Ancestors, descendants, and gardens of Eden in reaction systems[1]

Alberto Dennunzio[a], Enrico Formenti[b], Luca Manzoni[a], Antonio E. Porreca[a]

[a]*Dipartimento di Informatica, Sistemistica e Comunicazione*
Università degli Studi di Milano-Bicocca
Viale Sarca 336/14, 20126 Milano, Italy
[b]*Université Nice Sophia Antipolis, CNRS, I3S, UMR 7271*
06900 Sophia Antipolis, France

Abstract

This paper analyses several problems related to finding and counting ancestor and descendant states, as well as gardens of Eden (i.e., states without predecessors) in reaction systems. The focus is on the complexity of finding and counting preimages and ancestors that are minimal with respect to cardinality. It turns out that the problems concerning gardens of Eden seem to require the presence of a **NP**-oracle to be solved. All the problems studied are intractable, with a complexity that ranges form $\mathbf{FP}^{\mathbf{NP}[\log n]}$ to **FPSPACE**(poly).

Keywords: natural computing, reaction systems, computational complexity, discrete dynamical systems

1. Introduction

Recently many new computational models have been introduced. Most of them are inspired by natural phenomena. This is also the case of Reaction Systems (RS), proposed by Ehrenfeucht and Rozenberg in [4], which are a metaphor for basic chemical reactions. Informally, a reaction system is made of a (finite) set of entities (molecules) and a (finite) set of reactions. Each reaction is a triple of sets: *reactants*, *inhibitors* and *products* (clearly the set of reactants and the one of inhibitors are disjoint). Given a set of reactants T, a reaction (R, I, P) is applied if $R \subseteq T$ and if there are no inhibitors (i.e., $T \cap I$ is empty); the result is the replacement of T by the set of products P. Given a set of reactants T, all enabled reactions are applied in parallel. The final set of products is the union of all single sets of products of each reaction which is enabled in T.

Studying RS is interesting for a number of reasons, not only as a clean computational model allowing precise formal analysis (see [2] for combinatorics issues) but also as a reference with respect to other computing systems. For instance, in [7], the authors showed an embedding of RS into Boolean automata networks (BAN), a well-known model used in a number of application domains. Remark that for BAN the precise complexity of only a bunch of problems about the dynamical behaviour is known. Via the embedding of RS into BAN, all the complexity results about RS are indeed lower bounds for the corresponding ones for BAN.

In this paper, we continue the exploration of the computational complexity of properties of RS started in [7, 6]. The focus is on preimages and ancestors of minimal size. In more practical terms, this could be useful when minimising the number of chemical entities necessary to obtain a target compound. Indeed, given a current state T, the minimal preimage (resp., n-th-ancestor) problem or MPP (resp., MAP) consists in finding the minimal set (with respect to cardinality) of reactants which produces T in one step (resp., n steps). Variants of MPP and MAP consider counting the number of preimages (#PP) and of minimal preimages (#MPP); counting the number of ancestors (#AP) and of minimal ancestors (#MAP); or computing the size of a minimal preimage (SMPP) or of a minimal ancestor (SMAP). We also count the number of descendants of a state (#DP). Finally, we investigate problems related to gardens of Eden, namely finding them (GEP) and counting them (#GEP).

[1] This paper is an extended and improved version of the paper [3], presented at the conference LATA 2015.

Email addresses: dennunzio@disco.unimib.it (Alberto Dennunzio), enrico.formenti@unice.fr (Enrico Formenti), luca.manzoni@disco.unimib.it (Luca Manzoni), porreca@disco.unimib.it (Antonio E. Porreca)

More precisely, we prove that (Section 2 recalls the definitions of the complexity classes):

- #PP is **#P**-complete under parsimonious reductions;
- MPP $\in \mathbf{FP}^{\mathbf{NP}}$ and it is $\mathbf{FP}^{\mathbf{NP}}_{\|}$-hard under metric reductions;
- #MPP is in $\#\mathbf{P}^{\mathbf{NP}}$ and it is **#P**-hard under parsimonious reductions;
- SMPP is $\mathbf{FP}^{\mathbf{NP}[\log n]}$-complete under metric reductions;
- #AP is **FPSPACE**(poly)-complete under Cook reductions;
- MAP, #MAP, and #DP are complete for **FPSPACE**(poly) under metric reductions;
- SMAP is **FPSPACE**(log)-complete under metric reductions.
- GEP is $\mathbf{FNP}^{\mathbf{NP}}$-complete under metric reductions;
- #GEP is $\#\mathbf{P}^{\mathbf{NP}}$-complete under metric reductions.

These results are important for further understanding the computational capabilities of RS but they also provide clean new items to the (relatively) short list of examples of problems in high functional complexity classes. Remark that the problem of preimage existence has been proved to be in **NP** by Salomaa [12]. However, here the complexity is higher because of the minimality requirement.

The paper is structured as follows: in Section 2 the necessary background notions are recalled; Section 3 is concerned with preimages problems, while Section 4 with ancestors problems. Gardens of Eden are studied in Sections 5. Finally, in Section 6 we draw our conclusions and provide some open questions.

2. Basic Notions

This section briefly recalls the basic notions about RS as introduced in [5]. It is important to remark that throughout the whole paper the sets of reactants and inhibitors of a reaction are required to be nonempty, as it is sometimes enforced in the literature. However, the results also hold when empty sets are allowed, unless explicitly specified otherwise.

