Chapter 6

Other automata models for pomsets

The covering relation of a pomset without autoconcurrency is a Σ-dag. This allows us to speak of the set of pomsets that is accepted by an asychronous cellular automaton: A pomset $(V, \leq, \lambda)$ is accepted by the Σ-ACA $\mathcal{A}$ iff its Hasse-diagram $(V, -\!\!<, \lambda)$ belongs to $L(\mathcal{A})$. Actually, this was the original intention when asynchronous cellular automata were generalized from dependence graphs to more general structures in [DG96] (cf. also [Kus98, DGK00]).

For pomsets, other automata models have been considered in the literature. In particular, Arnold considered P-asynchronous automata [Arn91] and Lodaya & Weil dealt with branching automata [LW98a, LW98b, LW00]. The primary aim of this chapter is to compare the expressive power of these automata with the expressive power of our Σ-ACAs. This is achieved by a comparision of the expressive power of these automata models with that of monadic second order logic.

6.1 Branching automata by Lodaya & Weil

In several papers, Lodaya & Weil considered branching automata and proved results analogous to Kleene's and to Myhill-Nerod's Theorems [LW98b, LW98a, LW00]. Their automata work on so called series-parallel pomsets, sp-pomsets for short, defined as follows: A labeled partial order $(V, \leq, \lambda)$ is a *sp-pomset* if the partially ordered set $(N, \leq_N)$ cannot be embedded into $(V, \leq)$ (cf. Figure 6.1). To give an alternative description of sp-pomsets (that also explains the name) we need some more notation: Let $t_1 = (V_1, \leq_1, \lambda_1)$ and $t_2 = (V_2, \leq_2, \lambda_2)$ be labeled partial orders with $V_1 \cap V_2 = \emptyset$. The *serial product* $t_1 \cdot t_2$ of them is the labeled partial order

$$(V_1 \cup V_2, \leq_1 \cup V_1 \times V_2 \cup \leq_2, \lambda_1 \cup \lambda_2).$$

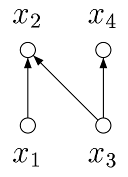

Figure 6.1: The partially ordered set $(N, \leq)$

Thus, in $t_1 \cdot t_2$, the pomset t_2 is put on top of the pomset t_1. On the contrary, the *parallel product* $t_1 \parallel t_2$ is defined to be

$$(V_1 \cup V_2, \leq_1 \cup \leq_2, \lambda_1 \cup \lambda_2),$$

i.e. here the two partial orders are set side by side. Now it is a result in the folklore of order theory that a partially ordered set is series-parallel iff it can be constructed from the one-point partial orders by the application of the operations $\cdot$ and $\parallel$ (cf. [Gis88]). In other words, the set of all sp-pomsets SP(Σ) over the alphabet Σ is the least class of Σ-labeled partial orders containing the one-point pomsets that is closed under the application of the serial product $\cdot$ and the parallel product $\parallel$.

Lodaya & Weil introduced an automata model that is suitable to accept sp-pomsets:

Definition 6.1.1 A *branching automaton* is a tuple $\mathcal{B} = (S, T_s, T_f, T_j, I, A)$ where
S is a finite set of states,
I and A are subsets of S of initial and accepting states, respectively,
$T_s \subseteq S \times \Sigma \times S$ is the set of *sequential transitions*,
$T_f \subseteq S \times (S \times S)$ is the set of *fork transitions*, and
$T_j \subseteq (S \times S) \times S$ is the set of *join transitions*.

Branching automata were introduced in [LW98b] in a slightly more general form (they allowed branching automata to branch into more than two subprocesses), but the definition given here yields the same expressive power. Since branching automata are meant to run on sp-pomsets, their runs can be defined by induction on the construction of sp-pomsets from the one-point pomsets by serial and parallel product. Let $p, r \in S$ be states of the branching automaton $\mathcal{B}$ and let a be the one-point a-labeled pomset. Then there is a run from p to r on a (denoted $p \xrightarrow{a} r$) iff $(p, a, r) \in T_s$. Now let s, t be sp-pomsets. Then there is a run $p \xrightarrow{s \cdot t} r$ iff there exists a state $q \in S$ and runs $p \xrightarrow{s} q$ and $q \xrightarrow{t} r$. Finally, there is a run $p \xrightarrow{s \parallel t} r$ iff there are states $p_1, p_2, r_1, r_2 \in S$, a fork transition $(p, (p_1, p_2)) \in T_f$, runs $p_1 \xrightarrow{s} r_1$ and $p_2 \xrightarrow{t} r_2$, and a join transition

