_q)(\ell) = p(y_{q+1})$ for all $0 \leq q < i$,
- $x_0 \multimap x_1 \cdots \multimap x_j = x$ with x_0 minimal and $r(x_q)(\ell) = p(x_{q+1})$ for all $0 \leq q < j$.

Now, we show by induction that $x_q = y_q$ for all $0 \leq q \leq \min(i, j)$. Indeed, $\ell \in \mathrm{dom}(r(x_0)) \cap \mathrm{dom}(r(y_0)) = \Lambda(x_0) \cap \Lambda(y_0)$. Hence $x_0 = y_0$ by the first point shown above. Now assume that $x_q = y_q$ for some $0 \leq q < \min(i, j)$. We have $p(x_{q+1}) = r(x_q)(\ell) = r(y_q)(\ell) = p(y_{q+1})$. Since $x_q \multimap x_{q+1}$ and $y_q \multimap y_{q+1}$ it follows that $x_{q+1} = y_{q+1}$. Finally, using $x_j = x \multimap y = y_i$, we deduce that $j = i - 1$ and we obtain $r(x)(\ell) = p(y)$.

2. Assume now that Λ is a k-chain mapping. We will construct an F^β-successful R^--run r of $\mathcal{A}_k^-$ such that $\mathrm{dom} \circ r = \Lambda$. Let $x \in V$. Indeed, the domain of the partial function $r(x) \in \mathrm{part}(k, n)$ will be $\Lambda(x)$. Now, for all $\ell \in \mathrm{dom}(r(x)) = \Lambda(x)$, there exists at most one $y \in V$ such that $x \multimap y$ and $\ell \in \Lambda(y)$ (Definition 6.5 (3)). If such a y exists then we set $r(x)(\ell) = p(y)$ and otherwise we set $r(x)(\ell) = 1$ (in this last case, we could give any value since it will never be used).

Let $f \in \mathrm{part}(k, n)$ be the partial function defined by $\ell \in \mathrm{dom}(f)$ iff there exists a minimal vertex $x \in V$ with $\ell \in \Lambda(x)$ and in this case we set $f(\ell) = p(x)$. Note that f is well-defined thanks to Definition 6.5 (1).

We will show that indeed r is a run of $\mathcal{A}_k^-(f)$. Clearly, if $x \in V$ is minimal then we have $\mathrm{dom}(r(x)) = \Lambda(x) = f^{-1}(p(x))$ as required by the initial transitions of $\mathcal{A}_k^-(f)$.

Now, let $x \in V$ be non minimal. For all $j \in \mathrm{R}^-(x)$, let $x_j \multimap x$ be such that $p(x_j) = j$. We will show that $r(x) \in \delta_{\lambda(x),\mathrm{R}^-(x)}(r(x_j)_{j \in \mathrm{R}^-(x)})$. First, for all $j \in \mathrm{R}^-(x)$, by Definition 6.5 (4), there exists $\ell \in \Lambda(x) \cap \Lambda(x_j) = \mathrm{dom}(r(x)) \cap \mathrm{dom}(r(x_j))$. By definition of $r(x_j)$, it follows that $r(x_j)(\ell) = p(x)$. Second, for $\ell \in \mathrm{dom}(r(x)) = \Lambda(x)$, there exists $j \in \mathrm{R}^-(x)$ with $\ell \in \Lambda(x_j) = \mathrm{dom}(r(x_j))$ (Definition 6.5 (2)). By definition of $r(x_j)$, it follows that $r(x_j)(\ell) = p(x)$. Thus we have shown that r is an R^--run of $\mathcal{A}_k^-(f)$ which concludes the proof. $\square$

Corollary 6.8 *For $k \in \mathbb{N}$ and $\beta \in \{+, -\}$, we have* $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}_k^-) = \mathcal{P}_k$.

Proof. This is immediate by the lemma above and by Lemma 6.6. $\square$

We now define a trace alphabet (Γ, D) as follows: For $i \in [n]$ let $\Gamma_i := \Sigma_i \times (2^{[k]} \setminus \{\emptyset\})$ and $\Gamma = \bigcup_{i\in[k]} \Gamma_i$. The dependence relation D is defined by $D = \{((a, M), (b, N)) \mid M \cap N \neq \emptyset\}$. This binary relation on Γ is obviously reflexive (since we excluded $\emptyset$ from the possible second components) and symmetric. Thus (Γ, D) is indeed a dependence alphabet. Let $\mathbb{M}(\Gamma, D)$ denote the trace monoid over (Γ, D). For a trace $(V, \leq, \lambda_\Gamma)$ from $\mathbb{M}(\Gamma, D)$ let $\Pi(V, \leq, \lambda) = (V, \leq, \pi_1 \circ \lambda_\Gamma)$, i.e. the mapping Π just *forgets* the second component of the labeling λ_Γ. Furthermore, let the set $\mathbb{M}'$ consist of all traces $(V, \leq, \lambda_\Gamma)$ from $\mathbb{M}(\Gamma, D)$ such that for all $x, y \in V$, $x \parallel y$ implies $p(x) \neq p(y)$. This set is easily definable by a sentence of the monadic second order logic over the alphabet Γ.

Lemma 6.9 *With the definitions above, we have* $\mathbb{M}' = \Pi^{-1}(\mathcal{P}_k)$. *If* Λ *is a* k*-chain mapping of the* k*-pomset* $t = (V, \leq, \lambda)$*, then* $(V, \leq, \lambda \times \Lambda) \in \mathbb{M}'$.

Proof. Let $t = (V, \leq, \lambda_\Gamma) \in \mathbb{M}'$ and $\lambda := \pi_1 \circ \lambda_\Gamma$. For $x, y \in V$ with $p(x) = p(y)$, x and y are comparable with respect to $\leq$ by the definition of $\mathbb{M}'$. Thus, $\Pi(t)$ is in $\mathbb{P}(\vec{\Sigma})$. Let $C_\ell = \{x \in V \mid \ell \in \pi_2 \circ \lambda_\Gamma(x)\}$ for $\ell \in [k]$. For $x, y \in C_\ell$ thus $\pi_2 \circ \lambda_\Gamma(x) \cap \pi_2 \circ \lambda_\Gamma(y) \neq \emptyset$, i.e. $(\lambda_\Gamma(x), \lambda_\Gamma(y)) \in D$. Hence x and y are comparable proving that C_ℓ is a chain in t and therefore in $\Pi(t)$. Now suppose $x, y \in V$ with $x \multimap y$ Then, since t is a trace, $(\lambda_\Gamma(x), \lambda_\Gamma(y)) \in D$. Hence there exists $\ell \in [k]$ such that $\ell \in \pi_2 \circ \lambda_\Gamma(x) \cap \pi_2 \circ \lambda_\Gamma(y)$, i.e. $x, y \in C_\ell$. Thus, we showed that $(C_\ell)_{\ell\in[k]}$ is a covering as required, i.e. that $\Pi(t) \in \mathcal{P}_k$.