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 nonempty 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 (RS)* is a pair $\mathcal{A} = (S, A)$ 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 $I_a \cap T = \emptyset$. 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 $\mathrm{res}_a(T) = P_a$ if a is enabled in T, and $\mathrm{res}_a(T) = \emptyset$ otherwise. The definition of res_a naturally extends to sets of reactions: 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., the result function on the whole set of reactions. In this way, any RS $\mathcal{A} = (S, A)$ induces a discrete dynamical system where the state set is 2^S and the next state function is $\mathrm{res}_{\mathcal{A}}$.

Definition 3. Let $\mathcal{A} = (S, A)$ be a RS. For any $T \subseteq S$, an element $U \subseteq S$ is an *ancestor* of T if $\mathrm{res}^t_{\mathcal{A}}(U) = T$ for some $t \in \mathbb{N}$. In that case T is a *descendant* of U. If $t = 1$, U is called *preimage* of T. An ancestor (resp., preimage) U of T is *minimal* if $|U| \leq |V|$ for all ancestors (resp., preimages) V of T, where $|X|$ denotes the cardinality of set X.

A state always admits at least itself as ancestor but might not have a preimage. A state without preimages is called a *garden of Eden*. In dynamical systems admitting only a finite number of states, as RS are, either the next state function is bijective or the system has at least one garden of Eden.

The *orbit* of a given a state T of a RS $\mathcal{A}$ is defined as the sequence of states obtained by iterations of $\mathrm{res}_{\mathcal{A}}$ starting from T, namely the sequence $(T, \mathrm{res}_{\mathcal{A}}(T), \mathrm{res}^2_{\mathcal{A}}(T), \ldots)$. Being finite systems, RS only admit ultimately periodic orbits, i.e., orbits ending up in a cycle.

The complexity of preimage and ancestor problems for RS is conveniently described by complexity classes of function problems. For this reason, the rest of this section recalls the definitions of the relevant ones (for more details, see [10, 9]).

Let Σ be an alphabet. We say that a binary relation R over $\Sigma^\star$ is *polynomially-balanced* if there exists a polynomial p such that $R(x,y)$ implies $|y| \leq p(|x|)$, where $|x|$ denotes the length of the string x. A "choice function" for R is a function $f : \mathrm{dom}\, R \to \Sigma^\star$ that, for a given x, returns any y such that $R(x,y)$. The class **FP** (resp., $\mathbf{FP}^{\mathbf{NP}}$) consists of all polynomially-balanced binary relations R over $\Sigma^\star$ having a choice function computable in polynomial time by a deterministic Turing machine (TM) without access to oracles (resp., with access to an oracle for an **NP** decision problem). Additional requirements on the oracle queries define the subclasses $\mathbf{FP}^{\mathbf{NP}[\log n]}$ and $\mathbf{FP}^{\mathbf{NP}}_{\parallel}$ of $\mathbf{FP}^{\mathbf{NP}}$, where the number of allowed oracle queries is $O(\log n)$ and the queries are performed in parallel (i.e., every current query string does not depend on the results of previous queries), respectively. We have $\mathbf{FP} \subseteq \mathbf{FP}^{\mathbf{NP}[\log n]} \subseteq \mathbf{FP}^{\mathbf{NP}}_{\parallel} \subseteq \mathbf{FP}^{\mathbf{NP}}$.

The class **FNP** contains all polynomially-balanced binary relations R such that, given two strings (x,y), it is decidable in deterministic polynomial time whether $R(x,y)$ holds. Equivalently, a multi-valued partial function f is in **FNP** if its possible outputs on input x correspond to the possible outputs of a nondeterministic polynomial-time TM. By adding an **NP** oracle, we obtain the class $\mathbf{FNP}^{\mathbf{NP}}$.

The class **#P** consists of all functions $f : \Sigma^\star \to \mathbb{N}$ with a polynomial-time nondeterministic TM having exactly $f(x)$ accepting computations on every input x. If in addition an **NP** oracle is allowed, the class $\#\mathbf{P}^{\mathbf{NP}}$ is defined; we can always assume that at most one query is performed [13], that is, we have $\#\mathbf{P}^{\mathbf{NP}} = \#\mathbf{P}^{\mathbf{NP}[1]}$. Clearly, $\mathbf{FP} \subseteq \#\mathbf{P} \subseteq \#\mathbf{P}^{\mathbf{NP}[1]} = \#\mathbf{P}^{\mathbf{NP}}$.

In this paper we will also refer to **FPSPACE**, i.e., the collection of binary relations having a choice function computable in polynomial space using a separate, write-only output tape with no space restriction, and its two subclasses **FPSPACE**(poly) and **FPSPACE**(log), in which the output is limited to polynomial length and logarithmic length, respectively, rather than exponential length. Remark that **FPSPACE**(poly) is just ♮**PSPACE**, i.e., the class of functions $f : \Sigma^\star \to \mathbb{N}$ such that there exists a polynomial-space nondeterministic TM performing only a polynomial number of nondeterministic choices and having exactly $f(x)$ accepting computations on every input x.

