Cycles and Global Attractors of Reaction Systems*

Enrico Formenti[1], Luca Manzoni[1], and Antonio E. Porreca[2]

[1] Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271
06900 Sophia Antipolis, France
formenti@unice.fr luca.manzoni@i3s.unice.fr
[2] Dipartimento di Informatica, Sistemistica e Comunicazione
Università degli Studi di Milano-Bicocca
Viale Sarca 336/14, 20126 Milano, Italy
porreca@disco.unimib.it

Abstract. Reaction systems are a recent formal model inspired by the chemical reactions that happen inside cells and possess many different dynamical behaviours. In this work we continue a recent investigation of the complexity of detecting some interesting dynamical behaviours in reaction system. We prove that detecting global behaviours such as the presence of global attractors is PSPACE-complete. Deciding the presence of cycles in the dynamics and many other related problems are also PSPACE-complete. Deciding bijectivity is, on the other hand, a coNP-complete problem.

1 Introduction

This paper completes the investigations started in [7], in which we studied the complexity of a collection of problems related to finding fixed points and local fixed points attractors in *reaction systems* (RS). Here we study the complexity of determining the existence of cycles and global attractors (either fixed points or cycles) in the dynamics of RS. In the first half of this investigation [7], the problems studied were either NP, coNP, or at most Π_2^{P}-complete. We show that moving from fixed points to cycles and from local to global attractors pushes the complexity to PSPACE in the majority of the cases. Recall that RS are a computational model, inspired by chemical reactions, recently introduced by Ehrenfeucht and Rozenberg [6]. After its introduction, many different aspects of the model were investigated [4, 5, 2, 13]. Indeed, the success of these systems is essentially due to the fact that they can be used to study practical problems [3] and, at the same time, they are clean and formal enough to allow formal investigations. Roughly speaking, a reaction system is made of some finite set and

* This work has been supported by the French National Research Agency project EMC (ANR-09-BLAN-0164) and by Fondo d'Ateneo (FA) 2013 of Università degli Studi di Milano-Bicocca: "Complessità computazionale in modelli di calcolo bioispirati: Sistemi a membrane e sistemi a reazioni".

a list of generating rules. The finite set consists of entities, or chemical species, that are used as reactants, inhibitor and products. Finally, a generation rule is activated if the reactants are present and if the inhibitors are not present and then it replaces the current set with the set of products.

The study of the dynamical behaviour of RS is one of the current trends in this domain. For example, in [5], the authors analysed the dynamical behaviour that can be obtained under the constraint of limited resources. In [4], the author considered some particular state as a death state (the empty set in this case) and computed the probability of a system to reach the death state.

Notice that, from a certain point of view, the dynamics of reaction systems are pretty well-known. Indeed, these are finite systems and hence their dynamics is always ultimately periodic. However, there are very interesting questions which are both useful for practical applications and highly non-trivial. For example, one can ask if a particular product will appear at a certain point of the dynamics or not [15, 14]. The present paper follows this trend by greatly extending the first results on complexity proved in [6, 15, 14], where the idea that RS can be used to evaluate Boolean formulae was introduced. In particular, we investigate the complexity of establishing if a RS admits a fixed point global attractor (PSPACE-complete). We also study the complexity of finding if two RS share all fixed points that are global attractors (PSPACE-complete). This is in some sense a concept of equivalence *w.r.t.* global attractors. We also explore the difficulty of finding if a state is part of cycle, a local attractor cycle, or a global attractor cycle, resulting in all cases in PSPACE-completeness. On the other hand, deciding if a RS admits a local attractor cycle is NP-complete. The other decision problems studied are about finding if two RS share one or all of their local attractor cycles (both PSPACE-complete).

The paper is structured as follows. Section 2 provides the basic notions on RS and two lemmata that will be used in the remaining part of the paper. Section 3 gives a description in logical terms of the problems we consider. The decision problems regarding global fixed points attractors are presented in Section 4. The decision problems regarding cycles are investigated in Section 5. A summary of the results and a hint at possible future developments is given in Section 6.

2 Basic Notions

This section provides a brief recollection of all the basic notions of RS and of dynamical systems necessary for the rest of the paper. Notations are taken from [6]. First of all, we recall the definitions of *reaction*, *reaction system*, and of their dynamics.

Definition 1. *Consider a finite set S, whose elements are called* entities. *A* reaction *a over S is a triple (R_a, I_a, P_a) of subsets of S. The set R_a is the set of* reactants*, I_a the set of* inhibitors*, and P_a is the set of* products*. The set of all reactions over S is denoted by* $\mathrm{rac}(S)$.

Definition 2. *A* reaction system *$\mathcal{A}$ is a pair (S, A) where S is a finite set, called the* background set*, and $A \subseteq \mathrm{rac}(S)$.*

Given a *state* $T \subseteq S$, a reaction a is said to be *enabled* in T when $R_a \subseteq T$ and $I_a \cap T = \varnothing$. The *result function* $\mathrm{res}_a \colon 2^S \to 2^S$ of a, where 2^S denotes the power set of S, is defined as

$$\mathrm{res}_a(T) = \begin{cases} P_a & \text{if } a \text{ is enabled in } T \\ \varnothing & \text{otherwise.} \end{cases}$$

The definition of res_a naturally extends to sets of reactions. Indeed, given $T \subseteq S$ and $A \subseteq \mathrm{rac}(S)$, define $\mathrm{res}_A(T) = \bigcup_{a \in A} \mathrm{res}_a(T)$. The result function $\mathrm{res}_{\mathcal{A}}$ of a RS $\mathcal{A} = (S, A)$ is res_A, *i.e.*, it is the result function on the whole set of reactions.

