Chemical Pure Reaction Automata in Maximally Parallel Manner

Rocco Ascone$^{1[0009-0006-1390-4967]}$, Giulia Bernardini$^{1[0000-0001-6647-088X]}$, Francesco Leiter^{1}, and Luca Manzoni$^{1[0000-0001-6312-7728]}$

University of Trieste, Italy
rocco.ascone@phd.units.it,{giulia.bernardini,lmanzoni}@units.it,
francesco.leiter@studenti.units.it

Abstract. This work introduces a new class of reaction automata, named *Chemical Pure Reaction Automata* (CPRA), which combines the properties of chemical reaction automata, introduced and studied by [Okubo et al., 2016] and the recently defined pure reaction automata [Ascone et al., 2024]. In contrast with the standard model of chemical reaction automata, CPRA do not have permanence, ie the result states only consist of the products of the reactions that take place, while the reactants that are not consumed are lost.
We investigate the computational power of two variants of CPRA, both working in the maximally parallel manner. We first prove that *deterministic* CPRA (DCPRA), in which at every state there is a unique result state common to all multisets of enabled reactions for each input symbol, are not Turing complete. We then show that non-deterministic CPRA are Turing complete and thus strictly more powerful than DCPRA: namely, the set of languages accepted by CPRA in the maximally parallel manner contains the set of languages accepted by standard chemical reaction automata in the same manner.

Keywords: Reaction system · Reaction automata · Formal language · Computability

1 Introduction

Reaction automata, recently introduced by Okubo et al. [27,28,30], are computing devices that accept languages via manipulation of multisets of entities. Specifically, the states of a reaction automaton are defined as multisets drawn from a given finite set S of *entities*, and the transitions between the states are defined by a finite set of *reactions*. Each reaction consists of a multiset of reactants, a set of inhibitors, and a product multiset, all drawn from the set S. A reaction can happen in a state only if the multiset of reactants is contained in the state and none of the inhibitors has nonzero multiplicity: since several reactions can compete for the resources of a state, their relative dominance must be established through predetermined criteria known as *manners*. Reaction automata are further specified by a fixed initial state, receive input words over a fixed alphabet, and can accept such words based on a given set of final states.

Reaction automata can be seen as an extension of the simpler model of *reaction systems* [13,14,9], in which the states and the reactants, inhibitors and products are *sets* of entities rather than multisets. This initial assumption was motivated by the hypothesis that, after a certain threshold, the amount of chemical substances in a reaction is not important anymore, since there are enough of them for the reaction to take place. A crucial implication of this assumption is that any reaction system has only a finite set of possible states, making computational universality unattainable for such a model.

The properties of both reaction systems [16,15,17,12,7,20,37,5] and reaction automata [27,25,29,31,38,26] have been extensively studied, as well as several variations and restrictions of the models [10,34,6,8,24,4]. In this work, we introduce a new class of reaction automata as a variant of *Chemical Reaction Automata* (CRA), a model introduced and studied by Okubo et al. [30,31,26], and of *Pure Reaction Automata* (PRA), recently introduced by Ascone et al. [3]. Automata in this new class, which, to maintain coherence with the existing terminology, we named *Chemical Pure Reaction Automata* (CPRA), combine the characterising properties of CRA, in which no inhibitors are present in the reactions, and PRA, in which the reactants that are not consumed by the reactions happening at a state are not conserved in the result state, as it happens with reaction systems, in contrast with the standard CRA.

The addition of multiplicities in reaction automata and the most recent CRA and CPRA increases the similarity between these models and P systems [32,33]. In particular, like P systems, a multiset of objects or entities is evolved following rules inspired by the biochemical reactions that happen inside cells. One important difference is that in the large majority of P systems, the space is subdivided into regions by membranes (or cells), they actively participate in the computation and are essential to expand the computational power of the models [36]. As an example, consider the membrane nesting depth for P systems with active membranes with charges and its influence in the complexity classes that different depths characterise [35,21,22]. While investigating different derivation modes for the selection of reactions (or rules) to be applied is a new research direction for models related to reaction systems, it is a well-developed field of study for P systems. In particular, asynchronous and sequential derivations already exist for P systems in addition to the most common maximal parallelism criterion, which itself can be declined in different ways. Of particular interest is the case of maximal parallelism when the P system is used as a generator for numbers of Parikh sets [2], which has some similarity with the derivation modes for CPRA studied in this paper. For more details, we refer the reader to the ample literature on the topic of derivation modes for P systems [18,19,2,1].

Our results. We distinguish two sub-classes of CPRA and determine their computational power as language acceptors. In Section 3, we consider the restricted model of *deterministic* CPRA, in which at every state there is a unique result state common to all multisets of enabled reactions for each input symbol, and we prove that they are not Turing complete (Theorem 4). To arrive at this result, we make an unusual and somewhat surprising use of Dickson's Lemma, a standard result in commutative algebra, which we recall in Section 2.1.

In Section 4, we direct our attention to non-deterministic CPRA and prove that they are strictly more powerful than deterministic CPRA. In particular, the set of languages accepted by CPRA in the maximally parallel manner contains the set of languages accepted by standard chemical reaction automata in the same manner (Theorem 6).

2 Preliminaries

Let S be a finite alphabet and S^* the set of words over S, that is, all finite sequences of elements of S. A *multiset* over S can be defined as a function $V : S \to \mathbb{N}$ such that $V(a) \in \mathbb{N}$ is the multiplicity of $a \in S$ in the multiset; $S^{\#}$ denotes the set of all multisets over S. Given V and W two multisets over S, we can define a partial order given by multiset inclusion and the following multiset operators:

- Inclusion: $V \leq W$ if $V(a) \leq W(a)\ \forall\ a \in S$;
- Sum: $(V + W)(a) := V(a) + W(a)\ \forall\ a \in S$;
- Intersection: $(V \cap W)(a) := \min\{V(a), W(a)\}\ \forall\ a \in S$;
- Difference: $(V - W)(a) := V(a) - W(a)\ \forall\ a \in S$ (only defined for $W \leq V$);
- Symmetric difference: $(V \triangle W)(a) := (V + W)(a) - (V \cap W)(a)\ \forall\ a \in S$.

We also define a belonging relation for multisets: given $V \in S^{\#}$ and $a \in S$, $a \in V$ if and only if $V(a) \geq 1$, i.e., a letter belongs to V if and only if it has positive multiplicity. We can thus define the *set* underlying a multiset $V \in S^{\#}$ as the set of letters with positive multiplicity:

$$\mathrm{set}(V) := \{a \in S \mid a \in V\}.$$

The following equivalences follow immediately for any $V, W \in S^{\#}$:

$$\mathrm{set}(V + W) = \mathrm{set}(V) \cup \mathrm{set}(W), \quad \mathrm{set}(V \cap W) = \mathrm{set}(V) \cap \mathrm{set}(W).$$

Furthermore, $V \leq W$, implies $\mathrm{set}(V) \subseteq \mathrm{set}(W)$, but the converse is not true: eg given $V = \{a, a, b\}$ and $W = \{a, b, c\}$, it holds $\mathrm{set}(V) \subseteq \mathrm{set}(W)$ but $V \not\leq W$.