Hardness for these classes is defined in terms of three kinds of reductions. The first one is the many-one reduction, also called *parsimonious* when dealing with counting problems: a function f is many-one reducible to g if there exists a function $h \in \mathbf{FP}$ such that $f(x) = g(h(x))$ for every input x. A generalisation is the *metric reduction* [8]: a function f is metric-reducible to g if there exist functions $h_1, h_2 \in \mathbf{FP}$ such that $f(x) = h_2(x, g(h_1(x)))$ for every input x. Equivalently, f is metric-reducible to g if $f \in \mathbf{FP}^{g[1]}$, that is, it can be computed in polynomial time by making a single query to an oracle for g. Metric reductions can be further generalised to *Cook reductions*: f is Cook-reducible to g if $f \in \mathbf{FP}^{g}$, that is, any number of oracle queries is allowed. These notions of reduction can be generalised to reductions between binary relations.

Since we do not deal with sublinear space complexity, throughout this paper it is assumed, without loss of generality, that all TM computing functions use a unique tape, both for input and work. We also assume that they move their tape head to the leftmost cell before entering a final state.

3. Preimage Problems

First of all, inspired by the algorithm described in [11], we show a tight relation between the MPP and the problem of finding a minimal unary travelling salesman tour (TSP) [10], where the edge weights are encoded in unary and, hence, bounded by a polynomial in the number of vertices.

Lemma 4. *Finding a minimal unary TSP tour is metric reducible to* MPP.

Proof. Suppose we are given any set of vertices V, with $|V| = n$, and any unary-encoded weight function $w\colon V^2 \to \mathbb{N}$. We build a RS $\mathcal{A} = (S, A)$ admitting a state whose preimages encode the weighted simple cycles over V in unary. The background set is given by $S = E \cup W \cup \{\heartsuit, \spadesuit\}$, where

$$E = \{(u,v)_t : u, v \in V \text{ and } 0 \le t < n\}$$
$$W = \{\diamondsuit_{(u,v),i} : u, v \in V \text{ and } 1 \le i \le w(u,v)\}$$

while the reactions in A, with u, v, u_1, u_2, v_1, v_2 ranging over V and t, t_1, t_2 ranging over $\{0, \dots, n-1\}$, are

$$(\{(u_1,v_1)_t, (u_2,v_2)_t, \heartsuit\}, \{\spadesuit\}, \{\spadesuit\}) \quad \text{if } (u_1,v_1) \neq (u_2,v_2) \tag{1}$$
$$(\{(u,v_1)_{t_1}, (u,v_2)_{t_2}, \heartsuit\}, \{\spadesuit\}, \{\spadesuit\}) \quad \text{if } v_1 \neq v_2 \text{ or } t_1 \neq t_2 \tag{2}$$
$$(\{(u_1,v)_{t_1}, (u_2,v)_{t_2}, \heartsuit\}, \{\spadesuit\}, \{\spadesuit\}) \quad \text{if } u_1 \neq u_2 \text{ or } t_1 \neq t_2 \tag{3}$$
$$(\{(u_1,v_1)_t, (u_2,v_2)_{(t+1) \bmod n}, \heartsuit\}, \{\spadesuit\}, \{\spadesuit\}) \quad \text{if } v_1 \neq u_2 \tag{4}$$
$$(\{\heartsuit\}, \{(u,v)_t : u, v \in V\} \cup \{\spadesuit\}, \{\spadesuit\}) \tag{5}$$
$$(\{(u,v)_t, \heartsuit\}, \{\diamondsuit_{(u,v),i}, \spadesuit\}, \{\spadesuit\}) \quad \text{for } 1 \le i \le w(u,v) \tag{6}$$
$$(\{\diamondsuit_{(u,v),i}, \heartsuit\}, \{(u,v)_t : 0 \le t < n\} \cup \{\spadesuit\}, \{\spadesuit\}) \quad \text{for } 1 \le i \le w(u,v) \tag{7}$$
$$(\{\heartsuit\}, \{\spadesuit\}, \{\heartsuit\}) \tag{8}$$

The meaning of an element $(u,v)_t$ in a state of $\mathcal{A}$ is that edge (u,v) is the t-th edge (for $0 \le t < n$) of a simple cycle over V, i.e., of a candidate solution for the TSP. The edge weights of the cycle must also appear, in unary notation: if $(u,v)_t$ is part of a state and $w(u,v) = k$, then also $\diamondsuit_{(u,v),1}, \dots, \diamondsuit_{(u,v),k}$ must be part of the state. Hence, a length-n simple cycle $\mathbf{c} = (v_0, \dots, v_{n-1})$ over V is encoded as a state $T(\mathbf{c}) = E(\mathbf{c}) \cup W(\mathbf{c}) \cup \{\heartsuit\}$, where

$$E(\mathbf{c}) = \{(v_0,v_1)_0, (v_1,v_2)_1, \dots, (v_{n-2},v_{n-1})_{n-2}, (v_{n-1},v_0)_{n-1}\}$$

is the set of edges traversed by the cycle, indexed in order of traversal, and

$$W(\mathbf{c}) = \{\diamondsuit_{(v_t,v_{(t+1) \bmod n}),i} : 0 \le t < n, 1 \le i \le w(v_t, v_{(t+1) \bmod n})\}$$

contains the elements encoding the weights of the edges in $E(\mathbf{c})$.