Conversely, let $t = (V, \leq, \lambda) \in \mathcal{P}_k$ and let Λ be a k-chain mapping over t. We will show that $t' = (V, \leq, \lambda \times \Lambda) \in \mathbb{M}'$. Let $x, y \in V$. If $x \multimap y$, there is $\ell \in \Lambda(x) \cap \Lambda(y)$ and therefore $(\lambda(x), \Lambda(x))$ and $(\lambda(y), \Lambda(y))$ are dependent. If $x \parallel y$, we have $\Lambda(x) \cap \Lambda(y) = \emptyset$ since the sets $\{z \in V \mid \ell \in \Lambda(z)\}$ are chains by Lemma 6.6. Hence in this case $(\lambda(x), \Lambda(x))$ and $(\lambda(y), \Lambda(y))$ are independent. Thus we showed $(V, \leq, \lambda \times \Lambda) \in \mathbb{M}(\Gamma, D)$. If $x, y \in V$ with $p(x) = p(y)$, we get that x and y are comparable since t is a $\vec{\Sigma}$-pomset. Hence $(V, \leq, \lambda \times \Lambda) \in \mathbb{M}'$. $\square$

Lemma 6.10 *Let* ϕ *be a sentence of the monadic second order logic over the alphabet* Σ *and* $\beta \in \{+, -\}$. *Then there exists a deterministic* $\vec{\Gamma}$*-ACA* $\mathcal{A}_\phi$ *such that*

$$\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}_\phi, \mathbb{M}(\Gamma, D)) = \Pi^{-1}(L(\phi) \cap \mathcal{P}_k).$$

Proof. The sentence ϕ contains atomic formulas of the form $\lambda(x) = a$ for $a \in \Sigma$. Replace any occurrence of such a formula by $\bigvee_{\emptyset\neq M\subseteq[k]} \lambda_\Gamma(x) = (a, M)$. The result is denoted by ψ. Note that ψ is a sentence of the monadic second order logic over the alphabet Γ. Now let $s \in \mathbb{M}'$. Then it is easily seen that $s \models \psi$ iff $\Pi(s) \models \phi$. Furthermore, there is a monadic second order sentence ξ over the alphabet Γ such that $L(\xi) = \mathbb{M}'$. Thus, we have $L(\psi \wedge \xi) = \Pi^{-1}(L(\phi) \cap \mathcal{P}_k)$. By [Tho90, CMZ93], there exists an asynchronous mapping from $\mathbb{M}(\Gamma, D)$ to some finite set recognizing $L(\psi \wedge \xi)$. Using Proposition 4.3, we obtain the required deterministic $\vec{\Gamma}$-ACA $\mathcal{A}_\phi$. $\square$

Corollary 6.11 *Let ϕ be a sentence of the monadic second order logic over the alphabet Σ and $\beta \in \{+,-\}$. Then there exists a $\vec{\Sigma}$-ACA $\mathcal{A}$ such that* $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A},\mathcal{P}_k) = L(\phi)\cap\mathcal{P}_k$.

Proof. By Lemma 6.10 there exists a $\vec{\Gamma}$-ACA $\mathcal{A}_\phi = ((Q_i^\phi)_{i\in[n]},(\delta^\phi_{(a,M),J}),F^\phi)$ such that $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}_\phi,\mathbb{M}(\Gamma,D)) = \Pi^{-1}(L(\phi)\cap\mathcal{P}_k)$. Furthermore, let $\mathcal{A}_k^- = ((Q_i)_{i\in[n]},(\delta_{a,J}),F)$ be the ACA constructed above. Now we describe a $\vec{\Sigma}$-ACA $\mathcal{A}' = ((Q_i')_{i\in[n]},(\delta'_{a,J}),F')$ over the alphabet $\vec{\Sigma}$ as follows: $Q_i' = Q_i\times Q_i^\phi$ and a tuple $(g_i,q_i)_{i\in J}$ belongs to F' iff $(g_i)_{i\in J}\in F$ and $(q_i)_{i\in J}\in F^\phi$. To define the transition functions, let $\delta'_{a,J}((g_i,q_i)_{i\in J})$ be the set of all pairs (g,q) satisfying $g\in\delta_{a,J}((g_i)_{i\in J})$ and $q\in\delta^\phi_{(a,M),J}((q_i)_{i\in J})$ with $M=\mathrm{dom}(g)$. Note that a run of the $\vec{\Sigma}$-ACA $\mathcal{A}'$ "contains" a run of $\mathcal{A}_k^-$. This run "relabels" the k-pomset t in consideration into some trace $s\in\Pi^{-1}(t)$ (see Lemmas 6.7 and 6.9). The trace s is in fact the actual input of the automaton $\mathcal{A}_\phi$. Therefore, the k-pomset t is accepted by $\mathcal{A}'$ iff $s\in\Pi^{-1}(t)$ is accepted by $\mathcal{A}_\phi$, that is, iff $t\in L(\phi)$. □

6.3 The mode R^+

In this section, we construct a $\vec{\Sigma}$-ACA $\mathcal{A}_k^+$ that will recognize the set $\mathcal{P}_k$ of k-pomsets in the mode $\mathrm{R}^+\mathrm{F}^-$. Then we use this automaton to simulate in the mode R^+ the behavior of an ACA that works in the mode R^-. To achieve this, we proceed similarly to the construction of $\mathcal{A}_k^-$ by first fixing a partial function $f\in\mathrm{part}(k,n)$. The additional difficulty comes from the fact that in the R^+-mode one cannot distinguish immediate predecessors among all predecessors (cf. Examples 7.2 and 7.1). For this reason, we will use some fixed asynchronous mapping $\tau:\mathbb{M}(\Gamma,D)\to S$ given by Lemma 5.7 where (Γ,D) is defined as in Section 6.2. Note that τ allows to determine the labels of the maximal elements of a trace in $\mathbb{M}(\Gamma,D)$.

Notation. Let $t=(V,\leq,\lambda)$ be a $\vec{\Sigma}$-pomset and $\Lambda: V\to 2^{[k]}\setminus\{\emptyset\}$. Then we write (t,Λ) as an abbreviation for $(V,\leq,\lambda\times\Lambda)$. In addition, let $({\downarrow}x,\Lambda)$ denote the restriction of $(V,\leq,\lambda\times\Lambda)$ to ${\downarrow}x$, i.e. $({\downarrow}x,\leq\cap({\downarrow}x\times{\downarrow}x),(\lambda\times\Lambda)\restriction_{{\downarrow}x})$ and similarly for $({\Downarrow}x,\Lambda)$ whenever $x\in V$.

Given a partial mapping $f\in\mathrm{part}(k,n)$, the local state space (common for all processes) of the automaton $\mathcal{A}_k^+(f)$ will be $Q=\mathrm{part}(k,n)\times S$ where S is the image set of the asynchronous mapping τ. Let $t=(V,\leq,\lambda)$ be a $\vec{\Sigma}$-pomset and let $r:V\to Q$ be a mapping. We will define the automaton in such a way that r is an R^+-run of $\mathcal{A}_k^+(f)$ iff

1. $\pi_1\circ r$ is an R^--run of $\mathcal{A}_k^-(f)$, and
2. $\pi_2\circ r(x) = \tau({\downarrow}x,\mathrm{dom}\circ\pi_1\circ r)$ for all $x\in V$.

The construction of a run r will proceed as follows. Let $x\in V$ and assume that r is already defined on ${\Downarrow}x$ and satisfies the two conditions above. Let $\Lambda=\mathrm{dom}\circ\pi_1\circ r$ be the associated k-chain mapping of ${\Downarrow}x$ (cf. Lemma 6.7(1)). Since τ is an asynchronous mapping, we can first compute $\tau({\Downarrow}x,\Lambda)$ using the second components $(\pi_2\circ r(\partial_j{\Downarrow}x))_{j\in\mathrm{R}^+(x)}$

and we deduce $\mathrm{R}^-(x)$ using the property of τ. Then we compute the first component of the next state $r(x) = (g, s)$ according to $\mathcal{A}_k^-(f)$: $g \in \delta_{\lambda(x),\mathrm{R}^-(x)}((\pi_1 \circ r(\partial_j \Downarrow x))_{j \in \mathrm{R}^-(x)})$. Now we know the full (trace) label of the vertex x : $\lambda_\Gamma(x) = (\lambda(x), \mathrm{dom}(g))$ and we can compute the second component of the next state: $s = \tau(\downarrow x, \Lambda)$.

The formal construction of the automaton and the proof that it works as intended will be somewhat technical but the reader should already be convinced that indeed it is possible to construct such an automaton.