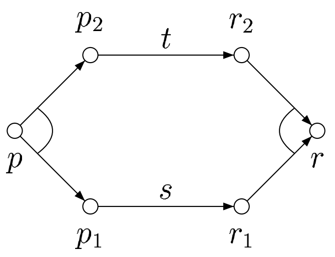

Figure 6.2: A run on $s \parallel t$

$((r_1, r_2), r) \in T_j$ (this definition is visualized in Figure 6.2, edges that form a fork or a join transition are connected by an angle at their sources or their targets). An sp-pomset s is *accepted* by $\mathcal{B}$ iff there exist $\iota \in I$ and $q \in A$ and a run $\iota \xrightarrow{s} q$. By $L(\mathcal{B})$ we denote the set of sp-pomsets accepted by $\mathcal{B}$.

To describe the expressive power of branching automata in the spirit of Kleene's theorem [Kle56], Lodaya & Weil introduced several classes of rational expressions for sets of sp-pomsets:

For $S, T \subseteq \mathrm{SP}(\Sigma)$, let

$$\begin{aligned} S \cdot T &:= \{s \cdot t \mid s \in S, t \in T\}, \\ S \parallel T &:= \{s \parallel t \mid s \in S, t \in T\}, \\ S^\star &:= \{s_1 \cdot s_2 \cdot s_3 \cdots s_n \mid n \geq 1, s_i \in S\}, \text{ and} \\ S^\oplus &:= \{s_1 \parallel s_2 \parallel s_3 \cdots \parallel s_n \mid n \geq 1, s_i \in S\}. \end{aligned}$$

A set $S \subseteq \mathrm{SP}(\Sigma)$ is *rational* if it can be constructed from the finite subsets of $\mathrm{SP}(\Sigma)$ by the operations $\cup$, $\cdot$, $\parallel$, *, and $^\oplus$. It is *weakly rational* if the operation $^\oplus$ is applied to languages of the form $K \cdot L$, only. Finally it is *series-rational* if it can be constructed from the finite subsets of $\mathrm{SP}(\Sigma)$ by the operations $\cup$, $\cdot$, $\parallel$, and * (i.e. without the parallel iteration). Clearly, any series-rational language is weakly rational and any weakly rational language is rational. These two implications cannot be inverted for $(a \parallel a)^\oplus$ is rational and not weakly rational and $a^\oplus$ is weakly rational but not series-rational. Since in the construction of series-rational languages the parallel iteration cannot appear, for any series-rational language S there exists an $n \in \mathbb{N}$ with $w(s) \leq n$ for any $s \in S$, i.e. any series-rational language is *width-bounded*.

The set $\mathrm{SP}(\Sigma)$ can be seen as an algebra with the two associative operations $\cdot$ and $\parallel$. A structure $(S, \cdot, \parallel)$ is an *sp-algebra* [LW98a] if $\cdot$ and $\parallel$ are associative binary operations on S and $\parallel$ is in addition commutative. A set $L \subseteq \mathrm{SP}(\Sigma)$ of N-free pomsets is *algebraically recognizable* if there exists a finite sp-algebra $(S, \cdot, \parallel)$ and a homomorphism $\eta : (\mathrm{SP}(\Sigma), \cdot, \parallel) \to (S, \cdot, \parallel)$ such that $L = \eta^{-1}\eta(L)$. Lodaya & Weil showed the following relation between the concepts introduced so far:

Theorem 6.1.2 ([LW00]) *Let $L \subseteq \mathrm{SP}(\Sigma)$. Then the following are equivalent*

1. *L is series-rational.*
2. *L is algebraically recognizable and width-bounded.*
3. *L can be accepted by a branching automaton and is width-bounded.*

Our next aim is to complete this picture concerning the recognizable languages of sp-pomsets in the spirit of Büchi's theorem, i.e., we want to relate the expressive power of monadic second order logic to that of branching automata.