A set $U \subseteq S$ naturally corresponds to a multiset V_U such that $V_U(a) = 1$ if $a \in U$ and $V_U(a) = 0$ otherwise. In particular, for each $a \in S$, we will often denote the multiset $V_{\{a\}}$ simply by a. We will denote the empty multiset by $0 \in S^{\#}$. The total number of elements in a multiset $V \in S$ is defined as $\|V\| := \sum_{a \in S} V(a)$.

2.1 Topology on $S^{\#}$

In this section, we define a topology on $S^{\#}$ and we prove that, under this topology, every subset of $S^{\#}$ is compact. These results will be useful in Section 3 to determine the computational power of deterministic chemical pure reaction automata.

Definition 1. *Given $V \in S^{\#}$, we define $\mathcal{U}_V := \{W \in S^{\#} \mid V \leq W\}$, and $\mathfrak{T}$ the topology over $S^{\#}$ generated by the family $\mathfrak{B} := \{\mathcal{U}_V \mid V \in S^{\#}\}$.*

Remark 1. The family $\mathfrak{B}$ is a base for the topology $\mathfrak{T}$. Indeed, $\mathfrak{B}$ covers $S^{\#}$ since $S^{\#} = \mathcal{U}_0$; and given any $\mathcal{U}_V, \mathcal{U}_W \in \mathfrak{B}$, the intersection $\mathcal{U}_V \cap \mathcal{U}_W$ is equal to $\mathcal{U}_T$ where $T(a) = \max\{V(a), W(a)\}$ for all $a \in S$. Moreover, note that $V \leq W$ implies $\mathcal{U}_V \supseteq \mathcal{U}_W$.

Given $S = \{x_1, \ldots, x_n\}$ a finite alphabet of n letters, let $K[S] = K[x_1, \ldots, x_n]$ be the ring of polynomials in n variables $x_1, \ldots, x_n$ over the field K and let $\Pi(S) = \Pi(x_1, \ldots, x_n)$ be the set of all monomials in $K[S]$.

Remark 2. The following maps:

$$\begin{array}{rlcrl} S^{\#} & \longrightarrow \mathbb{N}^{|S|} & \qquad & S^{\#} & \longrightarrow \Pi(S) \\ V & \longmapsto (V(x_1), \ldots, V(x_n)) & \qquad & V & \longmapsto x_1^{V(x_1)} \cdot x_2^{V(x_2)} \cdot \ldots \cdot x_n^{V(x_n)} =: x^V \end{array}$$

are isomorphisms of monoids. Furthermore, we have that $V \leq W$ if and only if x^V divides x^W.

Definition 2 (monomial ideal). *An ideal $I \subseteq K[x_1, \ldots, x_n]$ is* monomial *if there exists $A \subseteq \mathbb{N}^n$ such that I is generated by the family of monomials $\{x^\alpha \mid \alpha \in A\}$, and it is denoted by $I = \langle x^\alpha : \alpha \in A \rangle$.*

We next report, for completeness, two results by Cox et al. we rely upon. Lemma 1 can be found at p70 of [11] (therein it is called Lemma 2); Theorem 1 is Theorem 5 at p74 of [11].

Lemma 1 ([11]). *Let $I = \langle x^\alpha : \alpha \in A \rangle$ be a monomial ideal, then a monomial x^β lies in I if and only if x^β is divisible by x^α for some $\alpha \in A$.*

Theorem 1 (Dickson's lemma [11]). *Let $I = \langle x^\alpha : \alpha \in A \rangle \subseteq K[x_1, \ldots, x_n]$ be a monomial ideal, then there exists $\alpha_1, \ldots, \alpha_k \in A$ such that $I = \langle x^{\alpha_1}, \ldots, x^{\alpha_k} \rangle$.*

As a corollary of Dickson's Lemma, we prove the following result.

Theorem 2. *Let $X \subseteq S^{\#}$, then for every open covering of X given by elements of the basis $\mathfrak{B}$, there exists a finite subcovering of X.*

Proof. Let $\mathfrak{U} = \{\mathcal{U}_V \mid V \in Y \subseteq S^{\#}\} \subseteq \mathfrak{B}$ be a covering of X. By Dickson's Lemma, the monomial ideal $I = \langle x^V : V \in Y \rangle$ equals $\langle x^{V_1}, \ldots, x^{V_k} \rangle$ for some $V_1, \ldots, V_k \in Y$. We claim that $\{\mathcal{U}_{V_1}, \ldots, \mathcal{U}_{V_k}\}$ is a finite subcovering of X. Indeed, given $W \in X$ there exists $V \in Y$ such that $W \in \mathcal{U}_V$, thus $V \leq W$. As pointed out in Remark 2, $V \leq W$ if and only if x^V divides x^W. By Lemma 1, we have $x^W \in I = \langle x^{V_1}, \ldots, x^{V_k} \rangle$, hence (again applying Lemma 1) we get that x^W is divisible by x^{V_i} for some $i = 1, \ldots, k$. Therefore $W \geq V_i$, ie $W \in \mathcal{U}_{V_i}$. □

Corollary 1. *Every subset of $S^{\#}$ is compact for the topology $\mathfrak{T}$.*

Corollary 2. *Given a sequence of multisets $\{V_n\}_{n\in\mathbb{N}} \subseteq S^{\#}$, there exist $N, M \in \mathbb{N}, N \geq M$, such that $V_N \geq V_M$.*

Proof. The family of open sets $\{\mathcal{U}_{V_n}\}_{n\in\mathbb{N}}$ is a covering for $\{V_n\}_{n\in\mathbb{N}}$. Since by Corollary 1 every subset is compact, there exists a finite subcovering $\{\mathcal{U}_{V_{N_1}}, . . . , \mathcal{U}_{V_{N_k}}\}$ of $\{V_n\}_{n\in\mathbb{N}}$. Then, given any $N > \max_{i=1,...,k} N_i$, there exists $i \in \{1, . . . , k\}$ such that $V_N \in \mathcal{U}_{V_{N_i}}$, ie $V_N \geq V_{N_i}$. □

2.2 Chemical Reaction Automata

Definition 3 (Chemical reaction). *Given an alphabet of reactants S, a* chemical reaction *over S is a pair $\mathbf{a} = (R_{\mathbf{a}}, P_{\mathbf{a}})$, where $R_{\mathbf{a}} \in S^{\#}$ is the multiset of* reactants *and $P_{\mathbf{a}} \in S^{\#}$ is the multiset of* products. *The set of all chemical reactions over S is denoted by* $\mathrm{chr}(S)$.