We first define an automaton $\mathcal{A}_k^+(f)$ for any partial mapping $f \in \mathrm{part}(k, n)$. In this automaton, each tuple of local states is accepting. Then the automaton $\mathcal{A}_k^+$ is the disjoint union of the automata $\mathcal{A}_k^+(f)$ for $f \in \mathrm{part}(k, n)$. Given a partial mapping $f \in \mathrm{part}(k, n)$, the local state space (common for all processes) of the automaton $\mathcal{A}_k^+(f)$ is $Q = \mathrm{part}(k, n) \times S$. For $a \in \Sigma_i$, we let

$$\delta_{a,\emptyset} = \{(g, \tau(a, \mathrm{dom}(g))) \mid \mathrm{dom}(g) = f^{-1}(i)\}.$$

Now, let $\emptyset \neq J \subseteq [n]$ and $(g_j, s_j) \in Q$ for all $j \in J$ and let $a \in \Sigma_i$. Assume that there exists a trace $t \in \mathbb{M}'$ such that

(1) $J = p(t)$, and

(2) for all $j \in J$, $\tau(\partial_j t) = s_j$ and $\pi_2 \circ \lambda_\Gamma \circ \max(\partial_j t) = \mathrm{dom}(g_j)$.

Then, $\delta_{a,J}((g_j, s_j)_{j \in J})$ consists of all (g, s) such that

(3) $\forall j \in p \circ \max(t)$, $\exists \ell \in \mathrm{dom}(g)$ such that $g_j(\ell) = i$,

(4) $\forall \ell \in \mathrm{dom}(g)$, $\exists j \in p \circ \max(t)$ such that $g_j(\ell) = i$, and

(5) $s = \tau(t \cdot (a, \mathrm{dom}(g)))$.

We first show that this definition does not depend on the choice of a trace $t \in \mathbb{M}'$ satisfying (1) and (2).

Lemma 6.12 *Let $t, t' \in \mathbb{M}'$ be two traces satisfying (1) and (2). Then $p \circ \max(t) = p \circ \max(t')$.*
Let $N \subseteq [k]$ be such that $\forall j \in p \circ \max(t)$, $\exists \ell \in N$ with $g_j(\ell) = i$. Then $t \cdot (a, N)$ and $t' \cdot (a, N)$ are prime traces and $\tau(t \cdot (a, N)) = \tau(t' \cdot (a, N))$.

Proof. Clearly, $t = \bigvee_{j \in p(t)} \partial_j(t)$ and similarly for t'. Since τ is an asynchronous mapping, $\tau(t) = \tau(t')$ follows from $\tau\partial_j(t) = s_j = \tau\partial_j(t')$ for $j \in p(t) = J = p(t')$. By the choice of τ (Lemma 5.7), this implies $\lambda_\Gamma \circ \max(t) = \lambda_\Gamma \circ \max(t')$. In particular, $p \circ \max(t) = p \circ \max(t')$.

Let $j \in p \circ \max(t)$. We have $\mathrm{dom}(g_j) \cap N \neq \emptyset$. Since $\mathrm{dom}(g_j) = \pi_2 \circ \lambda_\Gamma \circ \max(\partial_j t)$, it follows that (a, N) and $\lambda_\Gamma \circ \max(\partial_j t)$ are dependent. Therefore $t \cdot (a, N)$ is a prime trace. Similarly, $t' \cdot (a, N)$ is prime. Since $\tau(t) = \tau(t')$ and τ is an asynchronous mapping, we deduce that $\tau(t \cdot (a, N)) = \tau(t' \cdot (a, N))$. □

We will now prove that $\mathcal{A}_k^+(f)$ satisfies the required property.

Lemma 6.13 *Let $t = (V, \leq, \lambda)$ be a $\vec{\Sigma}$-pomset and let $f \in \mathrm{part}(k, n)$. Then, a mapping $r : V \to Q$ is an R^+-run of $\mathcal{A}_k^+(f)$ iff*

(a) *$\pi_1 \circ r$ is an R^--run of $\mathcal{A}_k^-(f)$, and*

(b) *for all $x \in V$, $\pi_2 \circ r(x) = \tau(\downarrow x, \mathrm{dom} \circ \pi_1 \circ r)$.*

Proof. Throughout this proof, let $\delta_{a,J}^-$ denote the transition mappings of the ACA $\mathcal{A}_k^-$. Assume first that r is an R^+-run of $\mathcal{A}_k^+(f)$ and let $\Lambda = \mathrm{dom} \circ \pi_1 \circ r$. We show by induction on $x \in V$ that

(i) $\pi_1 \circ r$ is an R^--run of $\mathcal{A}_k^-(f)$ on $\downarrow x$, and

(ii) $\pi_2 \circ r(x) = \tau(\downarrow x, \Lambda)$.

First, let $x \in \min(t)$. Then $r(x) = (g, \tau(\lambda(x), \mathrm{dom}(g)))$ for some $g \in \mathrm{part}(k, n)$ with $\mathrm{dom}(g) = f^{-1}(p(x))$. Thus, (i) holds for x since $\pi_1 \circ r(x) = g$ with $\mathrm{dom}(g) = f^{-1}(p(x))$. The set $\downarrow x$ consists of the point x, only, and $(\lambda \times \Lambda)(x) = (\lambda(x), \mathrm{dom}(g))$. Hence $\tau(\downarrow x, \Lambda) = \tau(\lambda(x), \mathrm{dom}(g))$, i.e. (ii) holds.

Assume now that $x \in V$ is non minimal and that (i),(ii) hold for all $y < x$. Let $t' = (\Downarrow x, \Lambda)$. We will show that t' is a trace in $\mathbb{M}'$ satisfying conditions (1),(2) given before Lemma 6.12. To simplify the notation, let $(g, s) = r(x)$ and for all $j \in \mathrm{R}^+(x)$, let x_j be the maximal vertex of $\partial_j \Downarrow x$ and let $(g_j, s_j) = r(x_j)$. By the induction hypothesis, $\pi_1 \circ r$ is an R^--run of $\mathcal{A}_k^-(f)$ on $\downarrow x_j$, hence also on $\Downarrow x$. Using Lemmas 6.7 and 6.9 we deduce that $t' \in \mathbb{M}'$. By definition, $\mathrm{R}^+(x) = p(\Downarrow x) = p(t')$ and t' satisfies (1). Now, for all $j \in \mathrm{R}^+(x)$, we have $\partial_j(t') = (\downarrow x_j, \Lambda)$. Hence, using the induction hypothesis, we get $s_j = \pi_2 \circ r(x_j) = \tau(\downarrow x_j, \Lambda) = \tau(\partial_j(t'))$. Moreover, $\pi_2 \circ \lambda_\Gamma \circ \max(\partial_j t') = \Lambda(x_j) = \mathrm{dom} \circ \pi_1 \circ r(x_j) = \mathrm{dom}(g_j)$. Hence, t' satisfies (2) as well.

Now, $(g, s) \in \delta_{\lambda(x), \mathrm{R}^+(x)}((g_j, s_j)_{j \in \mathrm{R}^+(x)})$ and therefore, conditions (3),(4),(5) hold. Since $\mathrm{R}^-(x) = p \circ \max(t')$, from (3),(4) we deduce that $g \in \delta^-_{\lambda(x), \mathrm{R}^-(x)}((g_j)_{j \in \mathrm{R}^-(x)})$, i.e. (i) holds. Finally, (5) implies that $\pi_2 \circ r(x) = s = \tau(t' \cdot (\lambda(x), \mathrm{dom}(g))) = \tau(\downarrow x, \Lambda)$ and therefore (ii). This concludes the first direction of the proof, i.e. that for any R^+-run (a) and (b) hold.

Conversely, assume that the mapping $r : V \to Q$ satisfies (a) and (b) and let $\Lambda = \mathrm{dom} \circ \pi_1 \circ r$. By Lemmas 6.7 and 6.9, Λ is a k-chain mapping over t and (t, Λ) is a trace in $\mathbb{M}'$.