Example 6.1.3 Let a denote the a-labeled one-point pomset for any $a \in \Sigma$. Then $S = (a \parallel a)^{\oplus}$ is a rational language. It consists of all antichains of an even number of a-labeled vertices. In particular, S is not width-bounded and therefore not series-rational. Furthermore, S cannot be monadically axiomatizable since it is impossible to axiomatize the finite sets of even size in this logic relative to all finite sets.

The example above shows that not every rational sp-language can be monadically axiomatized. It is not clear which additional features should be adjoint to MSO to obtain precisely the expressive power of rational sp-languages. Even though there are rational languages that cannot be monadically axiomatized, this does not occur in the context of weakly rational languages:

Proposition 6.1.4 *Let L be a weakly rational language. Then there exists a monadic sentence σ such that $L = \{t \in \mathrm{SP}(\Sigma) \mid t \models \sigma\}$.*

Proof. Clearly, any finite set of sp-pomsets can be monadically axiomatized. Now let S and T be two sets of sp-pomsets axiomatized by the monadic sentences σ and τ, respectively. Then $S \cup T$ is axiomatized by $\sigma \vee \tau$. The set $S \parallel T$ consists of all sp-pomsets satisfying

$$\exists X(\forall x \forall y(x \in X \wedge y \notin X \to x \parallel y) \wedge \sigma \restriction X \wedge \tau \restriction X^{co})$$

where $\sigma \restriction X$ is the restriction of σ to the set X and $\tau \restriction X^{co}$ that of τ to the complement of X. Similarly, $S \cdot T$ is axiomatized by

$$\exists X(\forall x \forall y(x \in X \wedge y \notin X \to x < y) \wedge \sigma \restriction X \wedge \tau \restriction X^{co}).$$

Next we show that $S^{\star}$ can be described by a monadic sentence: The idea of a sentence axiomatizing $S^{\star}$ is to color the vertices of an sp-pomset s by two colors such that the coloring corresponds to a factorization in factors $s = s_1 \cdot s_2 \cdot s_3 \cdots s_n$ where every factor s_i belongs to S. The identification of the S-factors will be provided by the property of being a maximal convex one-colored set. More formally, we define $\varphi = \exists X \exists Y(\varphi_1 \wedge \varphi_X \wedge \varphi_Y)$ where φ_1 asserts that X and Y

form a partition of the set of vertices. The formula φ_X states that the maximal subsets of X that are convex satisfy σ, i.e.

$$\begin{aligned}\varphi_X \quad = \forall Z(\quad & Z \subseteq X \land Z \text{ is convex} \land \\ & \forall Z'(Z \subseteq Z' \subseteq X \land Z' \text{ is convex} \to Z = Z') \\ & \;\to \sigma \restriction Z)\end{aligned}$$

and the formula φ_Y is defined similarly with Y taking the place of X.

Finally, we have to deal with the parallel iteration $\oplus$. Recall that it is applied to languages of the form $S = S_1 \cdot S_2$, only. Hence, any element of the iterated language S (as a partial order) is connected. Thus, the informal sentence

$$\varphi = \forall Z(Z \text{ is connected } \to \sigma \restriction Z)$$

axiomatizes $S^\oplus$. Since in monadic second order logic the transitive closure of the comparability relation $\leq \cup \geq$ can be defined, one can express that a set is connected. Hence, $S^\oplus$ can be monadically axiomatized whenever S is the product of two monadically axiomatizable languages. □

Thus, rational languages are not necessarily monadically axiomatizable, but weakly rational languages and therefore series-rational languages are monadically axiomatizable. Hence, by Theorem 6.1.2, any algebraically recognizable and width-bounded set of sp-pomsets is monadically axiomatizable. The inverse implication is based on the following model theoretic notion: Let $s, t \in \mathrm{SP}(\Sigma)$ be sp-pomsets. Then s and t are *monadically indistinguishable of level* n ($s \equiv_n t$), if we have $s \models \varphi \iff t \models \varphi$ for any monadic sentence of quantifier depth at most n. The relation $\equiv_n$ is obviously an equivalence relation on $\mathrm{SP}(\Sigma)$. Furthermore, it has only finitely many equivalence classes since there are, up to logical equivalence, only finitely many monadic sentences of quantifier depth at most n. A simple application of monadic Ehrenfeucht-Fraïssé-games yields that $\equiv_n$ is even a congruence on $(\mathrm{SP}(\Sigma), \cdot, \|)$ (cf. [EF91, Proposition 2.1.4]). This fact implies