Let $\mathbf{a} = (R_{\mathbf{a}}, P_{\mathbf{a}})$, $\mathbf{b} = (R_{\mathbf{b}}, P_{\mathbf{b}}) \in \mathrm{chr}(S)$. A partial order over all possible chemical reactions over S can be naturally defined as $\mathbf{a} \leq_{\mathrm{r}} \mathbf{b}$ if and only if $R_{\mathbf{a}} \leq R_{\mathbf{b}}$. We also define the sum of the two chemical reactions as $\mathbf{a} + \mathbf{b} \coloneqq (R_{\mathbf{a}} + R_{\mathbf{b}}, P_{\mathbf{a}} + P_{\mathbf{b}})$. Given a finite set $\mathbf{A} \subseteq \mathrm{chr}(S)$, we denote by $\langle \mathbf{A} \rangle$ the Abelian semigroup generated by the elements of $\mathbf{A}$:

$$\langle \mathbf{A} \rangle \coloneqq \{\lambda_1 \mathbf{a}_1 + \cdots + \lambda_n \mathbf{a}_n \mid \mathbf{a}_i \in \mathbf{A}, \lambda_i \in \mathbb{N} \quad \forall i = 1, \ldots, n\}.$$

Definition 4. *Given $\mathbf{a} = (R_{\mathbf{a}}, P_{\mathbf{a}}) \in \mathrm{chr}(S)$ and $T \in S^{\#}$, $\mathbf{a}$ is* enabled *in T if $R_{\mathbf{a}} \leq T$. Furthermore, given $\mathbf{A}$ a finite set of chemical reactions over S and $\mathbf{a} \in \langle \mathbf{A} \rangle$ enabled in T, then $\mathbf{a}$ is enabled in a* maximally parallel manner (mp) *if there exists no $\mathbf{c} \in \langle \mathbf{A} \rangle$ such that $\mathbf{a} + \mathbf{c}$ is enabled in T, ie $\mathbf{a}$ is maximal w.r.t. addition.* $\mathrm{En}^{mp}_{\mathbf{A}}(T)$ *denotes the set of reactions from $\langle \mathbf{A} \rangle$ enabled in a state T in mp manner.*

Definition 5 ([30]). *The* result *of a set of chemical reactions $\mathbf{A}$ on a state T in mp manner is the set of states*

$$\mathrm{Res}^{mp}_{\mathbf{A}}(T) = \{P_a + (T - R_a) \mid \mathbf{a} = (R_a, P_a) \in \mathrm{En}^{mp}_{\mathbf{A}}(T)\}.$$

In the case where $\mathrm{En}^{mp}_{\mathbf{A}}(T) = \varnothing$, $\mathrm{Res}^{mp}_{\mathbf{A}}(T)$ *is undefined.*

Definition 6 ([30]). *A* chemical reaction automaton (CRA) *$\mathcal{A}$ is a tuple $\mathcal{A} = (S, \Sigma, \mathbf{A}, D_0, S_f)$, where S is a finite set of reactants, called the* background set *of $\mathcal{A}$; $\Sigma \subseteq S$ is the* input alphabet *of $\mathcal{A}$; $\mathbf{A} \subseteq \mathrm{chr}(S)$ is a finite set of chemical reactions over S; $D_0 \in S^{\#}$ is the initial multiset; and $S_f \subseteq S^{\#}$ is a set of final multisets.*

Definition 7. *Consider a chemical reaction automaton $\mathcal{A} = (S, \Sigma, \mathbf{A}, D_0, S_f)$ and a word $w = w_1 \cdots w_n \in \Sigma^*$. An* interactive process in $\mathcal{A}$ with input w in *mp* manner *is an infinite sequence $\pi = D_0, \ldots, D_i, \ldots$ where*

$$\begin{cases} D_{i+1} \in \mathrm{Res}^{mp}_{\mathbf{A}}(w_{i+1} + D_i) & \textit{for } 0 \leq i \leq n-1 \\ D_{i+1} \in \mathrm{Res}^{mp}_{\mathbf{A}}(D_i) & \textit{for } i \geq n. \end{cases}$$

$\mathrm{IP}_{mp}(\mathcal{A}, w)$ denotes the set of all such interactive processes in $\mathcal{A}$ with input w. We say that a process π accepts *w if there exists $m \geq n = |w|$ such that $D_m \in S_f$. By $\mathrm{AIP}_{mp}(\mathcal{A}, w)$ we denote the set of all processes $\pi \in \mathrm{IP}_{mp}(\mathcal{A}, w)$ such that π accepts w. The* language accepted *by $\mathcal{A}$ is defined as*

$$L_{mp}(\mathcal{A}) = \{w \in \Sigma^* \mid \mathrm{AIP}_{mp}(\mathcal{A}, w) \neq \varnothing\}.$$

The set of languages accepted by chemical reaction automata working in mp manner is denoted by $\mathcal{CRA}_{mp}$: a language $L \in \mathcal{CRA}_{mp}$ if and only if there exists a chemical reaction automaton working in mp manner that accepts L.

The following definition extends Definition 7 to the case where the symbols of the input word may be interleaved with an arbitrary number of gaps, modelled as an extra symbol $\lambda \notin \Sigma$.

Definition 8. *Let $\Sigma_\lambda = \Sigma \cup \{\lambda\}$ and $w = w_1 \cdots w_n$ an input word over Σ. An interactive process π is said to be in* λ-input mode *when it allows as input a sequence of characters from Σ_λ $b_1, \ldots, b_m$ such that $m \geq n$ and the symbols of w form a subsequence of $b_1, \ldots, b_m$ interleaved with occurrences of λ: more formally, $\exists\ 1 \leq i_1 < i_2 < \ldots < i_n \leq m$ such that $b_{i_j} = w_j\ \forall j = 1, \ldots, n$ and $b_i = \lambda\ \forall i \in [1, m] \setminus \{i_1, i_2, \ldots, i_n\}$. The notation $\mathrm{IP}^{\lambda}_X(\mathcal{A}, w)$, $\mathrm{AIP}^{\lambda}_X(\mathcal{A}, w)$, $L^{\lambda}_X(\mathcal{A})$ and $\mathcal{CRA}^{\lambda}_X$ naturally extends the corresponding notation of Definition 7 to λ-input mode.*

An ordinary interactive process, where no λ-input is used, will be said to be *real-time* to distinguish from λ-input mode processes.

Okubo et al. [30] proved that the computational power of chemical reaction automata working in *mp* manner in λ-input mode is equivalent to that of Turing machines: we report this result in Theorem 3 for completeness ($\mathcal{RE}$ denotes recursively enumerable languages).

Theorem 3. *[30, Theorem 1] $\mathcal{CRA}^{\lambda}_{mp} = \mathcal{RE}$.*

3 Deterministic Chemical Pure Reaction Automata

We begin by introducing a different kind of chemical reaction automata in which the result of a set of reactions only consists of the union of their products, while the reactants that are not consumed by the reactions are lost. This is in contrast with Definition 5 and the usual notion of reaction automaton. Definition 9 formalizes this concept.

Definition 9 (Pure result). *The* pure result *of a finite set of chemical reactions $\mathbf{A}$ on a state T in mp manner is*

$$\widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(T) = \{P_a \mid \mathbf{a} = (R_a, P_a) \in \mathrm{En}^{mp}_{\mathbf{A}}(T)\},$$

and if $\mathrm{En}^{mp}_{\mathbf{A}}(T) = \varnothing$, $\widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(T)$ is undefined.

Example 1 (Pure result). Let $\mathbf{A} = \{\mathbf{r}_1 = (a,b), \mathbf{r}_2 = (b+a,a)\} \in \mathrm{chr}(\{a,b\})$ and $T = 2b + a$, then $\mathrm{En}^{mp}_{\mathbf{A}}(T) = \{\mathbf{r}_1, \mathbf{r}_2\}$, thus $\widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(T) = \{b, a\}$. In contrast, $\mathrm{Res}^{mp}_{\mathbf{A}}(T) = \{3b, b+a\}$.