Let $x \in V$ be minimal and let $(g, s) = r(x)$. Since $\pi_1 \circ r$ is an R^--run of $\mathcal{A}_k^-(f)$ we have $\mathrm{dom}(g) = \mathrm{dom} \circ \pi_1 \circ r(x) = f^{-1}(p(x))$. Using (b) we get $s = \tau(\lambda(x), \mathrm{dom}(g))$ which shows that $(g, s) \in \delta_{\lambda(x), \emptyset}$.

Now, let $x \in V$ be non minimal and let $t' = (\Downarrow x, \Lambda) \in \mathbb{M}'$. To simplify the notation, let $(g, s) = r(x)$ and for all $j \in \mathrm{R}^+(x)$, let x_j be the maximal vertex of $\partial_j \Downarrow x$ and let $(g_j, s_j) = r(x_j)$. By definition, $\mathrm{R}^+(x) = p(\Downarrow x) = p(t')$ and t' satisfies (1). Now, for all $j \in \mathrm{R}^+(x)$, we have $\tau(\partial_j(t')) = \tau(\downarrow x_j, \Lambda) = \pi_2 \circ r(x_j) = s_j$ and $\pi_2 \circ \lambda_\Gamma \circ \max(\partial_j t') = \Lambda(x_j) = \mathrm{dom} \circ \pi_1 \circ r(x_j) = \mathrm{dom}(g_j)$. Hence, t' satisfies (2) as well. Moreover, since $\mathrm{R}^-(x) = p \circ \max(t')$ and $g \in \delta^-_{\lambda(x), \mathrm{R}^-(x)}((g_j)_{j \in \mathrm{R}^-(x)})$, we deduce that (3),(4) hold. Finally, using (b) we obtain $s = \pi_2 \circ r(x) = \tau(\downarrow x, \Lambda) = \tau(t' \cdot (\lambda(x), \mathrm{dom}(g)))$. Therefore, $(g, s) \in \delta_{\lambda(x),\mathrm{R}^+(x)}((g_j, s_j)_{j\in\mathrm{R}^+(x)})$. $\square$

Corollary 6.14 *For $k \in \mathbb{N}$ and $\beta \in \{+,-\}$, $\mathrm{R}^+\mathrm{F}^\beta(\mathcal{A}_k^+) = \mathcal{P}_k$.*

Proof. Let r be an F^β-successful R^+-run of $\mathcal{A}_k^+$ on the $\vec{\Sigma}$-pomset t. Then there exists $f \in \mathrm{part}(k,n)$ such that r is an R^+-run of $\mathcal{A}_k^+(f)$. By Lemma 6.13(a), $\pi_1 \circ r$ is an R^--run of $\mathcal{A}_k^-(f)$. Since any R^--run of $\mathcal{A}_k^-(f)$ is accepting, we get $t \in \mathcal{P}_k$ by Corollary 6.8.

Conversely, let $t = (V, \leq, \lambda) \in \mathcal{P}_k$ be a k-pomset. By Corollary 6.8, there exists $f \in \mathrm{part}(k,n)$ and an R^--run r^- of $\mathcal{A}_k^-(f)$ over t. By Lemma 6.7, the mapping $\Lambda = \mathrm{dom} \circ r^-$ is a k-chain mapping over t. Now, Lemma 6.9 shows that $t' = (t, \Lambda) \in \mathbb{M}'$ is a trace. Finally, the mapping $r : V \to Q$ defined by $r(x) = (r^-(x), \tau(\downarrow x, \Lambda))$ satisfies conditions (a),(b) of Lemma 6.13 and is thus an F^β-successful R^+-run of $\mathcal{A}_k^+(f)$. $\square$

A very important feature of the automaton $\mathcal{A}_k^+(f)$ is that it allows us to compute the restricted read domain R^-. More precisely, let $\emptyset \neq J \subseteq [n]$ and let $(g_j, s_j) \in Q$ for $j \in J$. We define $M((g_j, s_j)_{j\in J}) = p \circ \max(t)$ if there exists a trace $t \in \mathbb{M}'$ satisfying (1),(2). By Lemma 6.12, this is well-defined. Now, if r is an R^+-run of $\mathcal{A}_k^+(f)$ on $t = (V, \leq, \lambda)$ then for all $x \in V$, we have $\mathrm{R}^-(x) = M((r(\partial_j \Downarrow x))_{j\in\mathrm{R}^+(x)})$. Indeed, with $\Lambda = \mathrm{dom} \circ \pi_1 \circ r$ we have seen in the proof of Lemma 6.13 that $t' = (\Downarrow x, \Lambda)$ is a trace in $\mathbb{M}'$ satisfying (1),(2). By Lemma 5.7, one can then compute $\mathrm{R}^-(x) = p \circ \max(t')$ from the second component of the run $(\pi_2 \circ r(\partial_j \Downarrow x))_{j\in\mathrm{R}^+(x)} = (\tau(\partial_j \Downarrow x, \Lambda))_{j\in\mathrm{R}^+(x)}$ which proves our statement.

This will be used to prove the following proposition.

Proposition 6.15 *Let $\mathcal{B}^-$ be a $\vec{\Sigma}$-ACA, $k \in \mathbb{N}$ and $\beta \in \{+,-\}$. There exists a $\vec{\Sigma}$-ACA $\mathcal{B}^+$ such that $\mathrm{R}^+\mathrm{F}^\beta(\mathcal{B}^+, \mathcal{P}_k) = \mathrm{R}^-\mathrm{F}^\beta(\mathcal{B}^-, \mathcal{P}_k)$.*

Proof. The automaton $\mathcal{B}^+$ is meant to simulate an R^--run of $\mathcal{B}^-$ by an R^+-run. Therefore, the construction again uses the ACA $\mathcal{A}_k^+$ since as we have seen above, it allows to infer $\mathrm{R}^-(x)$ from an R^+-run of $\mathcal{A}_k^+$. So let $\mathcal{B}^- = ((Q_i^-)_{i\in[n]}, (\delta_{a,J}^-)_{a\in\Sigma,J\subseteq[n]}, F')$ and $\mathcal{A}_k^+ = ((Q)_{i\in[n]}, (\delta_{a,J})_{a\in\Sigma,J\subseteq[n]}, F)$. The local state spaces of $\mathcal{B}^+$ are given by $Q_i^+ = Q_i^- \times Q$ for $i \in [n]$. The transitions are defined by $(p', q') \in \delta_{a,J}^+((p_j, q_j)_{j\in J})$ iff

$$\begin{aligned} q' &\in \delta_{a,J}((q_j)_{j\in J}), \text{ and} \\ p' &\in \delta_{a,M}^-((p_j)_{j\in M}) \text{ with } M = M((q_j)_{j\in J}). \end{aligned}$$

Finally, a tuple $((p_j, q_j)_{j\in J})$ is accepting (i.e. in F'') iff $((p_j)_{j\in J}) \in F'$ and $((q_j)_{j\in J}) \in F$.

We will prove that $\mathrm{R}^+\mathrm{F}^\beta(\mathcal{B}^+, \mathcal{P}_k) = \mathrm{R}^-\mathrm{F}^\beta(\mathcal{B}^-, \mathcal{P}_k)$. Let $t = (V, \leq, \lambda)$ be a k-pomset. Let $r^+ : V \to \bigcup_{i\in[n]} Q_i^+$ be an F^β-successful R^+-run of $\mathcal{B}^+$ on t. Then, $\pi_2 \circ r^+$ is an R^+-run of $\mathcal{A}_k^+(f)$ on t for some $f \in \mathrm{part}(k,n)$. Using the remark above, we deduce that $\mathrm{R}^-(x) = M(\pi_2 \circ r^+(\partial_j(\Downarrow x))_{j\in\mathrm{R}^+(x)})$ for $x \in V$. Thus, $\pi_1 \circ r^+$ is an R^--run of $\mathcal{B}^-$. Since r^+ is F^β-successful, so is $\pi_1 \circ r^+$. Hence we showed the inclusion "$\subseteq$".