Theorem 6.1.5 *Let $L \subseteq \mathrm{SP}(\Sigma)$ be width-bounded. Then the following are equivalent*

1. *L is monadically axiomatizable.*
2. *L can be accepted by a branching automaton.*

Proof. The implication 2⇒1 is a special case of Proposition 6.1.4 together with Theorem 6.1.2. For the other implication, let L be axiomatized by the monadic sentence φ. Let n be the quantifier depth of φ. Furthermore, let $(S, \cdot, \|)$ be the quotient of the sp-algebra $(\mathrm{SP}(\Sigma), \cdot, \|)$ w.r.t. $\equiv_n$. Then $\mathrm{nat} : \mathrm{SP}(\Sigma) \to S$ is a homomorphism such that $s \models \varphi \iff \varphi \in \mathrm{nat}(s) \iff \varphi \in \mathrm{nat}(t) \iff t \models \varphi$ for any $s, t \in \mathrm{SP}(\Sigma)$ with $s \in L$ and $t \in \mathrm{nat}^{-1}\mathrm{nat}(s)$. Hence the homomorphism nat recognizes L, i.e., L is algebraically recognizable and can therefore be accepted by a branching automaton by Theorem 6.1.2. □

So far, we proved in this section a Büchi-type theorem for sp-pomsets which answers a question left open in [LW00]. Next, we will investigate the relation between the expressive power of branching automata and asynchronous-cellular automata: Branching automata accept pomsets which, in general, are no Σ-dag. Therefore, we will consider the set of Hasse-diagrams of pomsets accepted by a given branching automaton. Conversely, an ACA can accept Σ-dags that are not Hasse-diagrams of pomsets. But, for any ACA $\mathcal{A}$, we find a branching automaton $\mathcal{B}$ accepting the same Hasse-diagrams of sp-pomsets as $\mathcal{A}$ does:

Corollary 6.1.6 *Let $\mathcal{A}$ be a Σ-ACA. Then there exists a branching automaton $\mathcal{B}$ such that* $\mathrm{Ha}(L(\mathcal{B})) = L(\mathcal{A}) \cap \mathrm{Ha}(\mathrm{SP}(\Sigma))$.

Proof. By Theorem 5.1.1, the set $L(\mathcal{A})$ is a monadically axiomatizable set of Σ-dags. Similarly, $\mathrm{Ha}(\mathrm{SP}(\Sigma))$ is monadically axiomatizable. Hence there is a monadic sentence φ that is satisfied by an sp-pomset t iff the Hasse-diagram of t belongs to $L(\mathcal{A})$, i.e., $L(\mathcal{A}) \cap \mathrm{Ha}(\mathrm{SP}(\Sigma)) = \{\mathrm{Ha}(t) \mid t \in \mathrm{SP}(\Sigma) \mid t \models \varphi\}$. Now Theorem 6.1.5 ensures the existence of a branching automaton $\mathcal{B}$ as required. □

Thus, any asynchronous-cellular automaton can be simulated by a branching automaton. To show that the converse holds as well, we first prove that sp-pomsets permit k-chain-coverings:

Lemma 6.1.7 *Let $n \in \mathbb{N}$. Then there exists a natural number k such that* $\mathrm{Ha}(t) \in \mathbb{D}_k$ *for any sp-pomset $t \in \mathrm{SP}(\Sigma)$ of width at most n.*

Proof. Let $t = (V, \leq, \lambda) \in \mathrm{SP}(\Sigma)$ be of width at most n. We show that any totally unconnected set in spine(Ha(t)) (cf. page 61) contains at most $2n(n+1)-1$ elements: By contradiction, let $X \subseteq V$ be a totally unconnected set in spine(Ha(t)) of size $2n(n+1)$. Then it contains a chain of size $2n+2$ since n is the maximal size of an antichain in t. Let $x_1 < x_2 < \cdots < x_{2n+2}$ be a chain in X. Since x_{n+1} and x_{n+2} are unconnected in spine(Ha(t)), there is $y \in V$ with $x_{n+1} \longrightarrow\!\!< y$ that is incomparable with x_{n+2}, or there exists $z \in V$ with $z \longrightarrow\!\!< x_{n+2}$ that is incomparable with x_{n+1}.