We name this new kind of reaction automata *Chemical Pure Reaction Automata* (CPRA). We define interactive processes in chemical pure reaction automata in much the same way as standard chemical reaction automata, as specified by Definition 10.

Definition 10. *Let* $\mathcal{M} = (S, \Sigma, \mathbf{A}, D_0, S_f)$ *be a* CPRA, $w = w_1 \cdots w_n \in \Sigma^*$. *An interactive process in* $\mathcal{M}$ *with input* w *in mp manner is an infinite sequence* $\pi = D_0, \ldots, D_i, \ldots$ *where*

$$\begin{cases} D_{i+1} \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(w_{i+1} + D_i) & \textit{for } 0 \le i \le n-1 \\ D_{i+1} \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(D_i) & \textit{for } i \ge n. \end{cases}$$

Exactly as for chemical reaction automata, we say that π *accepts* w *if there exists* $m \ge n = |w|$ *such that* $D_m \in S_f$. *We also define* $\mathrm{IP}_{mp}(\mathcal{M}, w)$, $\mathrm{AIP}_{mp}(\mathcal{M}, w)$, *and* $L_{mp}(\mathcal{M})$ *in the same way as for chemical reaction automata. The set of languages accepted by* CPRA *working in mp manner is denoted by* $\mathcal{CPRA}_{mp}$.

The definition of λ-input mode process (Definition 8) holds also in the pure case. In particular, given a chemical pure reaction automata $\mathcal{M}$ and w input word, we will use the following notations $\mathrm{IP}^{\lambda}_{mp}(\mathcal{M}, w)$, $\mathrm{AIP}^{\lambda}_{mp}(\mathcal{M}, w)$, $L^{\lambda}_{mp}(\mathcal{M})$. The set of languages accepted by CPRA working in *mp* manner with λ-input mode is denoted by $\mathcal{CPRA}^{\lambda}_{mp}$. We will sometimes represent an interactive process π with the following "arrow notation", which extends the notation proposed by [38]:

$$\pi : D_0 \xrightarrow[w_1]{\mathbf{a}_1} D_1 \xrightarrow[w_2]{\mathbf{a}_2} D_2 \xrightarrow[w_3]{\mathbf{a}_3} \cdots D_{n-1} \xrightarrow[w_n]{\mathbf{a}_n} D_n \xrightarrow{\mathbf{a}_{n+1}} D_{n+1} \xrightarrow{\mathbf{a}_{n+1}} \cdots$$

where $D_{i-1} \xrightarrow[w_i]{\mathbf{a}_i} D_i$ means w_i is the input letter at state D_{i-1}, $\mathbf{a}_i \in \langle \mathbf{A} \rangle$ is the reaction enabled in $D_{i-1} + w_i$ which takes place, and $D_i \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(w_i + D_{i-1})$.

The restricted class of *deterministic* CPRA, formalized in Definition 11, requires that the pure result of any reachable state consists of one state only.

Definition 11. *Given* $\mathcal{M} = (S, \Sigma, \mathbf{A}, D_0, S_f)$ *a* CPRA *working in mp manner, we say that* $\mathcal{M}$ *is* deterministic (DCPRA) *if and only if for any reachable state* V, *the pure result* $\widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(V)$ *consists of one element only.*

The set of languages accepted by DCPRA working in *mp* manner is denoted by $\mathcal{DCPRA}_{mp}$, while $\mathcal{DCPRA}^{\lambda}_{mp}$ denotes the languages accepted with λ-input mode. We remark that the notion of determinism given by Definition 11 differs from the notion given by Okubo et al. [31] for CRA, as their definition is too restrictive for the objects of our study. Note that any CPRA with the reactions defined in Example 1 is not deterministic. In contrast, Example 2 showcases a simple deterministic CPRA according to Definition 11.

Example 2. Given a background set $S = \{a, b, s_0, s_1, f, \clubsuit\}$, an input alphabet $\Sigma = \{a, b\}$, and a set of chemical reactions $\mathbf{A} = \{\mathbf{r}_1 = (s_0 + a, s_1), \mathbf{r}_2 = (s_0 + b, \clubsuit), \mathbf{r}_3 = (s_1 + b, s_0), \mathbf{r}_4 = (s_1 + a, \clubsuit), \mathbf{r}_5 = (\clubsuit, \clubsuit), \mathbf{r}_6 = (f, f)\}$, let $\mathcal{A} = (S, \{a, b\}, \mathbf{A}, s_0 + f, \{f\})$ be a chemical reaction automaton working in mp manner. It is possible to show that $\mathcal{A}$ is deterministic and that the language accepted by $\mathcal{A}$ is $L_{mp}(\mathcal{A}) = \{(ab)^n \mid n \geq 0\}$. Indeed, if we consider the input word $w = abab \in \Sigma^*$, we get the accepting process:

$$s_0 + f \xrightarrow[a]{\mathbf{r}_1+\mathbf{r}_6} s_1 + f \xrightarrow[b]{\mathbf{r}_3+\mathbf{r}_6} s_0 + f \xrightarrow[a]{\mathbf{r}_1+\mathbf{r}_6} s_1 + f \xrightarrow[b]{\mathbf{r}_3+\mathbf{r}_6} s_0 + f \xrightarrow{\mathbf{a}_6} f.$$

Instead, if we choose $w = abba \in \Sigma^*$, we get the process:

$$s_0{+}f \xrightarrow[a]{\mathbf{r}_1+\mathbf{r}_6} s_1{+}f \xrightarrow[b]{\mathbf{r}_3+\mathbf{r}_6} s_0{+}f \xrightarrow[b]{\mathbf{r}_2+\mathbf{r}_6} \clubsuit{+}f \xrightarrow[a]{\mathbf{r}_5+\mathbf{r}_6} \clubsuit{+}f \xrightarrow{\mathbf{r}_5+\mathbf{r}_6} \clubsuit{+}f \xrightarrow{\mathbf{r}_5+\mathbf{r}_6} \cdots$$

which clearly does not accept $abba$ since $f + \clubsuit$ is not a final state.

In this section, we investigate the computational power of deterministic chemical pure reaction automata and prove that this class is not Turing complete. Observe that reaction systems can be seen as a simpler kind of pure reaction automata in which the multiplicities of reactants are not considered. We can thus think of inhibitorless reaction systems as a simpler version of the chemical pure reaction automata. A consequence of [23, Prop. 20] is that the result function for inhibitorless reaction systems is monotone. In Lemma 2, we extend this result for chemical pure reaction automata working in mp manner.

Lemma 2. *Let $\mathbf{A}$ be a set of chemical reactions over $S^{\#}$, $D, D' \in S^{\#}$ be two multisets such that $D \leq D'$, $\widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(D) = \{P\}$, $\widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(D') = \{P'\}$, then $P \leq P'$.*

Proof. Let $\mathbf{a} = (R_{\mathbf{a}}, P) \in \mathrm{En}^{mp}_{\mathbf{A}}(D)$. Since $D' \geq D$, $\mathbf{a}$ is also enabled by D'. Since the reactions happen according to the mp manner (see Definition 4) there exists $\mathbf{a}' = (R_{\mathbf{a}'}, P') \in \mathrm{En}^{mp}_{\mathbf{A}}(D')$ and $\mathbf{c} \in \langle \mathbf{A} \rangle \cup \{(0, 0)\}$ such that $\mathbf{a}' = \mathbf{a} + \mathbf{c}$. We conclude that $P' = P + P_{\mathbf{c}}$, thus $P' \geq P$.