Example 1 (NAND gate). To implement a NAND gate using a RS we use as background set $S = \{0_a, 1_a, 0_b, 1_b, 0_{\mathrm{out}}, 1_{\mathrm{out}}\}$. The first four elements represent the two inputs (denoted by the subscripts a and b), the last two, on the other hand, denote the two possible outputs. The reactions used to model a NAND gate are the followings: $(\{0_a, 0_b\}, \varnothing, \{1_{\mathrm{out}}\})$, $(\{0_a, 1_b\}, \varnothing, \{1_{\mathrm{out}}\})$, $(\{1_a, 0_b\}, \varnothing, \{1_{\mathrm{out}}\})$, and $(\{1_a, 1_b\}, \varnothing, \{0_{\mathrm{out}}\})$. Similarly to NAND gates, others gates can be simulated and it is possible to build circuits with gates of limited fan-in using only a number of entities and reactions that is linear in the size of the modeled circuit.

After this brief introduction of RS, the reader may notice that there exist different bio-inspired models that are, in some sense, similar or related to RS. The most prominent are membrane systems [12], chemical reaction networks [17], and Boolean automata networks (BAN) [10, 16]. In the first two cases the object of the evolution is a multiset of symbols. In the last case one can show that, for each RS, there exists a BAN which simulates it and which is polynomial in the size of the given RS. However, an RS simulating a BAN may be exponentially larger than the original BAN [7].

We can now proceed by recalling the necessary definitions of the dynamical properties investigated in this work. Given a set $T \subseteq S$, the set of states visited by T is $(T, \mathrm{res}_{\mathcal{A}}(T), \mathrm{res}^2_{\mathcal{A}}(T), \ldots)$, that is, the sequence of $\mathrm{res}^i_{\mathcal{A}}(T)$ for $i \in \mathbb{N}$. Since 2^S is finite, every sequence of states is ultimately periodic, *i.e.*, there exists $h, k \in \mathbb{N}$ such that for all $t \in \mathbb{N}$, $\mathrm{res}^{h+kt}_{\mathcal{A}}(T) = \mathrm{res}^{h+k}_{\mathcal{A}}(T)$. The integer h is usually called the *length of the transient* and k the *length of the period*. A state $T \subseteq S$ is in a cycle if there exists $k \in \mathbb{N}$ such that $\mathrm{res}^k_{\mathcal{A}}(T) = T$. The smallest such k is the length of the cycle. If T is in a cycle of length 1 we say that T is a *fixed point*.

The notion of attractor is a central concept in the study of dynamical systems. Recall that an invariant set for $\mathcal{A}$ is a set of states $\mathcal{U}$ with $\bigcup_{U \in \mathcal{U}} \{\mathrm{res}_{\mathcal{A}}(U)\} = \mathcal{U}$. For RS all invariant sets consist of cycles. A local attractor in a RS is an invariant set $\mathcal{U}$ such that there exists $T \notin \mathcal{U}$ with $\mathrm{res}_{\mathcal{A}}(T) \in \mathcal{U}$. Intuitively, a local attractor is a set of states $\mathcal{U}$ from which the dynamics never escapes and such that there exists at least one external state whose dynamics ends up in $\mathcal{U}$. A global attractor for an RS $\mathcal{A}$ is an invariant set of states $\mathcal{U}$ such that for all $T \in 2^S$ there exists $t \in \mathbb{N}$ such that $\mathrm{res}^t_{\mathcal{A}}(T) \in \mathcal{U}$. A global attractor $\mathcal{U}$ a *global fixed-point attractor* if $\mathcal{U} = \{T\}$ and hence, necessarily, T is a fixed point. Similarly, we call $\mathcal{U}$ a *global attractor cycle* if all the states in $\mathcal{U}$ belong to the same cycle.

2.1 Counters and Turing Machines

We conclude this section with two technical results that are useful in the sequel. The first one tells us that it is always possible to build a RS containing a cycle of a given period. In the constructions that will follow, this cycle will be used as a kind of internal counter. This construction is inspired by Brijder et al. [2]. The second lemma ensures that it is possible to simulate a Turing machine (TM) with a family of RS and fixes the bounds for some simulation parameters. For an introduction, basic results, and notation on Turing machines we refer the reader to [8].

Lemma 1. *For every integer $k \geq 1$ there exists a RS $\mathcal{C}_k = (S, A)$ with background set of cardinality $n = |S| = \lceil \log_2 k \rceil$ having a cycle of period k reachable from any initial configuration after at most $2^n - k$ steps,* i.e.*,, $\mathcal{C}_k$ is a binary counter of n bits overflowing at $k \leq n$.*

Proof. Let $S = \{b_0, \ldots, b_{n-1}\}$. A subset B of S is interpreted as an n-bit integer m defined as $\sum_{b_i \in B} 2^i$, *i.e.*, the i-th bit of m is 1 iff $b_i \in B$. Let B_m be the subset of S representing the integer m. We shall design the reactions in A in order to obtain the following result function

$$\mathrm{res}_{\mathcal{C}_k}(B_m) = \begin{cases} B_{(m+1) \bmod k} & \text{if } m < k \\ B_{(m+1) \bmod 2^n} & \text{if } m \geq k \end{cases} \tag{1}$$

where $(x \bmod y)$ is the remainder of the integer division of x and y. The RS can thus be seen as a binary counter overflowing at k; if the state instead represents a number larger than $k-1$, the counter is increased normally until it overflows at 2^n.

The set A contains the following reactions:

$$(\{b_i\} \cup \{b_\ell\}, \{b_j\}, \{b_i\}) \qquad \text{for } 0 \leq i < n,\ 0 \leq j < i,\ b_\ell \notin B_{k-1} \tag{2}$$

$$(\{b_i\}, \{b_j\} \cup \{b_\ell\}, \{b_i\}) \qquad \text{for } 0 \leq i < n,\ 0 \leq j < i,\ b_\ell \in B_{k-1} \tag{3}$$

$$(\{b_0, \ldots, b_{i-1}\} \cup \{b_\ell\}, \{b_i\}, \{b_i\}) \qquad \text{for } 0 \leq i < n,\ b_\ell \notin B_{k-1} \tag{4}$$

$$(\{b_0, \ldots, b_{i-1}\}, \{b_i\} \cup \{b_\ell\}, \{b_i\}) \qquad \text{for } 0 \leq i < n,\ b_\ell \in B_{k-1}. \tag{5}$$

Reactions of types (2) and (3) preserve the bits set to 1 if there is a less significant bit set to 0, since they are not going to be modified by the increment operation. Reactions of types (4) and (5) set a currently null bit to 1 when all the less significant bits are 1. Notice that the empty state (corresponding to a null value of the counter) is mapped to state $\{b_0\}$ by the reactions of type (5) having no reactants. By construction, none of these reactions are enabled when the current state of $\mathcal{C}_k$ is B_{k-1}, because reactions of types (2) and (4) require the missing b_ℓ, and those of types (3) and (5) are inhibited by b_ℓ occurring in B_{k-1}.