Suppose that both, y and z, exist. If they are incomparable, we obtain $y > x_{n+1} < x_{n+2} > z$, $y \parallel \{x_{n+2}, z\}$, and $z \parallel x_{n+1}$. Hence the subposet of t consisting of $\{y, x_{n+1}, x_{n+2}, z\}$ is isomorphic to $(N, \leq_N)$, a contradiction. Hence y and z

have to be comparable. Since $x_{n+1} < y$ and $x_{wn+1} \parallel z$, this implies $z < y$. Since x_{n+1} and x_{n+2} are unconnected in the spine of $\mathrm{Ha}(t)$ and comparable, there exists $u \in V$ with $x_{n+1} < u < x_{n+2}$. Now $y > u$ is impossible since $x_{n+1} \multimap\!< y$. Thus, in particular, $z \not> u$. Similarly, $z < u$ is impossible since $z \multimap\!< x_{n+2}$ which also implies $y \not< u$. Thus u is incomparable from y and z. As we already know that y and x_{n+2} are incomparable, we found a copy of $(N, \leq_N)$: $y > z < x_{n+2} > u$, a contradiction. Thus, we cannot find y and z with the properties described above.

In other words, either we find an element y that covers x_{n+1} and is incomparable with x_{n+2}, or there is z covered by x_{n+2} and incomparable with x_{n+1}. We only consider the first case since the second one is symmetric.

So any lower neighbor z of x_{n+2} dominates x_{n+1} and therefore $x_i \leq z$ for $1 \leq i \leq n+1$. So let $1 \leq i \leq n+1$. Since x_i and x_{n+2} are unconnected in $\mathrm{spine}(\mathrm{Ha}(t))$, there is $y_i \in V$ with $x_i \multimap\!< y_i$ such that y_i is incomparable with x_{n+2}. Hence in particular y_i and x_j for $i < j$ are incomparable. Now let $1 \leq i < j \leq n+1$ and assume $y_i \leq y_j$. Then the elements y_i, y_j, x_j and x_{n+2} form an N which is impossible. Hence the elements y_i are mutually incomparable for $1 \leq i < j \leq n+1$. But this contradicts the fact that any antichain in t has at most n elements.

Thus, indeed, the size of totally unconnected sets in $\mathrm{spine}(\mathrm{Ha}(t))$ is bounded for $t \in \mathrm{SP}(\Sigma)$ with $w(t) \leq n$. Now the statement follows from Lemma 5.2.3. □

Let $\mathcal{B}$ be a branching automaton. Then the elements of $L(\mathcal{B})$ can have autoconcurrency, i.e., $\mathrm{Ha}(L(\mathcal{B}))$ is not necessarily a set of Σ-dags. But if we consider only those elements of $\mathrm{Ha}(L(\mathcal{B}))$ that are Σ-dags, we obtain a set recognizable by a Σ-ACA:

Corollary 6.1.8 *Let $\mathcal{B}$ be a branching automaton. Then there exists a Σ-ACA $\mathcal{A}$ such that* $\mathrm{Ha}(L(\mathcal{B})) \cap \mathbb{D} = L(\mathcal{A})$.

Proof. By Theorem 6.1.5, the sets $L(\mathcal{B})$ and therefore $\mathrm{Ha}(L(\mathcal{B})) \cap \mathbb{D}$ are monadically axiomatizable. $\mathrm{Ha}(L(\mathcal{B})) \cap \mathbb{D} = L(\mathcal{A})$ is in addition width-bounded. Hence, by Lemma 6.1.7, $\mathrm{Ha}(L(\mathcal{B})) \cap \mathbb{D} \subseteq \mathbb{D}_k$ for some $k \in \mathbb{N}$. Now Theorem 5.2.10 completes the proof. □

6.2 P-asynchronous automata by Arnold

Generalizing the asynchronous automata for traces, Arnold defined P-asynchronous automata that are meant to accept Σ-labeled pomsets without autoconcurrency [Arn91]. In this section, we present in a condensed form some of his definitions and then show that the accepting power of P-asynchronous automata is captured by that of Σ-ACAs.