Moreover, consider a state $T \subseteq S$. If $\heartsuit \notin T$, then necessarily $\mathrm{res}_{\mathcal{A}}(T) = \varnothing$, since all reactions have $\heartsuit$ as a reactant. Similarly, $\spadesuit \in T$ implies $\mathrm{res}_{\mathcal{A}}(T) = \varnothing$, since $\spadesuit$ inhibits all reactions.

Now suppose $\heartsuit \in T$ and $\spadesuit \notin T$. Reactions (1)–(5) produce $\spadesuit$ from T when any of the following conditions (implying that $T \cap E$ does *not* encode a simple cycle over V) occur:

(1) the state T contains two distinct elements denoting edges occurring as the t-th edge of the candidate cycle;
(2) one of the vertices of the candidate cycle has outdegree greater than 1;
(3) one of the vertices of the candidate cycle has indegree greater than 1;
(4) two consecutive edges do not share an endpoint;
(5) no edge is the t-th edge of the candidate cycle, for some $0 \le t < n$.

Any set $T \cap E$ where *none* of the above apply[2] encodes a valid simple cycle over V.

Reaction (6) produces $\spadesuit$ if an edge $(u,v)_t$ occurs, but some element $\diamondsuit_{(u,v),i}$, encoding a unit of the weight of the edge, is missing. Conversely, reaction (7) produces $\spadesuit$ if a unit of the weight of a missing edge occurs. These reactions are all simultaneously disabled exactly when $T \cap W$ contains the weights of the edges in $T \cap E$. Finally, reaction (8) preserves the $\heartsuit$. The result function of $\mathcal{A}$ is thus

$$\mathrm{res}_{\mathcal{A}}(T) = \begin{cases} \{\heartsuit\} & \text{if } T \text{ encodes a weighted simple cycle over } V \\ \{\heartsuit, \spadesuit\} & \text{if } \heartsuit \in T,\ \spadesuit \notin T, \text{ but } T \text{ fails to encode} \\ & \quad \text{a weighted simple cycle over } V \\ \varnothing & \text{if } \heartsuit \notin T \text{ or } \spadesuit \in T \end{cases}$$

[2] Reactions (2) and (3), as well as the corresponding conditions, are redundant when taken together with the remaining ones; however, they are both listed here for clarity.

Hence, the preimages of $\{\heartsuit\}$ are exactly the weighted simple cycles over V. Since each preimage T of $\{\heartsuit\}$ has size $|T| = n + 1 + \sum_{t=0}^{n-1} w(v_t, v_{(t+1) \bmod n})$, a preimage T of $\{\heartsuit\}$ of minimum size corresponds to a shortest tour over V, which can be extracted from T in polynomial time just by listing the elements in $T \cap E$, ordered by their subscript. Since the mapping $(V, w) \mapsto \mathcal{A}$ described by the above construction can be computed in polynomial time, the thesis follows. □

Lemma 5. *For RS* MPP *is metric reducible to the problem of finding, among the possible output strings of a polynomial-time nondeterministic TM, a string having the minimum number of* 1*s.*

Proof. Given any instance $(\mathcal{A} = (S, A), T)$ of the RS minimal preimage problem, let M be the nondeterministic TM which behaves as follows. M guesses a state $U \subseteq S$ and checks whether $\mathrm{res}_{\mathcal{A}}(U) = T$; if this is the case, then M outputs U as a binary string in $\{0,1\}^{|S|}$; M outputs $1^{|S|+1}$, otherwise. Clearly, M works in time $p(n)$ for some polynomial p, and its outputs are all the preimages of T (together with an easily distinguishable dummy output if no preimage exists); in particular, the outputs of M that are minimal with respect to the number of 1s correspond to the smallest preimages of T. □

The following result can be proved similarly to the equivalence of the binary TSP and the problem of finding the maximum binary output of a polynomial-time nondeterministic TM [10].

Lemma 6. *Finding a string with minimal number of* 1*s among those output by a nondeterministic polynomial-time TM is metric reducible to the unary TSP.*

We are now able to provide lower and upper bounds to the complexity of the RS minimal preimage problem.

Theorem 7. MPP *for RS is equivalent to the unary TSP under metric reductions. Hence,* MPP $\in \mathbf{FP}^{\mathbf{NP}}$ *and it is* $\mathbf{FP}_{\|}^{\mathbf{NP}}$*-hard under metric reductions.*

Proof. The equivalence is a consequence of Lemmata 4, 5, and 6. Moreover, TSP belongs to $\mathbf{FP}^{\mathbf{NP}}$ which is closed under metric reductions [10]. The travelling salesman with 0/1 weights is hard for $\mathbf{FP}_{\|}^{\mathbf{NP}}$ (see [1]) and can be reduced to the minimal RS preimage problem. □

The complexity of finding the *size* of minimal preimages (SMPP) is given in the following

Corollary 8. SMPP *is* $\mathbf{FP}^{\mathbf{NP}[\log n]}$*-complete under metric reductions.*

Proof. The size of the minimal preimage can be found by binary search, using an oracle answering the question "Is there a preimage of size at most k?", which belongs to **NP**. Hence, the problem is in $\mathbf{FP}^{\mathbf{NP}[\log n]}$. The hardness of the problem follows from the fact that the size of a minimal preimage is just the length of the shortest unary travelling salesman tour increased by $n + 1$ in the reduction of Lemma 4, and that the unary TSP is $\mathbf{FP}^{\mathbf{NP}[\log n]}$-complete [10]. □

Finally, we can also prove lower and upper bounds to the complexity of finding *the number of* minimal preimages or #MPP in short.