Conversely, let $r^- : V \to \bigcup_{i\in[n]} Q_i^-$ be an F^β-successful R^--run of $\mathcal{B}^-$ on t. By Corollary 6.14, there exists an R^+-run r of $\mathcal{A}_k^+(f)$ on t for some $f \in \mathrm{part}(k,n)$. Let $r^+ = r^- \times r$. For $x \in V$ we get immediately

$$\pi_2 \circ r^+(x) \in \delta_{\lambda(x),\mathrm{R}^+(x)}(\pi_2 \circ r^+(\partial_j(\Downarrow x))_{j\in\mathrm{R}^+(x)})$$

since $\pi_2 \circ r^+ = r$ is an R^+-run of $\mathcal{A}_k^+(f)$. Using again the remark above, we obtain $\mathrm{R}^-(x) = M(\pi_2 \circ r^+(\partial_j(\Downarrow x))_{j\in\mathrm{R}^+(x)})$. Thus r^+ is an R^+-run of $\mathcal{B}^+$ on t. Since r^- is F^β-successful, so is r^+. □

Theorem 6.16 *Let $k \in \mathbb{N}$, $\beta \in \{+,-\}$ and $L \subseteq \mathcal{P}_k$. Then the following are equivalent:*

1. *L is definable in the monadic second order logic over the alphabet Σ.*
2. *L is definable in the existential monadic second order logic over the alphabet Σ.*
3. *$L \in \mathrm{R}^-\mathrm{F}^\beta(\mathcal{P}_k)$.*
4. *$L \in \mathrm{R}^+\mathrm{F}^\beta(\mathcal{P}_k)$.*

Proof. By Corollary 6.11, the implication "1⇒3" is immediate. The implication "3⇒4" is Proposition 6.15. The implication "4⇒2" is Theorem 4.4 and the remaining implication "2⇒1" is trivial. □

Theorem 6.17 *Let $\alpha_i, \beta_i \in \{+,-\}$ for $i = 1,2$. Let $\Sigma_1, \Sigma_2, \ldots, \Sigma_n$ be mutually disjoint alphabets and $k \in \mathbb{N}$. There exists an algorithm that given two $\vec{\Sigma}$-ACAs $\mathcal{A}_1$ and $\mathcal{A}_2$ decides whether $\mathrm{R}^{\alpha_1}\mathrm{F}^{\beta_1}(\mathcal{A}_1, \mathcal{P}_k) = \mathrm{R}^{\alpha_2}\mathrm{F}^{\beta_2}(\mathcal{A}_2, \mathcal{P}_k)$.*

Proof. Using Theorem 4.4, one can effectively construct two MSO-sentences ϕ_1 and ϕ_2 such that $\mathrm{R}^{\alpha_i}\mathrm{F}^{\beta_i}(\mathcal{A}_i, \mathbb{P}(\vec{\Sigma})) = L(\phi_i)$. Since $\mathcal{P}_k$ is definable in MSO, we can assume $\mathrm{R}^{\alpha_i}\mathrm{F}^{\beta_i}(\mathcal{A}_i, \mathcal{P}_k) = L(\phi_i)$. In the proof of Lemma 6.10, we constructed an MSO sentence ψ_i over Γ from ϕ_i such that $\Pi^{-1}(L(\phi_i) \cap \mathcal{P}_k) = L(\psi_i) \cap \mathbb{M}(\Gamma, D)$, Hence, we can effectively construct formulas ψ_i of MSO over $\vec{\Gamma}$ from $\mathcal{A}_i$ such that $\mathrm{R}^{\alpha_i}\mathrm{F}^{\beta_i}(\mathcal{A}_i, \mathcal{P}_k) = \Pi(L(\psi_i) \cap \mathbb{M}(\Gamma, D))$. Hence $\mathrm{R}^{\alpha_1}\mathrm{F}^{\beta_1}(\mathcal{A}_1, \mathcal{P}_k) = \mathrm{R}^{\alpha_2}\mathrm{F}^{\beta_2}(\mathcal{A}_2, \mathcal{P}_k)$ iff $L(\psi_1) \cap \mathbb{M}(\Gamma, D) = L(\psi_2) \cap \mathbb{M}(\Gamma, D)$, and the equality of MSO-defined trace languages is decidable by [Tho90]. □

7 ACAs on general pomsets - negative results

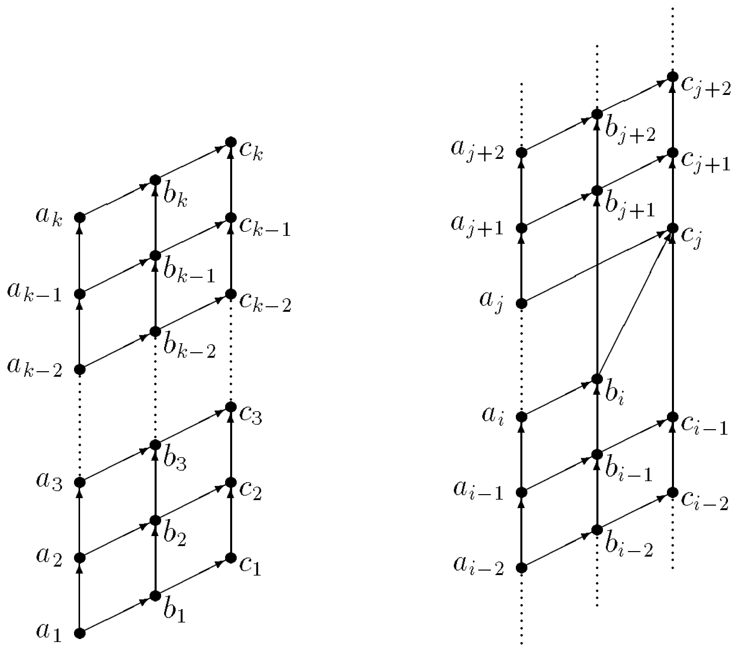
7.1 Separation of the nondeterministic classes

By Theorem 5.9 it is possible to simulate one reading mode by the other as long as we are interested in CROW-pomsets, only. There, we can even make a deterministic simulation. By Theorem 6.16, the simulation is still possible for k-pomsets. But there, one cannot simulate each deterministic ACA by another deterministic automaton in the other reading mode. Next, we will show that the simulation becomes impossible if we consider the class of all $\vec{\Sigma}$-pomsets, i.e. we will show that $\mathrm{R}^+\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ and $\mathrm{R}^-\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ are incomparable for any $\beta \in \{+,-\}$. At the end of this section, we will describe the complete inclusion structure of the classes $(\mathrm{d})\mathrm{R}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$.

Example 7.1 Let $n = 3$, $\Sigma_1 = \{a\}$, $\Sigma_2 = \{b\}$ and $\Sigma_3 = \{c\}$. Let L consist of all $\vec{\Sigma}$-pomsets $t = (V, \leq, \lambda)$ where no c-labeled element covers an a-labeled one. Then L is in $\mathrm{dR}^-\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$, but not in $\mathrm{R}^+\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ for any $\beta \in \{+,-\}$.

Proof. By Example 3.2, $L \in \mathrm{dR}^-\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$.

We show $L \notin \mathrm{R}^+\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ by contradiction: Assume $\mathcal{A}$ is a $\vec{\Sigma}$-ACA with $L = \mathrm{R}^+\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma}))$. Now let $k = |Q_2| \cdot |Q_3| + 2$ and consider the $\vec{\Sigma}$-pomset $t = (V, \leq, \lambda)$ with $V = \{a_\ell, b_\ell, c_\ell \mid \ell = 1, 2, \ldots, k\}$, $a_\ell \leq a_m$, $\{a_\ell, b_\ell\} \leq b_m$ and $\{a_\ell, b_\ell, c_\ell\} \leq c_m$ iff $\ell \leq m$ and no further comparabilities (cf. Figure 3). The labeling λ is defined canonically.