The tuple $\mathcal{B} = ((S_i)_{i\in I}, (\delta_{a,J})_{a\in\Sigma, J\subseteq I}, \iota, F, I, (D_a)_{a\in\Sigma})$ is a *P-asynchronous automaton over the alphabet* Σ provided

1. I is a finite set of *indices* with $\Sigma \subseteq I$,
2. S_i for $i \in I$ is a finite set of local states of process i,
3. $\delta_{a,J} : \prod_{j\in J} S_j \to \prod_{j\in J} S_j$ is a local transition function for $a \in \Sigma$ and $\emptyset \neq J \subseteq I$,
4. $D_a : S_a \to 2^I \setminus \{\emptyset\}$ is a mapping for $a \in \Sigma$,
5. $\iota \in \prod_{i\in I} S_i$ is the *initial state* and $F \subseteq \prod_{i\in I} S_i$ is the set of *accepting states.*

Above, I said that P-asynchronous automata are meant to accept pomsets. But the way they do this is more involved than for ACAs. First, from an P-asynchronous automaton, one defines a sequential automaton over Σ as follows: The set of states is the direct product of the local state spaces $S = \prod_{i\in I} S_i$. The transition function is defined in two steps: Let $a \in \Sigma$ and $s = (s_i)_{i\in I} \in S$. Then $J := D_a(s)$ is a subset of I. Let $(s'_j)_{j\in J} := \delta_{a,J}((s_j)_{j\in J})$ and, for $i \in I \setminus J$, $s'_i := s_i$. Then $\delta(s,a) := (s'_i)_{i\in I}$. In other words, the transition function $\delta : S \times \Sigma \to S$ changes only some components of its state space. The function D_a decides, which components are changed according to which local transition function $\delta_{a,J}$. Then the tuple (S, δ, ι, F) is a classical sequential automaton over the alphabet Σ. Now let $w = a_1 a_2 \ldots a_n \in \Sigma^\star$ be some word over Σ. Since the sequential automaton derived from $\mathcal{B}$ is total and deterministic, there is an initial computation path of the form

$$\iota = (s_i^0)_{i\in I} \xrightarrow{a_1} (s_i^1)_{i\in I} \xrightarrow{a_2} (s_i^2)_{i\in I} \xrightarrow{a_3} \cdots \xrightarrow{a_n} (s_i^n)_{i\in I}.$$

For $1 \leq \ell \leq n$, let $J_\ell := D_{a_\ell}(s_{a_\ell}^{\ell-1}) \neq \emptyset$, i.e. J_ℓ is the set of components of I that are changed in the ℓth computation step. Then $\sigma(w) := (a_1, J_1)(a_2, J_2)\ldots(a_n, J_n)$ is a word over $\Gamma := \Sigma \times (2^I \setminus \{\emptyset\})$. On the alphabet Γ, we again consider the dependence relation

$$D = \{((a, M), (b, N)) \in \Gamma^2 \mid M \cap N \neq \emptyset \text{ or } a = b\}.$$

Let $[\sigma(w)] = (V, \leq, \lambda_\Gamma)$ denote the trace from $\mathbb{M}(\Gamma, D)$ associated to the word $\sigma(w)$. Finally, let $\pi_1 : \Gamma \to \Sigma$ be the projection to the first component of a letter from Γ. Then $[w]_{\mathcal{B}} := (V, \leq, \pi_1 \circ \lambda_\Gamma)$ is a Σ-labeled partially ordered set. Note that it is completely determined by the word $w \in \Sigma^\star$, i.e. the P-asynchronous automaton $\mathcal{B}$ defines a mapping from $\Sigma^\star$ into the set of Σ-labeled partial orders. The image of this mapping is denoted by $P(\mathcal{B}) = \{[w]_{\mathcal{B}} \mid w \in \Sigma^\star\}$. A set of Σ-labeled pomsets P is *a-regular* if there exists a P-asynchronous automaton $\mathcal{B}$ with $P(\mathcal{B}) = P$.

Let $w \in \Sigma^\star$ and let $(V, \leq, \lambda) = [w]_{\mathcal{B}}$ denote the associated partial order. Since in (Γ, D) pairs with the same letter from Σ are dependent, this partial order has no autoconcurrency. Furthermore, the Σ-dag $(V, \multimap, \lambda)$ admits a $|D|$-chain covering since the Hasse-diagram of the trace $[\sigma(w)]$ is a $(\Gamma, |D|)$-dag by Example 5.2.1.