Hence, the RS $\mathcal{C}_k$ defines the result function of Equation 1, and has a cycle of period k with a pre-period of at most $2^n - k < k$ steps. □

It is important to remark that the RS $\mathcal{C}_k$ of Lemma 1 can be constructed in polynomial time from the binary encoding of the integer k.

Theorem 1. *Let M be a Turing machine, $x \in \{0,1\}^\star$, and $m \geq |x|$ an integer. Then, there exists a RS $\mathcal{M} = \mathcal{M}_{M,m} = (S, A)$ and a state $X \subseteq S$ such that M reaches its final state q_F in t steps on input x, using at most m tape cells if and only if* $\mathrm{res}^t_{\mathcal{M}}(X) = \{q_\mathrm{F}\}$.

Proof. Let M be a Turing machine $(Q, \Sigma, \Gamma, \delta, q_\mathrm{I}, q_\mathrm{F})$, where Q is the set of states, $\Sigma = \{0,1\}$ is the input alphabet, $\Gamma = \Sigma \cup \{\triangleright\}$ is the tape alphabet, $q_\mathrm{I}, q_\mathrm{F} \in Q$ are the initial and final states, respectively (we assume $q_\mathrm{I} \neq q_\mathrm{F}$), and the transition function is $\delta\colon (Q - \{q_\mathrm{F}\}) \times \Gamma \to Q \times \Gamma \times \{-1, 0, +1\}$. Without loss of generality, we assume that M accepts by final state and rejects by diverging and that the tape is delimited on the left by $\triangleright$, a symbol that is never written or overwritten by M, and that forces the tape head to move to the right.

The RS $\mathcal{M}$ has $S = \{q_i : q \in Q - \{q_\mathrm{F}\}, -1 \leq i \leq m\} \cup \{q_\mathrm{F}\} \cup \{w_0, \ldots, w_{m-1}\}$ as its set of entities. A configuration of M is encoded as a subset of S as follows: q_i is present when the machine is in state q and the head is positioned on the i-th tape cell (the final state q_F has no subscript); the presence (resp., the absence) of w_i indicates that the i-th tape cell of M contains the symbol 1 (resp., the symbol 0). The state of the Turing machine tape in the RS is preserved by means of reactions which rewrite the entities w_i into themselves when there is no object q_i, (*i.e.*, the head of the Turing machine is not on the i-th tape cell). The entity q_i rewrite itself by looking at the presence or absence of the entity w_i (*i.e.*, by reading the tape) and according to the transition function of the Turing machine.

A transition $\delta(q, \sigma) = (r, \tau, d)$ of M not entering the final state is implemented by the following reactions

$$
\begin{array}{ll}
(\{q_i\}, \{w_i, q_\mathrm{F}\}, \{r_{i+d}\}) & \text{for } 0 \leq i < m, \text{ if } \sigma = 0,\ \tau = 0,\ r \neq q_\mathrm{F} \\
(\{q_i, w_i\}, \{q_\mathrm{F}\}, \{r_{i+d}\}) & \text{for } 0 \leq i < m, \text{ if } \sigma = 1,\ \tau = 0,\ r \neq q_\mathrm{F} \\
(\{q_i\}, \{w_i, q_\mathrm{F}\}, \{r_{i+d}, w_i\}) & \text{for } 0 \leq i < m, \text{ if } \sigma = 0,\ \tau = 1,\ r \neq q_\mathrm{F} \\
(\{q_i, w_i\}, \{q_\mathrm{F}\}, \{r_{i+d}, w_i\}) & \text{for } 0 \leq i < m, \text{ if } \sigma = 1,\ \tau = 1,\ r \neq q_\mathrm{F}.
\end{array}
$$

If the transition enters the final state, we produce the symbol q_F with no subscript, e.g., $\delta(q, 0) = (q_\mathrm{F}, 1, -1)$ is simulated by $(\{q_i\}, \{w_i, q_\mathrm{F}\}, \{q_\mathrm{F}, w_i\})$ for $0 \leq i < m$. If the tape head exceeds the space constraint of m cells (*i.e.*, we have an item of the form q_m), we do not generate any other state, effectively halting the simulation.

If the tape head reaches the $\triangleright$ symbol (cell -1) in state q, the RS simulates the subsequent transition $\delta(q, \triangleright) = (r, \triangleright, +1)$ as $(\{q_{-1}\}, \{q_\mathrm{F}\}, \{r_0\})$.

Finally, in order to preserve the portion of tape not currently scanned by M we use the reactions $(\{w_i\}, Q_i, \{w_i\})$ for $0 \leq i < m$, where $Q_i = \{q_i : q \in Q\}$.

Now let $X = \{q_{\mathrm{I},0}\} \cup \{w_i : x_i = 1\}$, where $x = x_1 \cdots x_n$ is the input string for M. Each iteration of $\mathrm{res}_{\mathcal{M}}$ starting from X simulates a computation step of M, as long as M does not exceed m tape cells; if this happens, the simulation stops at a fixed point state not containing q_F. The statement follows. □

Notice that the RS $\mathcal{M}$ of Theorem 1 can be built in polynomial time *w.r.t.* the size of the description of the Turing machine M, the length of the input x, and the integer m.

3 Logical description

This section recalls a logical description of RS and formulae related to their dynamics (see [7] for its first introduction) that will be used in the rest of the paper. This description (or a slight adaptation) will usually be sufficient for proving membership in many complexity classes. For the background notions of logic and descriptive complexity we refer the reader to the classical book of Neil Immerman [9].