The $\vec{\Sigma}$-pomset t The $\vec{\Sigma}$-pomset t'

Figure 3: cf. Proof of Example 7.1

Since t is in L, there is an F^β-successful R^+-run r of $\mathcal{A}$ on t. Since $k > |Q_2| \cdot |Q_3| + 1$, there are $i < j < k$ such that $r(b_i) = r(b_j)$ and $r(c_{i-1}) = r(c_{j-1})$. We erase the vertices $b_{i+1}, \ldots, b_j, c_i, \ldots, c_{j-1}$ from t obtaining the $\vec{\Sigma}$-pomset t' (cf. Figure 3).

Note that t' is not in L since c_j covers a_j. Let $r' = r \restriction V'$. We show that r' is an R^+-run on t: Recall that the read domain of any element $x \in V'$ is $\mathrm{R}^+(x) = \{\partial_i(\Downarrow x) \mid i = 1, 2, 3\}$. This set differs in t and in t' for $x = b_{j+1}$ and for $x = c_j$, only.

Thus, for $x \notin \{b_{j+1}, c_j\}$, we have $r'(x) \in \delta_{\lambda(x),\mathrm{R}^+(x)}(r'(\partial_i(\Downarrow x)_{i\in\mathrm{R}^+(x)})$. But this holds for $x \in \{b_{j+1}, c_j\}$ also since $r(b_i) = r(b_j)$ and $r(c_{i-1}) = r(c_{j-1})$. Thus, r' is an R^+-run of $\mathcal{A}$ on t'. Since the two $\vec{\Sigma}$-pomsets coincide on their top layer, it is F^β-successful, implying $t' \in \mathrm{R}^+\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma}))$. But this contradicts our assumption $L = \mathrm{R}^+\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma}))$. □

Example 7.2 Let $n = 4$, $\Sigma_1 = \{a\}$, $\Sigma_2 = \{b\}$, $\Sigma_3 = \{c\}$ and $\Sigma_4 = \{d\}$. Let L consist of all $\vec{\Sigma}$-pomsets $t = (V, \leq, \lambda)$ such that any $x \in V$ with $\lambda(x) = d$ dominates an even number of a-labeled elements. Then L is in $\mathrm{dR}^+\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$, but not in $\mathrm{R}^-\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ for any sign β.

Proof. In Example 3.3 we showed that there is a deterministic $\vec{\Sigma}$-ACA $\mathcal{A}$ that allows an R^+-run on a pomset t iff $t \in L$. Thus, when all tuples of local states in this automaton are accepting, we get $L \in \mathrm{dR}^+\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$.

It remains to show that L is not in $\mathrm{R}^-\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$. By contradiction, assume $\mathcal{A}$ is a $\vec{\Sigma}$-ACA such that $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma})) = L$ for $\beta = +$ or $\beta = -$. Let $k = |Q_2| \cdot |Q_4| + 2$ and consider the $\vec{\Sigma}$-pomset $t = (V, \leq, \lambda)$ (cf. Figure 4) with $V = \{a_\ell \mid \ell = 1, 2, \ldots, 2k\} \cup \{b_\ell, c_\ell, d_\ell \mid \ell = 1, 2, \ldots, k\}$ and the natural labeling λ. The nontrivial part of the order relation $\leq$ is defined by

$$\begin{aligned} b_\ell, c_\ell \leq d_m &\quad \text{iff} \quad \ell \leq m \\ a_\ell \leq b_m, d_m &\quad \text{iff} \quad 2\ell \leq m \text{ and} \\ a_\ell \leq c_m &\quad \text{iff} \quad 2\ell - 1 \leq m. \end{aligned}$$

Note that t is in L since d_ℓ dominates 2ℓ elements labeled by a for any ℓ. Hence there is an F^β-successful R^--run r of $\mathcal{A}$ on t. Since $k > |Q_2| \cdot |Q_4| + 1$, there are $i < j < k$ such that $r(b_i) = r(b_j)$ and $r(d_{i-1}) = r(d_{j-1})$. We define a second $\vec{\Sigma}$-pomset $t' = (V', \leq', \lambda')$ from t by erasing the elements b_ℓ and $d_{\ell-1}$ for $\ell = i+1, i+2, \ldots, j$ (cf. Figure 4).

More formally, $V' = V \setminus \{b_\ell, d_{\ell-1} \mid \ell = i+1, i+2, \ldots, j\}$ and λ' is the restriction of λ to V'. The partial order relation $\leq'$ is the reflexive and transitive closure of the following relations (where ℓ and m range over all suitable values):

$$\begin{aligned} b_\ell, c_\ell \leq' d_m &\quad \text{iff} \quad \ell \leq m \\ a_\ell \leq' b_m &\quad \text{iff} \quad 2\ell \leq m \text{ and} \\ a_\ell \leq' c_m &\quad \text{iff} \quad 2\ell - 1 \leq m. \end{aligned}$$

Note that $a_\ell \leq' d_m$ iff there is some k with $a_\ell \leq' b_k \leq' d_m$ or $a_\ell \leq' c_k \leq' d_m$. In particular $a_{2j} \parallel d_j$ since b_j does not belong to V'. Hence d_j dominates the a-labeled elements $a_1, \ldots, a_{2j-1}$ in t', i.e. an odd number, only. Hence t' is not in L.

Finally, since $r(b_i) = r(b_j)$ and $r(d_{i-1}) = r(d_{j-1})$ one can easily check that $r' := r \restriction V' : V' \to \bigcup_{i\in[4]} Q_i$ is an F^β-successful R^--run of $\mathcal{A}$ on t'. Since t and t' coincide on their top layer, it is an F^β-successful R^--run on t' since r is one on t. Thus, $t' \in \mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma}))$ contradicting $\mathrm{R}^-\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma})) = L$ since t' is not in L. □

Theorem 7.3 *In Figure 5 and 6, arrows denote proper inclusions, and no further inclusions hold.*

The $\vec{\Sigma}$-pomset t The $\vec{\Sigma}$-pomset t'

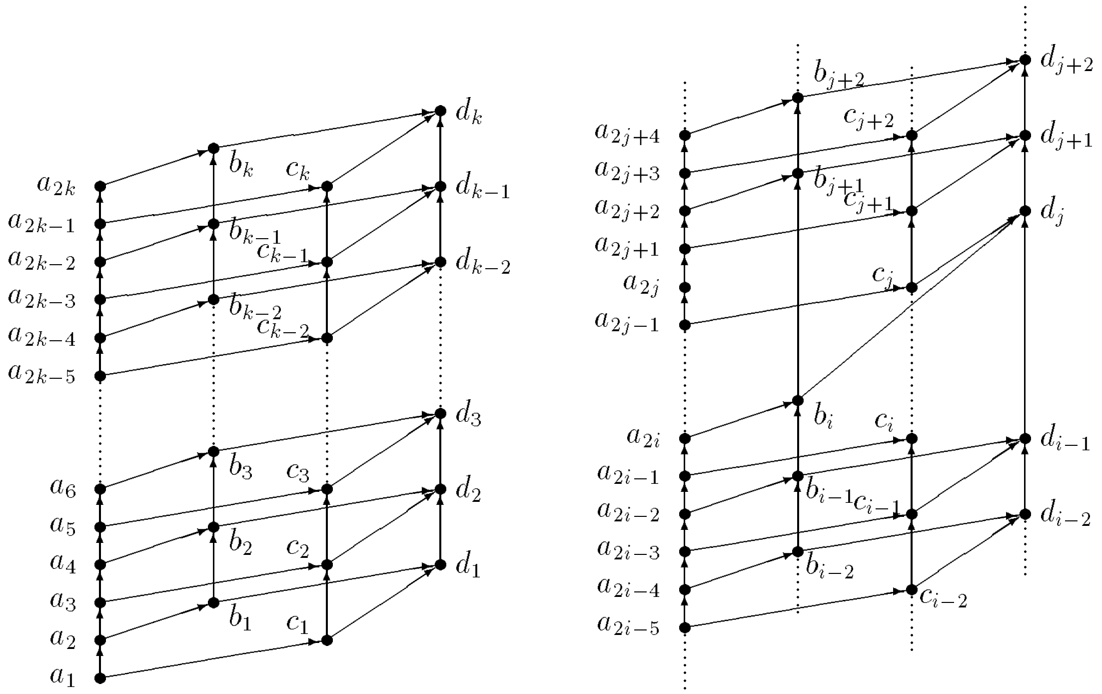


Figure 4: cf. Proof of Example 7.2.