Theorem 9. #MPP *is in* $\#\mathbf{P}^{\mathbf{NP}[1]}$ *and it is* $\#\mathbf{P}$*-hard under parsimonious reductions.*

Proof. The following algorithm shows the membership in $\#\mathbf{P}^{\mathbf{NP}[1]}$: given $(\mathcal{A}, T)$, guess a state U, then check if $\mathrm{res}_{\mathcal{A}}(U) = T$. If so, make a single query $(\mathcal{A}, T, |U|)$ to an oracle for the following language

$$L = \{(\mathcal{A}, T, k) : \text{ there exists } V \text{ such that } \mathrm{res}_{\mathcal{A}}(V) = T \text{ and } |V| < k\}$$

and accept if and only if the oracle gave a negative answer. The number of accepting computations thus coincides with the number of minimal preimages of T. Furthermore, we have $L \in \mathbf{NP}$, since a polynomial-time nondeterministic TM can guess a state of $\mathcal{A}$ and check if it is a preimage of T with less than k elements.

In order to prove the $\#\mathbf{P}$-hardness of the problem, we perform a reduction from #SAT (a variant of [7, Theorem 4]). Let $\varphi = \varphi_1 \wedge \cdots \wedge \varphi_m$ be a Boolean formula in conjunctive normal form over the variables $V =$

$\{x_1, \ldots, x_n\}$. Let $\mathcal{A} = (S, A)$ be a RS with $S = V \cup \overline{V} \cup C \cup \{\spadesuit\}$, where $\overline{V} = \{\bar{x}_1, \ldots, \bar{x}_n\}$ and $C = \{\varphi_1, \ldots, \varphi_m\}$, and A consisting of the following reactions:

$$(\{x_i\}, \{\spadesuit\}, \{\varphi_j\}) \qquad \text{for } 1 \leq i \leq n,\ 1 \leq j \leq m, \text{ if } x_i \text{ occurs in } \varphi_j \tag{9}$$

$$(\{\bar{x}_i\}, \{\spadesuit\}, \{\varphi_j\}) \qquad \text{for } 1 \leq i \leq n,\ 1 \leq j \leq m, \text{ if } \bar{x}_i \text{ occurs in } \varphi_j \tag{10}$$

$$(\{s\}, \{x_i, \bar{x}_i, \spadesuit\}, \{\spadesuit\}) \qquad \text{for } 1 \leq i \leq n,\ s \notin \{x_i, \bar{x}_i, \spadesuit\} \tag{11}$$

$$(\{x_i, \bar{x}_i\}, \{\spadesuit\}, \{\spadesuit\}) \qquad \text{for } 1 \leq i \leq n. \tag{12}$$

A state $T \subseteq S$ encodes a valid assignment for φ if, for each $1 \leq i \leq n$, it contains either x_i or $\bar{x}_i$ (denoting the truth value of variable x_i), but not both, and no further element. The reactions of type (9) (resp., (10)) produce the set of elements representing the clauses satisfied when x_i is assigned a true (resp., false) value. Hence, the formula φ has as many satisfying truth assignments as the number of states $T \subseteq S$ encoding valid assignment such that $\mathrm{res}_{\mathcal{A}}(T) = C$, the whole set of clauses. Any such T contains exactly n elements.

If a state T has n elements or less, but it is not a valid assignment to φ, then there is at least one literal x_i or $\bar{x}_i$ missing in T: thus, either $T = \varnothing$, or reaction (11) is enabled and $\spadesuit \in \mathrm{res}_{\mathcal{A}}(T)$; in both cases $\mathrm{res}_{\mathcal{A}}(T) \neq C$. If T has strictly more than n elements, then either it is an inconsistent assignment, containing both x_i and $\bar{x}_i$ for some i, and in that case $\spadesuit \in \mathrm{res}_{\mathcal{A}}(T) \neq C$ by reaction (12), or it has a subset $T' = T \cap (V \cup \overline{V})$ with $|T'| = n$ such that $\mathrm{res}_{\mathcal{A}}(T') = \mathrm{res}_{\mathcal{A}}(T)$.

Thus, the number of *minimal* preimages of C in $\mathcal{A}$ is exactly the number of assignments satisfying φ, as required. □

This proof can be adapted to show that counting the number of preimages, without requiring minimality, is #**P**-complete.

Corollary 10. #PP *is* #**P**-*complete under parsimonious reductions.*

Proof. The problem is in #**P**, since a polynomial-time nondeterministic TM can guess a state and then check efficiently if it is a preimage [7]. The problem is #**P**-hard under parsimonious reductions because the minimal preimages of the state C from the proof of Theorem 9 are as many as the satisfying assignments of φ. □

4. Ancestor Problems

We now turn our attention to the ancestor problems, which show a (supposedly) higher complexity. First of all, we need a few technical results, providing us with a **FPSPACE**(poly)-complete function suitable for reductions.