Now let again $\mathcal{B}$ be a P-asynchronous automaton and let (S, δ, ι, F) denote the sequential automaton derived from $\mathcal{B}$. We write $W(\mathcal{B})$ for the set of *words* that are accepted by (S, δ, ι, F) and call this set the *word language accepted by* $\mathcal{B}$. The *pomset-language accepted by* $\mathcal{B}$ is defined by

$$L(\mathcal{B}) := \{[w]_{\mathcal{B}} \mid w \in W(\mathcal{B})\}.$$

Note that any set of Σ-labeled pomsets that can be accepted by a P-asynchronous automaton is contained in some a-regular set since $L(\mathcal{B}) \subseteq P(\mathcal{B})$. In particular, for any P-asynchronous automaton $\mathcal{B}$, the set $\mathrm{Ha}(L(\mathcal{B}))$ of Hasse-diagrams of accepted pomsets consists of (Σ, k)-dags for some k. On the other hand, for $k > 1$, there is no P-asynchronous automaton $\mathcal{B}$ with $\mathrm{Ha}(L(\mathcal{B})) = \mathbb{D}_k$. In particular, P-asynchronous automata cannot accept $\mathbb{D}_k$ relative to $\mathbb{D}_{k+1}$. Since this is possible by a Σ-ACA (cf. Corollary 5.2.7), the expressive power of Σ-ACAs is not captured by that of P-asynchronous automata. But, on the contrary, any P-asynchronous automaton can be simulated by a Σ-ACA:

Theorem 6.2.1 *Let $\mathcal{B}$ be a P-asynchronous automaton over Σ. Then there exists a Σ-ACA $\mathcal{A}$ with* $\mathrm{Ha}(L(\mathcal{B})) = L(\mathcal{A})$.

Proof. The word language $\sigma(W(\mathcal{B})) = \{\sigma(w) \mid w \in W(\mathcal{B})\}$ is recognizable in $\Gamma^\star$. By [Arn91, Lemma 5.1], $\sigma(W(\mathcal{B}))$ is closed with respect to the trace equivalence, i.e. if $w' \in \sigma(W(\mathcal{B}))$ and $v' \in \Gamma^\star$ with $[w'] = [v']$, then $v' \in \sigma(W(\mathcal{B}))$. Hence the trace language $\{[\sigma(w)] \mid w \in W(\mathcal{B})\} \subseteq \mathbb{M}(\Gamma, D)$ is recognizable. By [EM96], it can be monadically axiomatized, say, by the sentence φ. In φ, replace any subformula of the form $\lambda_\Gamma(x) = (a, M)$ by $\lambda(x) = a \wedge \{i \in I \mid x \in C_i\} = M$ and denote the resulting formula by φ'. Now consider the following sentence ψ:

$$\begin{aligned} \exists_{i\in I} C_i : \quad (\quad & C_i \text{ is a chain for } i \in I \\ & \wedge \forall x, y : (x \multimap y \to (\lambda(x) = \lambda(y) \vee x, y \in C_i \text{ for some } i \in I)) \\ & \wedge \forall x, y : (x \parallel y \to \lambda(x) \neq \lambda(y)) \\ & \wedge \varphi' \\ ) \quad & \end{aligned}$$

Then ψ axiomatizes $L(\mathcal{B})$. Since the order relation $\leq$ can be monadically defined from the covering relation, we get that $\mathrm{Ha}(L(\mathcal{B}))$ can be monadically axiomatized. This is a set of Σ-dags and, even more, it is contained in $\mathbb{D}_k$ with $k = |D|$. Hence, by Theorem 5.2.10, it can be accepted by some Σ-ACA $\mathcal{A}$. Thus, we showed

$\mathrm{Ha}(L(\mathcal{B})) = L(\mathcal{A})$ for some Σ-ACA $\mathcal{A}$. $\square$

Thus, the advantage of P-asynchronous automata is that they are deterministic and therefore can easily be complemented. But this complementation always refers to the set $P(\mathcal{B})$, i.e. the complemented P-asynchronous automaton accepts $P(\mathcal{B}) \setminus L(\mathcal{B})$. On the other hand, the expressive power of P-asynchronous automata is strictly weaker than that of asynchronous cellular automata.