Proof. We start with the classes in $\mathbb{P}(\vec{\Sigma})$: We first explain why the inclusions hold. By Theorem 4.4, the class at the top of MSO-definable languages subsumes all classes $(\mathrm{d})\mathrm{R}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$. The equalities $\mathrm{R}^\alpha\mathrm{F}^-(\mathbb{P}(\vec{\Sigma})) = \mathrm{R}^\alpha\mathrm{F}^+(\mathbb{P}(\vec{\Sigma}))$ are proven in Theorem 4.1. The inclusions $\mathrm{dR}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma})) \subseteq \mathrm{R}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ are trivial and the inclusions of the least class in all deterministic classes is Proposition 4.3.

Then, we show that the inclusions are proper. By Examples 7.1 and 7.2, the classes $\mathrm{R}^+\mathrm{F}^-(\mathbb{P}(\vec{\Sigma}))$ and $\mathrm{R}^-\mathrm{F}^-(\mathbb{P}(\vec{\Sigma}))$ are incomparable. Hence they are proper subclasses of the class of MSO-definable languages. Similarly, by Proposition 6.4, the deterministic classes $\mathrm{dR}^\alpha\mathrm{F}^+(\mathbb{P}(\vec{\Sigma}))$ and $\mathrm{dR}^\alpha\mathrm{F}^-(\mathbb{P}(\vec{\Sigma}))$ are incomparable. Hence they are proper subclasses of $\mathrm{R}^\alpha\mathrm{F}^-(\mathbb{P}(\vec{\Sigma}))$ and the least class is properly contained in each of them.

Finally, it remains to show that no further inclusions hold. We have already seen that the deterministic classes are incomparable (Proposition 6.4). It remains to prove that $\mathrm{dR}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ is not contained in $\mathrm{R}^\gamma\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ for $\alpha \neq \gamma$. But this follows from Example 7.2 and 7.1.

Now we deal with the classes in $\mathcal{P}_k$: The inclusions from above are inherited. In addition, any definable language in $\mathcal{P}_k$ can be accepted by an ACA by Theorem 6.16. By Proposition 6.4, the deterministic classes $\mathrm{dR}^\alpha\mathrm{F}^\beta(\mathcal{P}_k)$ and $\mathrm{dR}^{\alpha'}\mathrm{F}^{\beta'}(\mathcal{P}_k)$ are incomparable. Hence they are proper subclasses of the top class and the least class is properly contained in each of them. $\square$

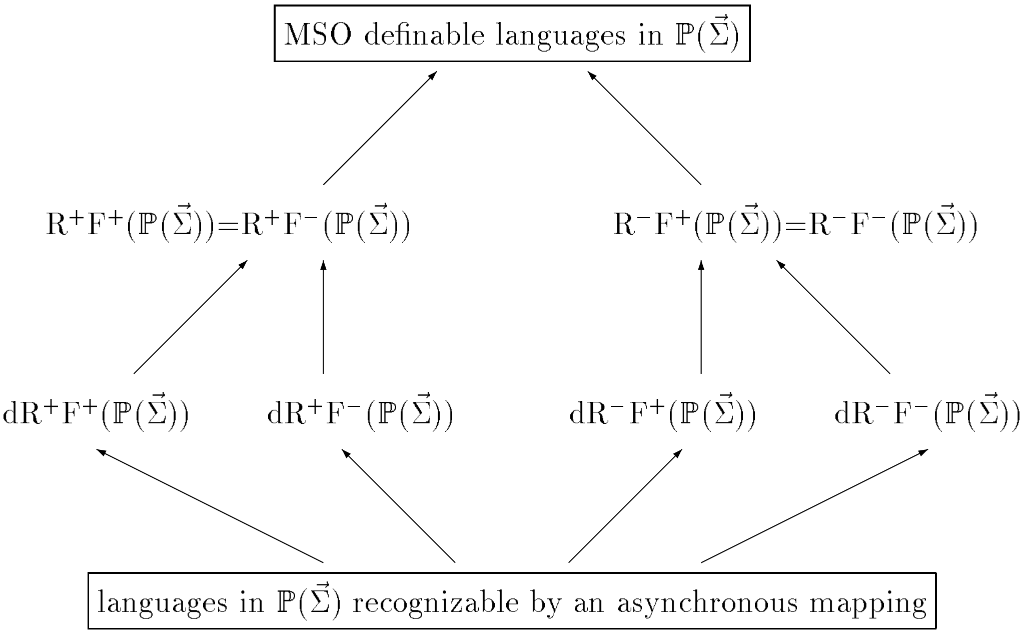


Figure 5: The inclusion structure in $\mathbb{P}(\vec{\Sigma})$ for $n \geq 4$

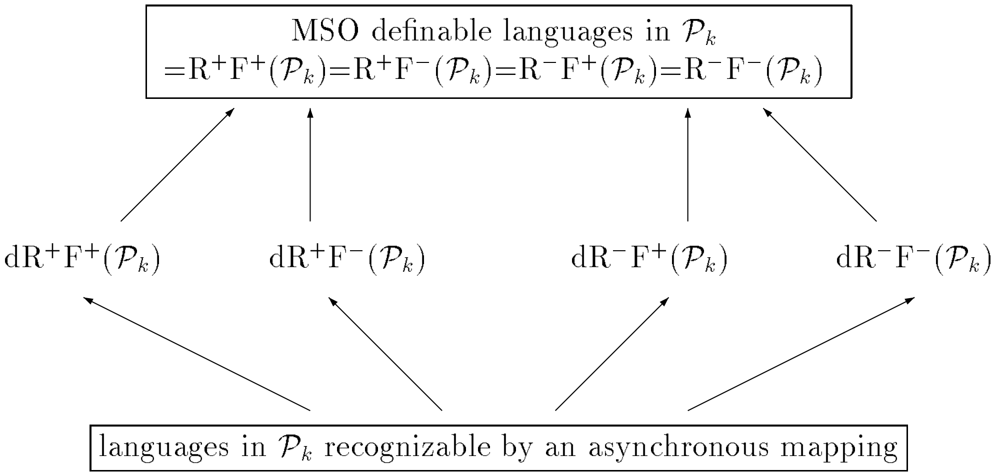


Figure 6: The inclusion structure in $\mathcal{P}_k$ for $n \geq 3$ and $k \geq 2$

Recall that for CROW-pomsets (as well as for traces), all the classes in the pictures above coincide.

7.2 $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ is not closed under complement

Let L consist of all pomsets $(V, \leq, \lambda)$ over $(\{a\}, \{b\})$ such that any a-labeled vertex in V is covered by a b-labeled one. We prove that this set is not recognizable. This language seems to be a typical example of a language that is definable in MSO but not recognizable. The reason is that the condition that is posed requires something in the future of each vertex. But the automata work "bottom-up" on the pomset and have no knowledge of the future. On the other hand, we show that the complement of L is recognizable thereby showing that the set of recognizable languages is not closed under complement. This gives another proof of the fact that not every nondeterministic ACA can be transformed into an equivalent deterministic one.