Lemma 11. *The "universal" function $U(M, 1^m, x)$, defined as $M(x)$ if the TM M halts in space m, and undefined otherwise, is* **FPSPACE**(poly)*-complete under many-one reductions.*

Proof. We have $U \in$ **FPSPACE**(poly), since there exist universal TMs having only a polynomial space overhead [10]. Let $R \in$ **FPSPACE**(poly), and let M be a polynomial-space TM for R. Then, there exists a polynomial p bounding both the working space and the output length of M. The mapping $f(x) = (M, 1^{p(|x|)}, x)$ can be computed in polynomial time, and $R(x, U(f(x)))$ holds for all $x \in \mathrm{dom}\, R$. This proves the **FPSPACE**(poly)-hardness of U. □

Lemma 12. *Let the binary relation $R((M, 1^m, y), x)$ hold if and only if the deterministic TM M, on input x, halts with output y on its tape, $|y| \leq m$, and M does not exceed space m during the computation. Then, the relation R is* **FPSPACE**(poly)*-complete under many-one reductions.*

Proof. The relation R is in **FPSPACE**(poly): a polynomial-space deterministic TM can try all strings x of length at most m, one by one, and simulate M on input x (within space m) until the output y is produced or the strings have been exhausted. We now reduce the function U from Lemma 11 to R. Given an instance $(M, 1^m, x)$ of U, consider the instance $(M', 1^k, 1)$ of R, where

- M' is the TM which on any input y, first simulates M on input x within space k; if $M(x) = y$, then M' outputs 1; otherwise, M' outputs 0.

- $k = p(m)$, where p is the polynomial space overhead needed by M' in order to simulate M.

We have $U(M, 1^m, x) = y$ if and only if M' outputs 1 in space k on input y, that is, if and only if $R((M', 1^k, 1), y)$. Since the mapping $(M, 1^m, x) \mapsto (M', 1^k, 1)$ can be computed in polynomial time, the relation R is **FPSPACE**(poly)-hard. □

By exploiting the ability of RS to simulate polynomial-space TMs [6], we obtain the following result.

Theorem 13. MAP *is complete for* **FPSPACE**(poly) *under metric reductions.*

Proof. The problem is in **FPSPACE**(poly), since a polynomial-space TM can enumerate all states of a RS in order of size, and check whether they lead to the target state (the reachability problem for RS is known to be in **PSPACE** [6]).

In order to prove the **FPSPACE**(poly)-hardness of the problem, we describe a variant of the simulation of polynomial-space TMs by means of RS from [6], where a distinguished state T, encoding the final configuration of the TM, has as minimal ancestors all states encoding the inputs of the TM leading to that configuration. Given a TM M and a space bound m, let Q and Γ be the set of states and the tape alphabet of M, including a symbol $\sqcup$ for "blank" cells. We define a RS $\mathcal{A}$ having background set

$$S = \{q_i : q \in Q, -1 \leq i \leq m\} \cup \{a_i : a \in \Gamma, 0 \leq i < m\} \cup \{\spadesuit\}.$$

We encode the configurations of M as states of $\mathcal{A}$ as follows: if M is in state q, the tape contains the string $w = w_0 \cdots w_{m-1}$, and the tape head is located on the i-th cell, then the corresponding state of $\mathcal{A}$ is $\{q_i, w_{0,0}, \ldots, w_{m-1,m-1}\}$, with an element q_i representing TM state q and head position i, and m elements corresponding to the symbols on the tape indexed by their position.

A transition $\delta(q, a) = (r, b, d)$ of M is implemented by the following reactions

$$(\{q_i, a_i\}, \{\spadesuit\}, \{r_{i+d}, b_i\}) \qquad \text{for } 0 \leq i < m$$

which update state, head position, and symbol under the tape head. The remaining symbols on the tape (which have an index different from the tape head position) are instead preserved by the following reactions:

$$(\{a_i\}, \{q_i : q \in Q\} \cup \{\spadesuit\}, \{a_i\}) \qquad \text{for } a \in \Gamma,\ 0 \leq i < m$$

If an element encoding TM state and position is not part of the current RS state, the element representing the initial state $s \in Q$ of M, with tape head on the first cell, is produced by the following reactions

$$(\{a_i\}, \{q_j : q \in Q, 0 \leq j < m\} \cup \{\spadesuit\}, \{s_0\}) \qquad \text{for } a \in \Gamma,\ 0 \leq i < m \tag{13}$$

and the simulation of M by $\mathcal{A}$ begins in the next time step with the same tape contents. If M exceeds its space bound, by moving the tape head to the left of position 0 or to the right of position m, the following reactions become enabled

$$(\{q_{-1}\}, \{\spadesuit\}, \{\spadesuit\}) \qquad \text{for } q \in Q$$
$$(\{q_m\}, \{\spadesuit\}, \{\spadesuit\}) \qquad \text{for } q \in Q$$

and produce the universal inhibitor $\spadesuit$, which halts the simulation in the next time step. The universal inhibitor is also produced by the following reactions when the state of $\mathcal{A}$ is not a valid encoding of a configuration of M, namely, when multiple state elements appear:

$$(\{q_i, r_j\}, \{\spadesuit\}, \{\spadesuit\}) \qquad \text{for } q, r \in Q,\ q \neq r,\ 0 \leq i < m,\ 0 \leq j < m$$

or when multiple symbols are located on the same tape cell:

$$(\{a_i, b_i\}, \{\spadesuit\}, \{\spadesuit\}) \qquad \text{for } a, b \in \Gamma,\ a \neq b,\ 0 \leq i < m$$

or when a tape cell does not contain any symbol (recall that a blank cell contains a specified symbol from Γ):

$$(\{s\}, \{a_i : a \in \Gamma\} \cup \{\spadesuit\}, \{\spadesuit\}) \qquad \text{for } s \notin \{a_i : a \in \Gamma\} \cup \{\spadesuit\},\ 0 \le i < m \tag{14}$$