Remark 3. Lemma 2 does not hold if we apply the standard notion of result (Def.5) instead of pure result (Def.9). Indeed, let $\mathcal{A} = \{(a, a), (3b, b)\}$ be two chemical reactions over $S = \{a, b\}$ and consider states $D' = a{+}3b$ and $D = a{+}2b$. Then $\mathrm{Res}^{mp}(D) = \{a{+}2b\} = \{D\}$ and $\mathrm{Res}^{mp}(D') = \{a{+}b\}$, thus $a{+}2b > a{+}b$ despite being $D < D'$.

Theorem 4. *Given any deterministic pure chemical reaction automaton $\mathcal{A}$, it is always possible to determine in a finite number of steps, for any finite input, whether $\mathcal{A}$ will halt.*

Proof. Let $\mathcal{A} = (S, \Sigma, \mathbf{A}, D_0, S_f)$ be a DCPRA working in mp manner and $\pi = D_0, D_1, \ldots, D_i, \ldots \in \mathrm{IP}_{mp}(\mathcal{A}, w)$ a process, where $w = w_1 \cdots w_n \in \Sigma^*$ is the input word. Since there is a finite number of final states, we describe an algorithm that works for a single, fixed final state F, and then notice that it can be applied separately for each of the final states of $\mathcal{A}$ to obtain the statement. Consider the map

$$\begin{aligned} R : S^{\#} &\longrightarrow S^{\#} \\ T &\longmapsto R(T) \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(T). \end{aligned}$$

R is well defined on the elements of π because $\mathcal{A}$ is deterministic; in particular, it holds that $R(D_k) = D_{k+1}$ for all $k \geq n = |w|$. By Lemma 2, if R is well defined for two multisets $D \leq D'$, then $R(D) \leq R(D')$.

Let us now describe the halting algorithm. Given the sequence $D_n, D_{n+1}, \dots$, by Corollary 2 we can always find two integers $N \geq M$ such that $D_N \geq D_M$. Thus $R(D_N) \geq R(D_M)$, which translates to $D_{N+1} \geq D_{M+1}$. By induction, $D_{N+k} \geq D_{M+k}$ until k reaches $N' = N - M$, where we get $D_{N+N'} \geq D_{M+N'} = D_N \geq D_M$. This implies that any state D_k reached by the process later than D_N must belong to $\mathcal{U}_{D_{M+d}}$ for some $d \in [0, N']$: more precisely,

$$D_k \geq D_{M+((k-N) \mod N')} \quad \forall k \geq N, \quad \text{ie} \quad D_k \in \mathcal{U}_{D_{M+((k-N) \mod N')}}. \tag{1}$$

We call $D_k \leq D_{k+N'} \leq \dots$ the *chain* obtained from D_k as per Equation 1. The existence of such chains guarantees that the algorithm only needs to inspect a finite number of states to decide whether the DCPRA halts. Indeed, there are the following two cases. (1) If $F \notin \mathcal{U}_{D_k}$ (ie $F \ngeq D_k$) for any $k \in \{M, \dots, N-1\}$, then $\pi \notin \mathrm{AIP}_{mp}(\mathcal{A}, w)$ and we can immediately halt the algorithm. Otherwise, (2) there exists $k \in \{M, \dots, N-1\}$ such that $F \in \mathcal{U}_{D_k}$ (ie $F \geq D_k$): let $I = \{M_1, \dots, M_j\} \subseteq \{M, \dots, N-1\}$ be the subset of such indices, ie $F \geq D_k$ for all $k \in I$. Consider now $I_1 = \{M_1 + N', \dots, M_j + N'\}$ the set of indices of the successors in the chains starting from D_k, for all $k \in I$. If $F = D_h$ for some $h \in I_1 \cup I$, then $\pi \in \mathrm{AIP}_{mp}(\mathcal{A}, w)$, thus we can halt the algorithm. Therefore, assume $F > D_h$ for all $h \in I_1$. There are two subcases.

(i), for some index $M_k + N' \in I_1$, we find $D_{M_k+N'} = D_{M_k}$: this indicates that the process entered in a loop, thus $\pi \notin \mathrm{AIP}_{mp}(\mathcal{A}, w)$ and we can halt the algorithm. (ii), $D_{M_k+N'} > D_{M_k}$ for every $M_k \in I_1$. In this case, we iterate the procedure described for case (2) on I_2 (the successors of the chains corresponding to the indices of I_1), that is, we check if F is equal to any of those elements or it is found in any of the relative $\mathcal{U}$'s, we verify we have not entered a closed loop and then we proceed considering I_3. Remark that at every iteration we get closer to F, since we have strictly greater multisets at each step, and thus the maximum number of possible iterations is less than $\|F - D_{M_k}\|_1$ for all $M_k \in I_1$.

Since the DCPRA has only a finite number of final states, by applying this procedure separately for every possible final state we obtain the statement. □

Observe that Theorem 4 implies that the halting problem associated with the class of DCPRA working in *mp* is decidable. We deduce that this class of automata cannot be Turing complete, or equivalently, the set of languages accepted by DCPRA cannot coincide with the class of all recursively enumerable languages. We have arrived at the main result of this section.

Theorem 5. *$\mathcal{DCPRA}_{mp} \subsetneq \mathcal{RE}$ and $\mathcal{DCPRA}^{\lambda}_{mp} \subsetneq \mathcal{RE}$, ie the class of DCPRA working in mp manner is not Turing complete.*

4 Non-Deterministic Chemical PRA

In this section, we study the computational power of non-deterministic chemical pure reaction automata working in *mp* manner. Our main result, given in Theorem 6, is that the computational power of *pure* chemical reaction automata working in *mp* manner is the same as standard CRA working in the same manner, implying that CPRA are strictly more powerful than the *deterministic* CPRA studied in Section 3. We begin with an example of non-deterministic computations in CPRA.