Proposition 7.4 *Let $\alpha, \beta \in \{+, -\}$. Then L is not contained in* $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$.

Proof. By contradiction, let $\mathcal{A}$ be a $\vec{\Sigma}$-ACA such that $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A}, \mathbb{P}(\vec{\Sigma})) = L$, let Q_1 be the set of local states of the first process of $\mathcal{A}$ and let $m = |Q_1| + 2$.

Now consider the $\vec{\Sigma}$-pomset $t = (V, \leq, \lambda)$ defined as follows (cf. Figure 7 for the case $m = 8$): The set V equals $\{a_i, b_i \mid i = 1, 2, \ldots, m\}$ with the natural labeling and the partial order relation is defined by $a_i \leq a_j$, $a_i \leq b_j$, $b_i \leq b_j$ iff $i \leq j$.

The $\vec{\Sigma}$-pomset t:

The $\vec{\Sigma}$-pomset t':

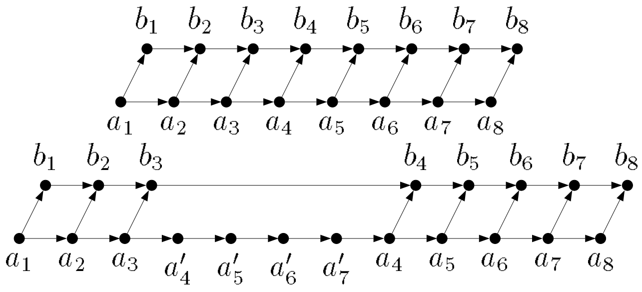


Figure 7: cf. Proof of Proposition 7.4.

Note that $t \in L$. Hence there exists an F^β-successful R^α-run r of $\mathcal{A}$ on t. Since $m - 1 > |Q_1|$, there exist $i < j \leq m - 1$ such that $r(a_i) = r(a_j)$.

Secondly, consider the $\vec{\Sigma}$-pomset $t' = (V', \leq, \lambda')$ obtained from t by the insertion of a chain of length $j - i$ between a_i and a_{i+1} labeled by a. More formally, we define: $V' = V \cup \{a'_{i+1}, a'_{i+2}, \ldots, a'_j\}$, $\lambda' \restriction V = \lambda$ and $\lambda'(a'_k) = a$ for all suitable k. Furthermore, the only additional comparabilities are those induced by $a_i < a'_{i+1} < a'_{i+2} < \cdots < a'_j < a_{i+1}$ (cf. Figure 7 for $i = 3$ and $j = 7$). Let $r' : V' \to \bigcup_{i=1}^n Q_i$ be defined by $r' \restriction V = r$ and $r'(a'_k) = r(a_k)$ for all suitable k. Then one can easily check that r' is an F^β-successful R^α-run on t', i.e. $t' \in \mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A})$ contradicting the assumption $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A}) = L$. $\square$

Note that the language L is definable in first order logic. Hence the proposition above implies that $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$ does not contain all first-order definable languages.

Theorem 7.5 *For $n \geq 2$, the set $\mathrm{R}^\alpha\mathrm{F}^\beta(\vec{\Sigma})$ is not closed under complement.*

Proof. Note that L^{co} consists of all $(\{a\},\{b\})$-pomsets containing an a-labeled element that is not covered by a b-labeled one. We construct a $\vec{\Sigma}$-ACA that recognizes the language. Let $Q_1 = \{0,1,2\}$ and $Q_2 = \{0,2\}$. Initially, the processes will use the state 0 to label vertices. At some point the first process will have to guess the a-labeled vertex which is not covered by a b-labeled one. To do so, it marks this vertex by the state 1. If the second process reads the state 1 then the run cannot proceed and will thus be rejected. Otherwise the processes label by 2 the vertices which dominate the guessed vertex labeled by 1 and the run will be accepted.

More formally, the transition functions are defined by:

$$\delta_{a,J}((q_j)_{j\in J}) = \begin{cases} \{0,1\} & \text{if } q_j = 0 \text{ for all } j \in J, \\ \{2\} & \text{otherwise} \end{cases}$$

$$\delta_{b,J}((q_j)_{j\in J}) = \begin{cases} \{0\} & \text{if } q_j = 0 \text{ for all } j \in J, \\ \emptyset & \text{if } 1 \in J \text{ and } q_1 = 1, \\ \{2\} & \text{otherwise.} \end{cases}$$

Note that only the first process of $\mathcal{A}$ is nondeterministic. Finally, the set of accepting tuples is $F = \{(q_j)_{j\in J} \mid q_j \neq 0 \text{ for some } j \in J\}$ so that it is ensured that the guess occurred once. One can check that $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathcal{A},\mathbb{P}(\vec{\Sigma})) = L^{co}$ for any $\alpha,\beta \in \{+,-\}$. Since, by Proposition 7.4 L is not in $\mathrm{R}^\alpha\mathrm{F}^\beta(\mathbb{P}(\vec{\Sigma}))$, the theorem follows. $\square$

References

[CMZ93] R. Cori, Y. Métivier, and W. Zielonka. Asynchronous mappings and asynchronous cellular automata. *Information and Computation*, 106:159–202, 1993.

[DG96] M. Droste and P. Gastin. Asynchronous cellular automata for pomsets without autoconcurrency. In *CONCUR'96*, Lecture Notes in Comp. Science vol. 1119, pages 627–638. Springer, 1996.

[Die94] V. Diekert. A partial trace semantics for Petri nets. *Theoretical Computer Science*, 134:87–105, 1994.

[DM95] V. Diekert and A. Muscholl. Construction of asynchronous automata. In *[DR95]*, pages 249–267. 1995.

[DR95] V. Diekert and G. Rozenberg. *The Book of Traces*. World Scientific Publ. Co., 1995.

[Eil74] S. Eilenberg. *Automata, Languages and Machines vol. A*. Academic Press, New York, 1974.

[Gis88] J.L Gischler. The equational theory of pomsets. *Theoretical Comp. Science*, 61:199–224, 1988.

[Kus98] D. Kuske. Asynchronous cellular automata and asynchronous automata for pomsets. In *CONCUR'98*, Lecture Notes in Comp. Science vol. 1466, pages 517–532. Springer, 1998.

[Maz77] A. Mazurkiewicz. Concurrent program schemes and their interpretation. Technical report, DAIMI Report PB-78, Aarhus University, 1977.

[Maz86] A. Mazurkiewicz. Trace theory. In *Advanced Course on Petri Nets*, Lecture Notes in Comp. Science, pages 279–324. Springer, 1986.

[Mus96] A. Muscholl. *Über die Erkennbarkeit unendlicher Spuren*. Teubner Stuttgart und Leipzig, 1996.

[Pra86] V. Pratt. Modelling concurrency with partial orders. *Int. J. of Parallel Programming*, 15:33–71, 1986.

[Sta81] P.H. Starke. Processes in Petri nets. *Elektronische Informationsverarbeitung und Kybernetik*, 17:389–416, 1981.

[Tho90] W. Thomas. On logical definability of trace languages. In V. Diekert, editor, *Proceedings of a workshop of the ESPRIT BRA No 3166: Algebraic and Syntactic Methods in Computer Science (ASMICS) 1989*, Report TUM-I9002, Technical University of Munich, pages 172–182, 1990.

[Zie87] W. Zielonka. Notes on finite asynchronous automata. *R.A.I.R.O. - Informatique Théorique et Applications*, 21:99–135, 1987.

[Zie89] W. Zielonka. Safe executions of recognizable trace languages by asynchronous automata. In A.R. Meyer et al., editor, *Logical Foundations of Computer Science*, Lecture Notes in Comp. Science vol. 363, pages 278–289, 1989.