We describe a RS $\mathcal{A} = (S, A)$ where the background set is $S \subseteq \{0, \ldots, n-1\}$ and $|A| \leq n$ by the vocabulary $(\mathsf{S}, \mathsf{R}_\mathcal{A}, \mathsf{I}_\mathcal{A}, \mathsf{P}_\mathcal{A})$, where S is a unary relation symbol and $\mathsf{R}_\mathcal{A}$, $\mathsf{I}_\mathcal{A}$, and $\mathsf{P}_\mathcal{A}$ are binary relation symbols. The symbols have the following intended meaning: the background set is $S = \{i : \mathsf{S}(i)\}$ and each of reaction $a_j = (R_j, I_j, P_j) \in A$ is described by the three sets $R_j = \{i \in S : \mathsf{R}_\mathcal{A}(i,j)\}$, $I_j = \{i \in S : \mathsf{I}_\mathcal{A}(i,j)\}$, and $P_j = \{i \in S : \mathsf{P}_\mathcal{A}(i,j)\}$.

We will also use some additional vocabularies: $(\mathsf{S}, \mathsf{R}_\mathcal{A}, \mathsf{I}_\mathcal{A}, \mathsf{P}_\mathcal{A}, \mathsf{T})$, where T is a unary relation that represents a subset of S, $(\mathsf{S}, \mathsf{R}_\mathcal{A}, \mathsf{I}_\mathcal{A}, \mathsf{P}_\mathcal{A}, \mathsf{T}_1, \mathsf{T}_2)$ with two additional unary relations that represent sets, and $(\mathsf{S}, \mathsf{R}_\mathcal{A}, \mathsf{I}_\mathcal{A}, \mathsf{P}_\mathcal{A}, \mathsf{R}_\mathcal{B}, \mathsf{I}_\mathcal{B}, \mathsf{P}_\mathcal{B})$ that denotes two RS having the same background set. The following formulae describe basic properties of $\mathcal{A}$. The first is true if a reaction a_j is enabled in T: $\textsc{en}_\mathcal{A}(j,T) \equiv \forall i(\mathsf{S}(i) \Rightarrow (\mathsf{R}_\mathcal{A}(i,j) \Rightarrow T(j)) \wedge (\mathsf{I}_\mathcal{A}(i,j) \Rightarrow \neg T(j)))$ the latter is: $\textsc{res}_\mathcal{A}(T_1,T_2) \equiv \forall i(\mathsf{S}(i) \Rightarrow (T_2(i) \Leftrightarrow \exists j(\textsc{en}_\mathcal{A}(j,T_1) \wedge \mathsf{P}_\mathcal{A}(i,j))))$ and it is verified if $\mathrm{res}_\mathcal{A}(T_1) = T_2$ for $T_1, T_2 \subseteq S$. Notice that $\textsc{en}_\mathcal{A}$ and $\textsc{res}_\mathcal{A}$ are both first-order (FO) formulae.

Since FO logic is insufficient for our purposes, we will formulate some problems using stronger logics: existential second order logic $\mathsf{SO}\exists$ characterising NP (Fagin's theorem); universally quantified second order logic $\mathsf{SO}\forall$ giving coNP; second order logic with a transitive closure operator ($\mathsf{SO}(\mathsf{TC})$, characterizing PSPACE). We denote the transitive closure of a formula $\varphi(X,Y)$ with two free second-order variables by $\varphi^\star(X,Y)$. We define the bounded second order quantifiers $(\forall X \subseteq Y)\varphi$ and $(\exists X \subseteq Y)\varphi$ as shorthands for $\forall X(\forall i(X(i) \Rightarrow Y(i)) \Rightarrow \varphi)$ and $\exists X(\forall i(X(i) \Rightarrow Y(i)) \wedge \varphi)$, respectively. We say that a formula is $\mathsf{SO}\exists$ or $\mathsf{SO}\forall$ if it is logically equivalent to a formula in the required prenex normal form.

4 Global attractors

The study of fixed points that are global attractors is closely related to the analysis of local fixed point attractors presented in [7]. However, the difficulty of the corresponding decision problems appears to be higher since we require that any point eventually evolves to the fixed point. Using $\mathsf{SO}(\mathsf{TC})$, one can define a formula $\textsc{path}_\mathcal{A}(T_1,T_2) \equiv \textsc{res}^\star_\mathcal{A}(T_1,T_2)$ to denote the existence of a (possibly empty) path in $\mathcal{A}$ from state T_1 to T_2. A formula expressing that T is a global fixed point attractor of $\mathcal{A}$ is $\textsc{glob}_\mathcal{A}(T) \equiv \textsc{fix}_\mathcal{A}(T) \wedge (\forall U \subseteq \mathsf{S})\ \textsc{path}_\mathcal{A}(U,T)$, where $\textsc{fix}_\mathcal{A}(T) \equiv \textsc{res}_\mathcal{A}(T,T)$.

The following theorem tells us that deciding if a given point is a global fixed point attractor is PSPACE-complete since it is possible to design a RS that

simulates a polynomial-space bounded Turing machine such that the dynamics of the system ends in fixed point iff the Turing machine halts.

Theorem 2. *Given a RS $\mathcal{A} = (S, A)$ and a state $T \subseteq S$, it is* PSPACE*-complete to decide whether T is a global attractor of $\mathcal{A}$.*

Proof. The problem is in PSPACE, since $\textsc{glob}_{\mathcal{A}}(T)$ is a SO(TC) formula. In order to show PSPACE-hardness we perform a reduction from the PSPACE-complete [11] problem

$$L = \{(M, x, 1^m) : M \text{ is a TM halting on } x \in \{0,1\}^\star \text{ in space } m \geq |x|\}.$$

Given $(M, x, 1^m)$, there are $|Q| \geq 2$ states of M, $m+1$ possible head positions (including cell -1 containing the left delimiter $\triangleright$), and 2^m possible strings on the tape, for a total of $k = |Q| \times (m+1) \times 2^m$ potential configurations.