The result function of $\mathcal{A}$ can thus be described as follows:

$$\mathrm{res}_{\mathcal{A}}(T) = \begin{cases} T' & \text{if } T \text{ encodes a configuration of } M \text{ and } T' \text{ its next configuration} \\ T \cup \{s_0\} & \text{if } T \text{ encodes a tape of } M \\ T' \cup \{\spadesuit\} & \text{for some } T' \subseteq S, \text{ if } \spadesuit \notin T \text{ but } T \text{ does not encode a configuration of } M \\ \varnothing & \text{if } \spadesuit \in T \text{ or } T = \varnothing \end{cases}$$

Notice that all states of $\mathcal{A}$ encoding configurations of M have $m+1$ elements, and that either $\mathrm{res}_{\mathcal{A}}(T) = \varnothing$ or $\spadesuit \in \mathrm{res}_{\mathcal{A}}(T)$ if $|T| < m$.

Given an instance $(M, 1^m, y)$ of relation R from Lemma 12, we can ask for a minimal state X of $\mathcal{A}$ leading to Y, where Y encodes the configuration of M in its final state, with the string y on the tape.

If there exists a string x such that $M(x) = y$ and the space bound m is never exceeded by M during its computation, then there exists a state $X \subseteq S$ encoding a tape for M containing the input x (padded to length m with blanks) and such that $\mathrm{res}^t_{\mathcal{A}}(X) = Y$ for some $t \ge 0$. We have $|X| = m$, and X is minimal with respect to size among all states leading to Y, since all smaller states lead to $\varnothing$ in at most two steps. Furthermore, from X we can easily recover a string x with $M(x) = y$. Conversely, any state $X \subseteq S$ with $|X| = m$ and $\mathrm{res}^t_{\mathcal{A}}(X) = Y$ necessarily encodes a string x such that $M(x) = y$ within space m.

If $M(x) \neq y$ for all strings x (or all such computations exceed the space bound m), then all states T such that $\mathrm{res}^t_{\mathcal{A}}(T) = Y$ for some $t \ge 0$, and in particular the minimal ones, contain $m+1$ elements; indeed, given a state of size m encoding a tape, the RS attempts to simulate M on this tape, starting with the initial state, and by hypothesis this can never result in outputting y. By observing this fact, we can infer that no input of M produces the output y in space m.

Since the mapping $(M, 1^m, y) \mapsto (\mathcal{A}, Y)$ described by the above construction can be computed in polynomial time, and the answer for R can be extracted from the answer to the minimal RS ancestor search problem in polynomial time, the latter problem is **FPSPACE**(poly)-hard under metric reductions. □

We now deal with the problem of finding the size of a minimal ancestor.

Lemma 14. *Let $f(M, 1^m) = \min\{|x| : \text{the TM } M \text{ accepts } x \text{ in space } m\}$, or undefined if no such x exists. Then f is* **FPSPACE**(log)*-complete under many-one reductions.*

Proof. Given $g \in$ **FPSPACE**(log), let G be a deterministic TM computing g in space $p(n)$ for some polynomial p, and let $x \in \Sigma^\star$. Let M be a deterministic TM that, on input y, first simulates $G(x)$, then accepts if and only if $|y| \ge G(x)$. Hence, we have $g(x) = f(M, 1^{q(|x|)})$, where q is a polynomial bound on the space needed by M to simulate G. This proves the hardness of f. The function can be computed in **FPSPACE**(log) by simulating M on all strings of length at most m until one is accepted in space m, then outputting its length. □

Theorem 15. SMAP *is* **FPSPACE**(log)*-complete under metric reductions.*

Proof. The problem is in **FPSPACE**(log), since a polynomial-space TM can find an ancestor U of a state of a RS as in the proof of Theorem 13, by outputting the size of U rather than U itself. In order to show the hardness of the problem, we reduce the function f from Lemma 14 to it. Given $(M, 1^m)$, let $\mathcal{A}$ be the RS of Theorem 13, simulating M in space 1^m, and let $\mathcal{A}'$ be $\mathcal{A}$ modified as follows. First of all, we add $\heartsuit$ to S, and we also add it as a reactant to all reactions. Then, the reactions of type (13) are replaced by

$$(\{\heartsuit\}, \{a_i : a \in \Gamma\} \cup \{q_j : q \in Q, -1 \le j \le m\} \cup \{\spadesuit\}, \{\sqcup_i\}) \qquad \text{for } 0 \le i < m \tag{15}$$

$$(\{\heartsuit\}, \{q_j : q \in Q, -1 \le j \le m\} \cup \{\spadesuit\}, \{s_0\}) \tag{16}$$

The reactions of type (15) complete the tape of the TM by producing a blank symbol in position i if no symbol a_i and no state q_j occur. Reaction (16) produces the initial state of M in position 0 when no other state element occurs.

The reactions of type (14) are replaced by

$$(\{q_j\}, \{a_i : a \in \Gamma\} \cup \{\spadesuit\}, \{\spadesuit\}) \qquad \text{for } q \in Q,\ 0 \le i < m,\ -1 \le j \le m \tag{17}$$

which give an error (producing $\spadesuit$) when a tape symbol is missing, but only if a state element is already present. Finally, we add the reaction $(\{\heartsuit\}, \{\spadesuit\}, \{\heartsuit\})$, which preserves the $\heartsuit$ element.