Example 3. Given a background set $S = \{a, b, s_0, s_1, a'\}$, an input alphabet $\Sigma = \{a, b\}$, and a set of chemical reactions $\mathbf{A} = \{\mathbf{r}_1 = (s_0 + a, s_0 + a'), \mathbf{r}_2 = (s_0 + a' + b, s_1), \mathbf{r}_3 = (s_1 + a' + b, s_1), \mathbf{r}_4 = (a', a')\}$, let $\mathcal{A} = (S, \{a, b\}, \mathbf{A}, s_0, \{s_1\})$ be a chemical reaction automaton working in *mp*. It is possible to show that $\mathcal{A}$ accepts the context-free language $\{a^n b^n \mid n > 0\}$, both in real-time and in λ-input mode, ie $L^{\lambda}_{mp}(\mathcal{A}) = L_{mp}(\mathcal{A}) = \{a^n b^n \mid n > 0\}$. Indeed, consider eg the input word $w = aabb \in \Sigma^*$. We obtain the following accepting process:

$$s_0 \xrightarrow[a]{\mathbf{r}_1} s_0 + a' \xrightarrow[a]{\mathbf{r}_1} s_0 + 2a' \xrightarrow[b]{\mathbf{r}_2 + \mathbf{r}_4} s_1 + a' \xrightarrow[b]{\mathbf{r}_2} s_1.$$

Note that the non-determinism of the CPRA is relevant eg the third step, in which $T = s_0 + 2a'$ receives in input the symbol b: we have that $\mathrm{En}^{mp}_{\mathbf{A}}(T + b) = \{\mathbf{r}_2 + \mathbf{r}_4, 2\mathbf{r}_4\}$ thus $\widehat{\mathrm{Res}}^{mp}_{\mathbf{A}}(T + b) = \{2a', s_1 + a'\}$. Furthermore, it can be easily verified that any λ symbol added to *aabb* would force a non-accepting process: eg inserting a gap after the first a would make the process loop over the state $T' = a'$, as $\mathrm{En}^{mp}_{\mathbf{A}}(s_0 + a') = \{\mathbf{r}_4\} = \mathrm{En}^{mp}_{\mathbf{A}}(a')$.

Consider now $w = aab \notin \{a^n b^n \mid n > 0\}$. It is easy to verify that no process reaches the final state s_1, eg w corresponds to the following two processes with real-time input:

$$\begin{tikzcd}
s_0 \arrow[r, "\mathbf{r}_1", "a"'] & s_0 + a' \arrow[r, "\mathbf{r}_1 + \mathbf{r}_4", "a"'] & s_0 + 2a' \arrow[r, "\mathbf{r}_2 + \mathbf{r}_4", "b"'] \arrow[dr, "2\mathbf{r}_4", "b"'] & s_1 + a' \arrow[r, "\mathbf{r}_4"] & a' \arrow[r, "\mathbf{r}_4"] & a' \arrow[r, "\mathbf{r}_4"] & \cdots \\
 & & & 2a' \arrow[r, "2\mathbf{r}_4"] & 2a' \arrow[r, "2\mathbf{r}_4"] & \cdots &
\end{tikzcd}$$

Figure 1 summarizes all possible processes in $\mathcal{A}$ for every possible input.

Theorem 6. *Given any chemical reaction automaton $\mathcal{A} = (S, \Sigma, \mathbf{A}, D_0, S_f)$ working in mp manner, there exists a chemical pure reaction automaton $\mathcal{M}$ working in mp manner such that $L^{\lambda}_{mp}(\mathcal{A}) = L^{\lambda}_{mp}(\mathcal{M})$.*

Proof. Let $\mathcal{M} = (S \cup S' \cup \{\spadesuit\}, \Sigma, \mathbf{A}', D_0, S'_f)$ be a chemical pure reaction automaton operating in a maximally parallel manner such that:

- $S' = \{x' \mid x \in S\}$ is a set in a bijection with the elements of S (a "copy" of S). From now on, given a multiset X over S, X' will be naturally defined as the multiset consisting of the copies of the elements of X.

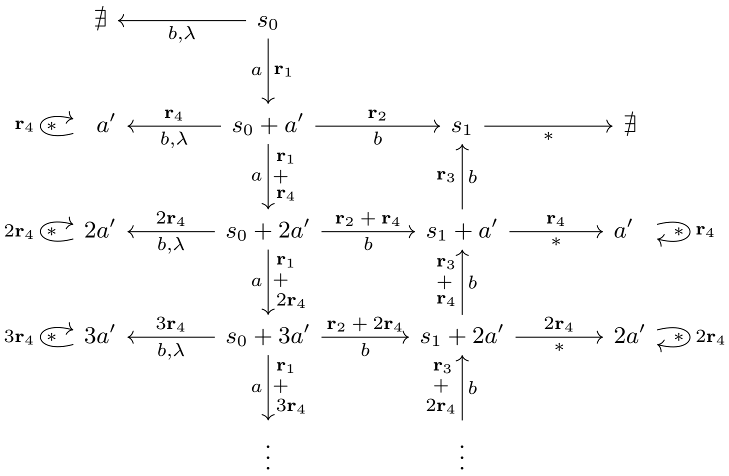

Fig. 1. Schematic representation of all possible processes of the chemical pure reaction automaton of Example 3, which accepts the language $\{a^n b^n \mid n > 0\}$. The notation $T \underset{*}{\xrightarrow{\mathbf{r}}} T'$ means that, regardless of whether the input symbol is a, b or λ, the reaction enabled by T is $\mathbf{r}$.

- Σ is the same input alphabet as $\mathcal{A}$;
- $\mathbf{A}' = \mathbf{A}_p \cup \mathbf{A}_c \cup \mathbf{A}_r \cup \{(\spadesuit, \spadesuit)\}$, where $\mathbf{A}_p = \{(x, x + x') \mid x \in S\}$, $\mathbf{A}_c = \{(R + R', P + P') \mid (R, P) \in \mathbf{A}\}$ and $\mathbf{A}_r = \{(R', \spadesuit) \mid (R, P) \in \mathbf{A}\}$.
- D_0 is the same initial state as $\mathcal{A}$;
- $S'_f = \{D + D' \mid D \in S_f\}$ is the new set of final states.

Remark that by reaction $(\spadesuit, \spadesuit)$, we know that $\spadesuit$ is preserved whenever generated, therefore, in order to reach a final state in S'_f, $\spadesuit$ should never be generated, ie reactions from $\langle \mathbf{A}_r \rangle$ should never be enabled. We formalize this idea in the following claim.

Claim. For any state W of $\mathcal{A}$, it holds $W \in \mathrm{Res}^{mp}_{\mathbf{A}}(V)$ if and only if $W + W' \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}'}(V + V')$.

Proof. Given a reaction $\mathbf{a} \in \langle \mathbf{A} \rangle$, we denote the corresponding reaction in $\langle \mathbf{A}_c \rangle$ by $\mathbf{a}'$ and the corresponding reaction in $\langle \mathbf{A}_r \rangle$ by $\mathbf{a}_{\spadesuit}$. We prove the two implications.

$\Rightarrow$) If $W \in \mathrm{Res}^{mp}_{\mathbf{A}}(V)$ then $\exists \mathbf{a} = (R_{\mathbf{a}}, P_{\mathbf{a}}) \in \mathrm{En}^{mp}_{\mathbf{A}}(V)$ such that $P_{\mathbf{a}} + V - R_{\mathbf{a}} = W$. Consider $\mathbf{a}'' \coloneqq \mathbf{a}' + (V - R_{\mathbf{a}}, V - R_{\mathbf{a}} + V' - R'_{\mathbf{a}}) = (V + R'_{\mathbf{a}}, W + W') \in \langle \mathbf{A}' \rangle$. Clearly, $V + V'$ enables $\mathbf{a}''$; we want to show that it is *mp* enabled. First, we remark that $\mathbf{a}'' + \mathbf{c}$ is not enabled by $V + V'$ for any $\mathbf{c} \in \langle \mathbf{A}_p \cup \mathbf{A}_c \cup \{(\spadesuit, \spadesuit)\} \rangle$. If $\mathbf{a}'' + \mathbf{b}_{\spadesuit}$ is enabled by $V + V'$, then $R'_{\mathbf{a}} + R'_{\mathbf{b}} \le V'$, thus $\mathbf{a} + \mathbf{b}$ is enabled by V, a contraddiction since $\mathbf{a}$ is *mp* enabled. We conclude that $\mathbf{a}'' \in \mathrm{En}^{mp}_{\mathbf{A}'}(V + V')$, hence $W + W' \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}'}(V + V')$.