We use a variant of the RS $\mathcal{C}_k = (S_{\mathcal{C}}, A_{\mathcal{C}})$ of Lemma 1 to count the configurations of M, which is simulated by means of a RS $\mathcal{M} = (S_{\mathcal{M}}, A_{\mathcal{M}})$, a modified version of that of Theorem 1. Denote by $\mathcal{A} = (S, A)$ the resulting RS.

We want the counter implemented by $\mathcal{C}_k$ to stop if and when the Turing machine simulated by $\mathcal{M}$ enters its final state q_{F}. Hence, we add q_{F} as an inhibitor to all reactions of $\mathcal{C}_k$.

Let $\blacktriangleleft\!\!\blacktriangleleft \notin S_{\mathcal{C}_k} \cup S_{\mathcal{M}}$ be a new entity. The entity $\blacktriangleleft\!\!\blacktriangleleft$ is used to reset the simulation of M to the initial configuration if the counter implemented by $\mathcal{C}_k$ reaches $k-1$ (*i.e.*, if the configuration of M are exhausted and the machine has entered a loop). To this purpose, $\blacktriangleleft\!\!\blacktriangleleft$ is added as an inhibitor to all reactions of $\mathcal{M}$, and we define the reaction $(B_{k-2}, (S_{\mathcal{C}} - B_{k-2}) \cup \{q_{\mathrm{F}}\}, \{\blacktriangleleft\!\!\blacktriangleleft\})$, which is enabled when the state of $\mathcal{A}$ restricted to $S_{\mathcal{C}_k}$ is B_{k-2}, *i.e.*, the representation of $k-2$ (see Lemma 1), and produces $\blacktriangleleft\!\!\blacktriangleleft$ when the counter reaches $k-1$. When $\blacktriangleleft\!\!\blacktriangleleft$ appears, it causes the restoration of the initial state of M by means of the reaction $(\{\blacktriangleleft\!\!\blacktriangleleft\}, \{q_{\mathrm{F}}\}, X)$, where $X = \{q_{\mathrm{I},0}\} \cup \{w_i : x_i = 1\}$ is the initial configuration of M on input $x = x_1 \cdots x_n$. One more reaction is needed to preserve the final state of M: $(\{q_{\mathrm{F}}\}, \varnothing, \{q_{\mathrm{F}}\})$. This is the only reaction enabled when the state of $\mathcal{A}$ contains q_{F}, ensuring that $\{q_{\mathrm{F}}\}$ is a fixed point.

Notice that $\{q_{\mathrm{F}}\}$ is the *only* fixed point, since in the absence of q_{F}, reactions incrementing the binary counter are always enabled (ensuring that the next state is different); on the other hand, if T contains q_{F} together with other entities it is never a fixed point, since $\mathrm{res}_{\mathcal{A}}(T) = \{q_{\mathrm{F}}\} \neq T$ for all $T \supsetneq \{q_{\mathrm{F}}\}$.

When the initial state of $\mathcal{A}$ is exactly X (hence, the binary counter is null) and M halts on x in space m, then, by Theorem 1, the RS reaches the state $\{q_{\mathrm{F}}\}$; on the other hand, if M does not halt or uses more than m tape cells, the binary counter eventually reaches $k-1$, resetting the configuration of $\mathcal{A}$ to X in the next step.

Any other initial state of $\mathcal{A}$ either reaches $\{q_{\mathrm{F}}\}$ before the counter reaches $k-1$ (this requires less than $2k$ steps, depending on the initial value of the counter as in Lemma 1), or the counter reaches $k-1$ and the state of $\mathcal{A}$ is set to X, once again eventually reaching $\{q_{\mathrm{F}}\}$ iff M halts on x in space m.

The mapping $(M, x, 1^m) \mapsto (\mathcal{A}, \{q_{\mathrm{F}}\})$ can be carried out in polynomial time and the PSPACE-hardness of the problem follows. □

By slightly tuning the proof of Theorem 2, it is possible to establish that determining if there exists a global fixed point attractor in a RS or if two RS share the same global fixed-point attractor are both PSPACE-complete problems. Furthermore, it follows directly from the previous proof that the reachability problem for RS is PSPACE-complete.

Corollary 1. *Given a RS $\mathcal{A} = (S, A)$ and two states $T_1, T_2 \subseteq S$, it is* PSPACE*-complete to decide whether T_2 is reachable from T_1,* i.e.*,* $\textsc{path}_{\mathcal{A}}(T_1, T_2)$ □

Corollary 2. *Given a RS $\mathcal{A} = (S, A)$, it is* PSPACE*-complete to decide whether $\mathcal{A}$ has a global fixed point attractor.*

Proof. The problem lies in PSPACE, since $(\exists T \subseteq \mathsf{S})\ \textsc{glob}_{\mathcal{A}}(T)$ is a SO(TC) formula. The reduction $(M, x, 1^m) \mapsto (\mathcal{A}, \{q_{\mathrm{F}}\})$ in the proof of Theorem 2 can be modified to obtain a reduction $(M, x, 1^m) \mapsto \mathcal{A}$. Notice that if $\mathcal{A}$ has a global fixed point attractor it is $\{q_{\mathrm{F}}\}$. Hence the PSPACE-hardness follows. □

Corollary 3. *Given two RS $\mathcal{A}$ and $\mathcal{B}$ over the same background set S, it is* PSPACE*-complete to decide whether they share a global fixed-point attractor.*

Proof. The problem lies in PSPACE since $(\exists T \subseteq \mathsf{S})(\textsc{glob}_{\mathcal{A}}(T) \wedge \textsc{glob}_{\mathcal{B}}(T))$ is a SO(TC) formula. The reduction $(M, x, 1^m) \mapsto (\mathcal{A}, \{q_{\mathrm{F}}\})$ in the proof of Theorem 2 can be adapted to a reduction $(M, x, 1^m) \mapsto (\mathcal{A}, \mathcal{B})$ where $\mathcal{B}$ is the RS having $(\varnothing, \varnothing, \{q_{\mathrm{F}}\})$ as its only reaction, *i.e.*, $\mathcal{B}$ has $\{q_{\mathrm{F}}\}$ as global attractor. The two RS share $\{q_{\mathrm{F}}\}$ as a global attractor iff $\{q_{\mathrm{F}}\}$ is a global attractor also for $\mathcal{A}$, that is to say, iff M accepts x in space m; Hence, the PSPACE-hardness follows. □

5 Cycles

We now turn our attention to the only other possible behaviour of RS, namely having cycles of length greater than 1. Notice that, while we only need to analyse the application of the result function when checking properties related to fixed points, dealing with cycles essentially involves reachability problems. This shifts the complexity of most of the decision problems to PSPACE.

Theorem 3. *Given a RS $\mathcal{A} = (S, A)$ and a state $T \subseteq S$, it is* PSPACE*-complete to decide whether T is part of a cycle,* i.e.*, whether* $\mathrm{res}^t_{\mathcal{A}}(T) = T$ *for some* $t \in \mathbb{N}$.

Proof. The problem lies in PSPACE, since it can be expressed by the following SO(TC) formula: $\textsc{cycle}_{\mathcal{A}}(T) \equiv \exists U(\textsc{res}_{\mathcal{A}}(T, U) \wedge \textsc{path}_{\mathcal{A}}(U, T))$. The reduction $(M, x, 1^m) \mapsto (\mathcal{A}, \{q_{\mathrm{F}}\})$ of Theorem 2 can be adapted to a reduction $(M, x, 1^m) \mapsto (\mathcal{A}, X)$, where X is the encoding of the initial state of the Turing machine. If M accepts x using space m, then the dynamics of X eventually reaches the fixed point $\{q_{\mathrm{F}}\}$, and X is not in a cycle; conversely, the state is eventually rewritten into X, *i.e.*, X belongs to a cycle. This proves that the problem is PSPACE-hard. Since the complement of a PSPACE-complete problem is also PSPACE-complete, the statement follows. □

Since every RS is finite, a cycle always exists. Therefore, in this case, it turn out to be much more interesting to study comparison problems between RS. That is, if they share one (resp., every) cycle.

Theorem 4. *Given two RS $\mathcal{A}$ and $\mathcal{B}$ over the same background set S, it is* PSPACE*-complete to decide whether they share a common cycle.*

Proof. The problem lies in PSPACE since it can be expressed as a SO(TC) formula. Let $\textsc{res}_{\mathcal{A},\mathcal{B}}(T,U) \equiv \textsc{res}_{\mathcal{A}}(T,U) \wedge \textsc{res}_{\mathcal{B}}(T,U)$ be the formula denoting the fact that T has the same image U under both $\textsc{res}_{\mathcal{A}}$ and $\textsc{res}_{\mathcal{B}}$. From it we define $\textsc{path}_{\mathcal{A},\mathcal{B}}(T,U) \equiv \textsc{res}^{\star}_{\mathcal{A},\mathcal{B}}(T,U)$ to denote that the same path leads from T to U in $\mathcal{A}$ and $\mathcal{B}$. We then express the fact that T belongs to a shared cycle by means of the formula $\textsc{cycle}_{\mathcal{A},\mathcal{B}}(T) \equiv \exists U(\textsc{res}_{\mathcal{A},\mathcal{B}}(T,U) \wedge \textsc{path}_{\mathcal{A},\mathcal{B}}(U,T))$. Finally, $(\exists T \subseteq \mathsf{S})\,\textsc{cycle}_{\mathcal{A},\mathcal{B}}(T)$ is a SO(TC) formula denoting the existence of a shared cycle between $\mathcal{A}$ and $\mathcal{B}$.

Consider the reduction $(M,x,1^m) \mapsto (\mathcal{A},\{q_{\mathrm{F}}\})$ of Theorem 2, this can be transformed into a reduction $(M,x,1^m) \mapsto (\mathcal{A},\mathcal{B})$, where $\mathcal{B}$ is equal to $\mathcal{A}$ except that the reaction $(\{q_{\mathrm{F}}\},\varnothing,\{q_{\mathrm{F}}\})$ is replaced by $(\{q_{\mathrm{F}}\},\varnothing,X)$, in which X represents the initial configuration of M on input x. The two RS behave as follows:

- If M halts on x within space m, then $\mathcal{A}$ has only the fixed point $\{q_{\mathrm{F}}\}$ as a cycle, while $\mathcal{B}$ has a cycle going from X to $\{q_{\mathrm{F}}\}$ and immediately back to X.
- Otherwise, $\mathcal{A}$ has two cycles: the fixed point $\{q_{\mathrm{F}}\}$ and the cycle starting from X and going back to X (when the binary counter overflows). The latter also exists in $\mathcal{B}$, since the behaviours of the two systems only differ *w.r.t.* states containing q_{F}.

Hence, $\mathcal{A}$ and $\mathcal{B}$ share a cycle if and only if M does *not* halt on input x within space m. Hence, the PSPACE-hardness of the problem follows. □

Theorem 5. *Given two RS $\mathcal{A}$ and $\mathcal{B}$ over the same background set S, it is* PSPACE*-complete to decide whether they share all their cycles.*

Proof. The problem is in PSPACE, since it can be expressed by the following SO(TC) formula: $(\forall T \subseteq \mathsf{S})(\textsc{cycle}_{\mathcal{A}}(T) \vee \textsc{cycle}_{\mathcal{B}}(T) \Rightarrow \textsc{cycle}_{\mathcal{A},\mathcal{B}}(T))$.

The reduction $(M,x,1^m) \mapsto (\mathcal{A},\{q_{\mathrm{F}}\})$ of Theorem 2 can be transformed into the reduction $(M,x,1^m) \mapsto (\mathcal{A},\mathcal{B})$, where $\mathcal{B}$ is the RS having $(\varnothing,\varnothing,\{q_{\mathrm{F}}\})$ as its only reaction. The fixed point $\{q_{\mathrm{F}}\}$ is the only cycle of $\mathcal{B}$, and it is shared by both systems. Remark that $\{q_{\mathrm{F}}\}$ is the only cycle of $\mathcal{A}$ only if M halts on x within space m. This reduction establishes the PSPACE-hardness. □

Theorem 6. *Given a RS $\mathcal{A} = (S,A)$ and a state $T \subseteq S$, it is* PSPACE*-complete to decide whether T is part of a local attractor cycle.*

Proof. Since $\textsc{attc}_{\mathcal{A}}(T) \equiv \textsc{cycle}_{\mathcal{A}}(T) \wedge (\exists U \subseteq \mathsf{S})(\textsc{path}_{\mathcal{A}}(U,T) \wedge \neg \textsc{path}_{\mathcal{A}}(T,U))$ is a SO(TC) formula, the problem lies in PSPACE.