$\Leftarrow$) If $W + W' \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}'}(V + V')$ then $\exists \mathbf{a}'' = \mathbf{a}' + (R, R + R') \in \mathrm{En}^{mp}_{\mathbf{A}'}(V + V')$, for some $\mathbf{a} \in \langle \mathbf{A} \rangle$ and $R \in S^{\#}$, such that $P_{\mathbf{a}''} = W + W'$. Since $\spadesuit \notin W + W'$, there can be no reaction from $\langle \mathbf{A}_r \rangle$ in the decomposition of $\mathbf{a}''$ as a sum of reactions from $\langle \mathbf{A}_p \rangle$, $\langle \mathbf{A}_c \rangle$, and $\langle \mathbf{A}_r \rangle$. We notice that $V - R_{\mathbf{a}} = R$, as otherwise for any $x \in V - R_{\mathbf{a}} - R$ the reaction $\mathbf{a}'' + (x, x + x')$ would be enabled by $V + V'$, in contradiction to the fact that $\mathbf{a}''$ is *mp* enabled. Remark that $\mathbf{a} \in \langle \mathbf{A} \rangle$ is enabled by V. We now prove it is also *mp* enabled. Suppose for a contradiction that there exists $\mathbf{b} \in \langle \mathbf{A} \rangle$ such that $\mathbf{a} + \mathbf{b}$ is enabled by V, then $\mathbf{a}'' + \mathbf{b}_{\spadesuit}$ would be enabled by $V + V'$, a contradiction since $\mathbf{a}''$ is *mp* enabled. Finally, we conclude that $\mathbf{a} \in \mathrm{En}^{mp}_{\mathbf{A}}(V)$, hence $P_{\mathbf{a}} + V - R_{\mathbf{a}} = W \in \mathrm{Res}^{mp}_{\mathbf{A}}(V)$.

Claim. For any state W of $\mathcal{A}$ and any input letter $a \in \Sigma$, it holds $W \in \mathrm{Res}^{mp}_{\mathbf{A}}(V)$ if and only if $W + W' + a + a' \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}'}(V + V' + a)$.

Proof. The statement follows from the previous claim and the fact that whenever $\mathbf{r} \in \mathrm{En}^{mp}_{\mathbf{A}'}(V + V')$, we have $\mathbf{r} + (a, a + a') \in \mathrm{En}^{mp}_{\mathbf{A}'}(V + V' + a)$.

Consider now an input word $w = w_1 \cdots w_n \in \Sigma^*_\lambda$. Let $\pi = D_0, D_1, \ldots, D_i, \ldots \in \mathrm{IP}^{\lambda}_{mp}(\mathcal{A}, w)$: the corresponding process in $\mathcal{M}$ $D_0, E_1 \ldots, E_n \in \mathrm{IP}^{\lambda}_{mp}(\mathcal{M}, w)$ is obtained as follows. If $w_1 = \lambda$, then $E_1 \coloneqq D_0 + D'_0 \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}'}(D_0)$ since the only reactions that are enabled are those in $\langle \mathbf{A}_p \rangle$. If $w_1 \neq \lambda$, then $E_1 \coloneqq D_0 + D'_0 + w_1 + w'_1 \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}'}(D_0 + w_1)$, by the second claim. Therefore in the next steps of the process we have:

$$\begin{cases} E_{k+1} = D_k + D'_k + w_{k+1} + w'_{k+1} \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}'}(E_k + w_{k+1}) & \text{if } w_{k+1} \neq \lambda \\ E_{k+1} = D_k + D'_k \in \widehat{\mathrm{Res}}^{mp}_{\mathbf{A}'}(E_k) & \text{if } w_{k+1} = \lambda \text{ or } k \geq n. \end{cases}$$

Hence the processes of $\mathcal{M}$ mimic those of $\mathcal{A}$ with one step of delay, ie E_1 contains D_0, E_2 contains D_1, and so forth. If $\pi \in \mathrm{AIP}^{\lambda}_{mp}(\mathcal{A}, w)$ then there exists $m \geq |w|$ such that $D_m \in S_f$; this holds true if and only if $E_{m+1} \in S'_f$. Therefore $\pi \in \mathrm{AIP}^{\lambda}_{mp}(\mathcal{A}, w)$ if and only if $\pi' \in \mathrm{AIP}^{\lambda}_{mp}(\mathcal{M}, w)$, and hence we obtain the thesis. □

The following corollary, showing the computational universality of $\mathcal{CPRA}^{\lambda}_{mp}$, follows directly from [30, Theorem 1] and Theorem 6.

Corollary 3. $\mathcal{CPRA}^{\lambda}_{mp} = \mathcal{RE}$.

5 Conclusions

We have investigated the computational power of both deterministic and non-deterministic chemical pure reaction automata working in *mp* manner as language acceptors, and proved that, while non-deterministic CPRA are Turing complete, DCPRA are not.

Several research questions remain open and are worthy of being investigated further. First of all, while we proved that DCPRA are not universal, their exact computation power is unknown. Hence, it would be interesting to see the relation between the class of languages recognised by DCPRA and the Chomsky hierarchy: does that class correspond to (or is neatly contained between) some level of the hierarchy or is it incomparable with it?

As a second question, there appears to be some relation between CPRA working in *mp* manner and Petri nets, similar to what happens with CPRA working in the sequential manner. Can this relation be formalised and, possibly, show that chemical (pure) reaction automata and their variations represent the "chemical version" of Petri net?

Recently, an additional working mode, the *maximally reactive* manner, was introduced in [3, Definition 3]. Thus, it would be interesting to investigate the computation power given by this new manner of execution for CPRA and DCPRA. In particular, exploring the effect of different ways in which the reactions are selected and the interplay with determinism can increase our understanding of *where* the computational power of chemical pure reaction automata comes from.

Finally, the proposed working modes have some analogies with the existing derivation modes for P systems. Thus, it would be an interesting research direction to see which modes can be "ported" to CPRA and DCPRA, and their influence on computational universality. In particular, a general framework to talk about working modes across multiple natural computing models is still absent, but it would be a worthwhile addition to the field.