We can transform the reduction $(M,x,1^m) \mapsto (\mathcal{A},\{q_{\mathrm{F}}\})$ of Theorem 2 into the reduction $(M,x,1^m) \mapsto (\mathcal{A},X)$, where X is the encoding of the initial state

of the Turing machine. Notice that if X belongs to a cycle then this cycle is an attractor. Irrespective of the behaviour of M, state X has at least two preimages, namely $\{\blacktriangleleft\}$ and $B_{k-1} \cup \{\blacktriangleleft\}$ (notice that $B_{k-1} \neq \varnothing$ since $k \geq 2$). Finally, if (and only if) M does not halt on x in space m, starting from the state X we eventually reach X again when the binary counter overflows. This shows that the problem is PSPACE-hard. □

Theorem 7. *Given a RS $\mathcal{A} = (S, A)$, deciding if $\mathrm{res}_{\mathcal{A}}$ is a bijection is* coNP*-complete.*

Proof. It is possible to express bijectivity with the following SO∀ formula:

$$\textsc{bij}_{\mathcal{A}} \equiv (\forall T \subseteq \mathsf{S})(\forall U \subseteq \mathsf{S})(\forall V \subseteq \mathsf{S})(\textsc{res}_{\mathcal{A}}(T, V) \wedge \textsc{res}_{\mathcal{A}}(U, V) \Rightarrow \textsc{eq}(T, U))$$

where $\textsc{eq}(T, U) \equiv \forall i(\mathsf{S}(i) \Rightarrow (T(i) \Leftrightarrow U(i)))$. Hence the problem is in coNP.

We reduce TAUTOLOGY [6] (also known as VALIDITY [11]) to this problem. Let φ be a Boolean formula in DNF (*i.e.*, $\varphi = \varphi_1 \vee \ldots \vee \varphi_m$, where each φ_j, for $1 \leq j \leq m$, is a conjunctive clause) over the variables $V = \{x_1, \ldots, x_n\}$.

Let $\mathcal{C}_{2^n} = (V, A')$ be the RS of Lemma 1, implementing a binary counter ranging over the set $\{0, \ldots, 2^n - 1\}$, with the entities $b_0, \ldots, b_{n-1}$ renamed as $x_1, \ldots, x_n$. Let $\mathcal{A}$ be a RS with $S = V \cup \{\heartsuit\}$ and having all the reactions of A' plus the following ones:

$$(\mathrm{pos}(\varphi_j) \cup \{\heartsuit\}, \mathrm{neg}(\varphi_j), \{\heartsuit\}) \qquad \text{for } 1 \leq j \leq m \tag{6}$$

where $\mathrm{pos}(\varphi_j)$ and $\mathrm{neg}(\varphi_j)$ denote the variables appearing, respectively, as positive and negative literals in φ_j. A set $X_i \subseteq V$ represents both the integer i, as in the proof of Lemma 1, and a truth assignment of φ, where the variables having a true value are those in X_i. If the state of $\mathcal{A}$ has the form $X_i \cup \{\heartsuit\}$, then φ is evaluated under X_i, preserving $\heartsuit$ if satisfied, by reaction (6); in any case, the subset of the state representing i is incremented modulo 2^n by the reactions in A'. Hence, the result function of $\mathcal{A}$ is

$$\mathrm{res}_{\mathcal{A}}(T) = \begin{cases} \{\heartsuit\} \cup X_{(i+1) \bmod 2^n} & \text{if } T = \{\heartsuit\} \cup X_i \text{ and } X_i \vDash \varphi \\ X_{(i+1) \bmod 2^n} & \text{if } T = X_i \text{ or if } T = \{\heartsuit\} \cup X_i \text{ and } X_i \nvDash \varphi. \end{cases}$$

If φ is not a tautology, then there exists an assignment X_i such that $X_i \nvDash \varphi$, thus $\mathrm{res}_{\mathcal{A}}(X_i)$ is equal to $\mathrm{res}_{\mathcal{A}}(X_i \cup \{\heartsuit\})$, *i.e.*, $\mathrm{res}_{\mathcal{A}}$ is not bijective. On the other hand, if φ is a tautology, then $\heartsuit$ is always preserved when present, and the dynamics of $\mathcal{A}$ consists of two disjoint cycles, namely $(X_0, \ldots, X_{2^n-1})$ and $(X_0 \cup \{\heartsuit\}, \ldots, X_{2^n-1} \cup \{\heartsuit\})$, *i.e.*, $\mathrm{res}_{\mathcal{A}}$ is a bijection. Since the mapping $\varphi \mapsto \mathcal{A}$ is computable in polynomial time, the problem is coNP-hard. □

Corollary 4. *Given a RS $\mathcal{A} = (S, A)$, it is* NP*-complete to decide whether $\mathcal{A}$ has a local attractor cycle.*

Proof. A finite system has an attractor cycle if and only if the next-state function is not a bijection. Hence, this is the complement of the coNP-complete problem of Theorem 7. □

Theorem 8. *Given two RS $\mathcal{A}$ and $\mathcal{B}$ over the same background set S, it is* PSPACE*-complete to decide whether $\mathcal{A}$ and $\mathcal{B}$ have a common local attractor cycle.*

Proof. The problem is in PSPACE since $(\exists T \subseteq \mathsf{S})\ \textsc{attc}_{\mathcal{A},\mathcal{B}}(T)$ is a SO(TC) formula, where

$$\textsc{attc}_{\mathcal{A},\mathcal{B}}(T) \equiv \textsc{cycle}_{\mathcal{A},\mathcal{B}}(T) \wedge (\exists U \subseteq \mathsf{S})(\textsc{path}_{\mathcal{A}}(U,T) \wedge \neg\textsc{path}_{\mathcal{A}}(T,U)) \wedge (\exists V \subseteq \mathsf{S})(\textsc{path}_{\mathcal{B}}(V,T) \wedge \neg\textsc{path}_{\mathcal{B}}(T,V)).$$

Consider the reduction $(M, x, 1^m) \mapsto (\mathcal{A}, \mathcal{B})$ of Theorem 4, and notice that X, the state encoding the initial configuration of M, has at least two distinct preimages (see the proof of Theorem 6). Hence, when $\mathcal{A}$ and $\mathcal{B}$ share a common cycle, it is always an attractor cycle. The PSPACE-hardness follows. □

Theorem 9. *Given two RS $\mathcal{A}$ and $\mathcal{B}$ over the same background set S, it is* PSPACE*-complete to decide whether $\mathcal{A}$ and $\mathcal{B}$ share all their local attractor cycles.*

Proof. The problem is in PSPACE since it can be expressed by the following SO(TC) formula: $(\forall T \subseteq \mathsf{S})(\textsc{attc}_{\mathcal{A}}(T) \vee \textsc{attc}_{\mathcal{B}}(T) \Rightarrow \textsc{attc}_{\mathcal{A},\mathcal{B}}(T))$.

Consider the reduction $(M, x, 1^m) \mapsto (\mathcal{A}, \mathcal{B})$ of the proof of Theorem 5; both RS share the fixed point $\{q_{\mathrm{F}}\}$, which is an attractor since every state containing q_{F} is mapped to $\{q_{\mathrm{F}}\}$. When M does *not* halt on x in space m, $\mathcal{A}$ has another cycle containing X, which is an attractor as shown in the proof of Theorem 6. This is enough to establish the PSPACE-hardness. □

Since a global attractor state is a special case of global attractor cycle, and the corresponding decision problems remain in PSPACE, we immediately have the following statement:

Corollary 5. *Given two RS $\mathcal{A}$ and $\mathcal{B}$ over the same background set S and a state $T \subseteq S$, it is* PSPACE*-complete to decide if (i) T is a part of a global attractor cycle in $\mathcal{A}$, (ii) $\mathcal{A}$ has a global attractor cycle, (iii) $\mathcal{A}$ and $\mathcal{B}$ have a common global attractor cycle.* □

6 Conclusions

In this paper we have studied the complexity of checking the presence of many different dynamical behaviours of a RS, extending the work started in [7]. When global fixed point attractors are considered, all problems are PSPACE-complete, differently from the case of local fixed points attractors, where all the problems lied in the polynomial hierarchy. We proved that PSPACE-completeness remains the most common complexity class for the decision problems regarding cycles that we analysed. While some PSPACE-completeness results are known for more expressive or different computational models [1], it is interesting that, even if they are quite simple, RS exhibit difficult decision problems.

The paper, while closing some open question, still discloses many interesting research directions. In particular, we have only studied deterministic RS, *i.e.*, the next-state is uniquely determined. However many significant modelling questions involve non-deterministic RS where at every time step an external device inserts some entities in the state of the RS (these kind of RS are called *RS with context* in the literature). It is interesting to understand how the complexity of decision problems about dynamics behaves in this case. Does everything shift to PSPACE? Is PSPACE the upper bound?

References

1. Barrett, C.L., Hunt III, H.B., Marathe, M.V., Ravi, S., Rosenkrantz, D.J., Stearns, R.E.: Complexity of reachability problems for finite discrete dynamical systems. Int. J. Found. Comput. Sci. 72(8), 1317–1345 (2006)
2. Brijder, R., Ehrenfeucht, A., Rozenberg, G.: Reaction systems with duration. In: Kelemen, J., Kelemenová, A. (eds.) Computation, Cooperation, and Life, LNCS, vol. 6610, pp. 191–202. Springer (2011)
3. Corolli, L., Maj, C., Marini, F., Besozzi, D., Mauri, G.: An excursion in reaction systems: From computer science to biology. Theor. Comp. Sci. 454, 95–108 (2012)
4. Ehrenfeucht, A., Main, M., Rozenberg, G.: Combinatorics of life and death for reaction systems. Int. J. Found. Comput. Sci. 21(03), 345–356 (2010)
5. Ehrenfeucht, A., Main, M., Rozenberg, G.: Functions defined by reaction systems. Int. J. Found. Comput. Sci. 22(1), 167–168 (2011)
6. Ehrenfeucht, A., Rozenberg, G.: Reaction systems. Fundam. Inform. 75, 263–280 (2007)
7. Formenti, E., Manzoni, L., Porreca, A.E.: Fixed points and attractors of reaction systems. In: Beckmann, A., Csuhaj-Varjú, E., Meer, K. (eds.) Language, Life, Limits, 10th Conference on Computability in Europe, CiE 2014. LNCS, vol. 8493. Springer (2014), to appear
8. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley (1979)
9. Immerman, N.: Descriptive Complexity. Graduate Texts in Computer Science, Springer (1999)
10. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. J. Theor. Biol. 22(3), 437–467 (1969)
11. Papadimitriou, C.H.: Computational Complexity. Addison-Wesley (1993)
12. Păun, Gh.: Computing with membranes. Journal of Computer and System Sciences 61(1), 108–143 (2000)
13. Salomaa, A.: Functions and sequences generated by reaction systems. Theoretical Computer Science 466, 87–96 (2012)
14. Salomaa, A.: Functional constructions between reaction systems and propositional logic. Int. J. Found. Comput. Sci. 24(1), 147–159 (2013)
15. Salomaa, A.: Minimal and almost minimal reaction systems. Natural Computing 12(3), 369–376 (2013)
16. Shmulevich, I., Dougherty, E.R.: Probabilistic boolean networks: the modeling and control of gene regulatory networks. SIAM (2010)
17. Soloveichik, D., Cook, M., Winfree, E., Bruck, J.: Computation with finite stochastic chemical reaction networks. Natural Computing 7, 615–633 (2008)