References

1. Alhazov, A., Freund, R., Ivanov, S., Verlan, S.: Variants of derivation modes for which catalytic p systems with one catalyst are computationally complete. Journal of Membrane Computing **3**(4), 233–245 (2021)
2. Alhazov, A., Freund, R., Verlan, S.: P systems working in maximal variants of the set derivation mode. In: Membrane Computing: 17th International Conference, CMC 2016, Milan, Italy, July 25-29, 2016, Revised Selected Papers 17. pp. 83–102. Springer (2017)
3. Ascone, R., Bernardini, G., Formenti, E., Leiter, F., Manzoni, L.: Pure reaction automata. Natural Computing (May 2024). https://doi.org/10.1007/s11047-024-09980-7
4. Ascone, R., Bernardini, G., Manzoni, L.: Fixed points and attractors of additive reaction systems. Natural Computing (Mar 2024). https://doi.org/10.1007/s11047-024-09977-2
5. Ascone, R., Bernardini, G., Manzoni, L.: Fixed points and attractors of reactantless and inhibitorless reaction systems. Theoretical Computer Science **984**, 114322 (2024). https://doi.org/10.1016/j.tcs.2023.114322
6. Azimi, S.: Steady states of constrained reaction systems. Theoretical Computer Science **701**, 20–26 (2017)
7. Barbuti, R., Bernasconi, A., Gori, R., Milazzo, P.: Computing preimages and ancestors in reaction systems. In: International Conference on Theory and Practice of Natural Computing. pp. 23–35. Springer (2018)

8. Bottoni, P., Labella, A., Rozenberg, G.: Reaction systems with influence on environment. Journal of Membrane Computing **1**, 3–19 (2019)
9. Brijder, R., Ehrenfeucht, A., Main, M., Rozenberg, G.: A tour of reaction systems. International Journal of Foundations of Computer Science **22**(07), 1499–1517 (2011)
10. Brijder, R., Ehrenfeucht, A., Rozenberg, G.: Reaction systems with duration. Computation, cooperation, and life **6610**, 191–202 (2011). https://doi.org/10.1007/978-3-642-20000-7_16
11. Cox, D.A., Little, J., O'Shea, D.: Gröbner Bases, pp. 49–119. Springer International Publishing, Cham (2015). https://doi.org/10.1007/978-3-319-16721-3_2, https://doi.org/10.1007/978-3-319-16721-3{_}2
12. Ehrenfeucht, A., Kleijn, J., Koutny, M., Rozenberg, G.: Evolving reaction systems. Theoretical Computer Science **682**, 79–99 (2017). https://doi.org/10.1016/j.tcs.2016.12.031
13. Ehrenfeucht, A., Rozenberg, G.: Basic notions of reaction systems. In: Developments in Language Theory, 8th International Conference (DLT). Lecture Notes in Computer Science, vol. 3340, pp. 27–29. Springer (2004). https://doi.org/10.1007/978-3-540-30550-7_3
14. Ehrenfeucht, A., Rozenberg, G.: Reaction systems. Fundam. Informaticae **75**(1-4), 263–280 (2007)
15. Formenti, E., Manzoni, L., Porreca, A.E.: Cycles and global attractors of reaction systems. In: Descriptional Complexity of Formal Systems: 16th International Workshop (DCFS). pp. 114–125. Springer (2014). https://doi.org/10.1007/978-3-319-09704-6_11
16. Formenti, E., Manzoni, L., Porreca, A.E.: Fixed points and attractors of reaction systems. In: Language, Life, Limits: 10th Conference on Computability in Europe (CiE). pp. 194–203. Springer (2014). https://doi.org/10.1007/978-3-319-08019-2_20
17. Formenti, E., Manzoni, L., Porreca, A.E.: On the complexity of occurrence and convergence problems in reaction systems. Natural Computing **14**, 185–191 (2015)
18. Freund, R., Verlan, S.: A formal framework for static (tissue) p systems. In: Membrane Computing: 8th International Workshop, WMC 2007 Thessaloniki, Greece, June 25-28, 2007 Revised Selected and Invited Papers 8. pp. 271–284. Springer (2007)
19. Freund, R., Verlan, S.: (tissue) p systems working in the k-restricted minimally or maximally parallel transition mode. Natural Computing **10**, 821–833 (2011)
20. Holzer, M., Rauch, C.: On the computational complexity of reaction systems, revisited. In: Computer Science–Theory and Applications: 16th International Computer Science Symposium in Russia, CSR 2021, Sochi, Russia, June 28–July 2, 2021, Proceedings 16. pp. 170–185. Springer (2021)
21. Leporati, A., Manzoni, L., Mauri, G., Porreca, A.E., Zandron, C.: Simulating elementary active membranes: with an application to the p conjecture. In: International Conference on Membrane Computing. pp. 284–299. Springer (2014)
22. Leporati, A., Manzoni, L., Mauri, G., Porreca, A.E., Zandron, C.: Membrane division, oracles, and the counting hierarchy. Fundamenta Informaticae **138**(1-2), 97–111 (2015)
23. Manzoni, L., Pocas, D., Porreca, A.E.: Simple reaction systems and their classification. International Journal of Foundations of Computer Science **25**(04), 441–457 (2014). https://doi.org/10.1142/S012905411440005X
24. Manzoni, L., Porreca, A.E., Rozenberg, G.: Facilitation in reaction systems. Journal of Membrane Computing **2**(3), 149–161 (2020)

25. Okubo, F.: Reaction automata working in sequential manner. RAIRO-Theoretical Informatics and Applications-Informatique Théorique et Applications **48**(1), 23–38 (2014)
26. Okubo, F., Fujioka, K., Yokomori, T.: Chemical reaction regular grammars. New Gener. Comput. **40**(2), 659–680 (2022). https://doi.org/10.1007/S00354-022-00160-8
27. Okubo, F., Kobayashi, S., Yokomori, T.: On the properties of language classes defined by bounded reaction automata. Theoretical Computer Science **454**, 206–221 (2012)
28. Okubo, F., Kobayashi, S., Yokomori, T.: Reaction automata. Theoretical Computer Science **429**, 247–257 (2012). https://doi.org/https://doi.org/10.1016/j.tcs.2011.12.045, magic in Science
29. Okubo, F., Yokomori, T.: Recent developments on reaction automata theory: A survey. In: Recent Advances in Natural Computing: Selected Results from the IWNC 7 Symposium. pp. 1–22. Springer (2015)
30. Okubo, F., Yokomori, T.: The computational capability of chemical reaction automata. Nat. Comput. **15**(2), 215–224 (2016). https://doi.org/10.1007/S11047-015-9504-7, https://doi.org/10.1007/s11047-015-9504-7
31. Okubo, F., Yokomori, T.: The Computing Power of Determinism and Reversibility in Chemical Reaction Automata, pp. 279–298. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-73216-9_13, https://doi.org/10.1007/978-3-319-73216-9_13
32. Păun, Gh., Rozenberg, G.: A guide to membrane computing. Theoretical Computer Science **287**(1), 73–100 (2002), https://doi.org/10.1016/S0304-3975(02)00136-6
33. Păun, Gh.: Membrane Computing: An Introduction. Springer (2002)
34. Salomaa, A.: Minimal reaction systems: Duration and blips. Theoretical Computer Science **682**, 208–216 (2017)
35. Sosík, P.: The computational power of cell division in p systems: Beating down parallel computers? Natural Computing **2**, 287–298 (2003)
36. Sosík, P.: P systems attacking hard problems beyond NP: a survey. Journal of Membrane Computing **1**, 198–208 (2019), https://doi.org/10.1007/s41965-019-00017-y
37. Teh, W.C., Lim, J.: Evolvability of reaction systems and the invisibility theorem. Theoretical Computer Science **924**, 17–33 (2022). https://doi.org/10.1016/j.tcs.2022.03.039
38. Yokomori, T., Okubo, F.: Theory of reaction automata: a survey. J. Membr. Comput. **3**(1), 63–85 (2021). https://doi.org/10.1007/S41965-021-00070-6