The behaviour of $\mathcal{A}'$ differs from $\mathcal{A}$ in the following ways. The input string $x = x_0 \cdots x_{n-1}$ of M is provided as a state $X = \{x_{0,0}, \ldots, x_{n-1,n-1}, \heartsuit\}$. In the first step of $\mathcal{A}'$, the tape is completed by adding blanks and the initial state of M (reactions (15)–(16)); this produces the state $X \cup \{\sqcup_n, \ldots, \sqcup_{m-1}, s_0\}$, which encodes the initial configuration of M. The simulation of M then proceeds as for $\mathcal{A}$ (with the additional element $\heartsuit$ always present).

The ancestors of state T describing an accepting configuration of M (empty tape, accepting state, head in position 0) and of minimal size encode the initial input x of M together with $\heartsuit$, if M accepts at least one string in space m (hence, such ancestors have size at most $m+1$); if no string is accepted, the minimal ancestors of T all have size at least $m+2$, by the same reasoning as in the proof of Theorem 13. Hence, $f(M, 1^m)+1$ is the size of a minimal ancestor of T, if the latter is at most $m+1$, and $f(M, 1^m)$ is undefined otherwise: this defines a metric reduction of f to this problem. □

Remark 16. The problem of Theorem 15 is actually complete under metric reductions that only increase linearly the length of the output; the class **FPSPACE**(log) is closed under such reductions, but not under general metric reductions.

Finally, we show that counting the number of minimal ancestors has the same complexity as finding one of them.

Lemma 17. *Given a TM M, a unary integer 1^m, and a string y, computing the number of strings x of length at most m such that $M(x) = y$ in space m is* **FPSPACE**(poly)*-complete under many-one reductions.*

Proof. Recalling that **FPSPACE**(poly) = ♮**PSPACE**, the following nondeterministic polynomial-space TM has the required number of accepting computations: on input $(M, 1^m, y)$, it guesses a string x of length at most m (this requires polynomially many guesses); then it simulates M on x, accepting if M outputs y without exceeding space m, and rejecting otherwise.

Given $f \in$ ♮**PSPACE**, let N be a nondeterministic, polynomial-space TM with $f(x)$ accepting computations on input x; let $p(n)$ be both a space bound for N and bound on the number of nondeterministic choices it makes. Consider the following polynomial-space TM M: on input $z \in \{0,1\}^\star$, it simulates a computation of N on input x, but replaces the nondeterministic choices of N with deterministic lookups to successive bits of z. If N exceeds space $m = p(|x|)$, or halts without having made exactly $|z|$ nondeterministic choices, or the simulated computation of N rejects, then M writes 0 as output; otherwise, M outputs 1.

Hence, M outputs 1 once for each accepting computation of N, that is, for exactly $f(x)$ input strings. Since the mapping $x \mapsto (M, 1^{p(|x|)}, 1)$ can be computed in polynomial time, the **FPSPACE**(poly)-hardness of the problem follows. □

Theorem 18. #MAP *is* **FPSPACE**(poly)*-complete under metric reductions.*

Proof. The problem is in **FPSPACE**(poly), since a polynomial-space TM can compute the size of a minimal ancestor of a state T of a RS, then enumerate all states of the same size and count how many of them lead to T.

Let $(M, 1^m, y)$ be an instance of the problem of Lemma 17, and let $\mathcal{A}$ be the RS simulating M as in the proof of Theorem 13. Let $Y = \{f_0, z_{0,0}, \ldots, z_{m-1,m-1}\}$ be the state of $\mathcal{A}$ encoding the final configuration of M with output y, where $z = y_0 \cdots y_{k-1} \sqcup^{m-k}$ is y padded to length m with blanks and $f \in Q$ is the final state of M. In order to distinguish between the presence or absence of at least a string x such that $M(x) = y$, we add a large number of ancestors of Y having size $m+1$, ensuring that they are minimal only if no such string exists. Let $\mathcal{A}'$ be $\mathcal{A}$ augmented with the following reactions:

$$(\{a_i, \spadesuit\}, \{q_j : q \in Q, -1 \le j \le m\}, \{z_{i,i}, f_0\}) \qquad \text{for } a \in \Gamma,\ 0 \le i < m. \tag{18}$$

When ♠ is present, and all q_j are missing, these reactions map each element representing a symbol in tape cell i, to the symbol z_i in tape cell i, together with the final state f of M in position 0. In particular, when at least one symbol per position i is present, the whole target state Y is produced. Hence, these reactions introduce exactly $|\Gamma|^m$ new ancestors of Y of size $m+1$. The state Y then has at least $|\Gamma|^m+1$ ancestors of size $m+1$, including Y itself. On the other hand, from the proof of Theorem 13 we may infer that the maximum number of ancestors of Y of size m is $|\Gamma|^m$.

The number of strings x of length at most m such that $M(x)=y$ in space m is then equal to the number of minimal ancestors of Y for $\mathcal{A}'$, if and only if this number is at most $|\Gamma|^m$. If the number is larger than $|\Gamma|^m$, then the minimal ancestors have size $m+1$, denoting the fact that no such string x exists. This defines a metric reduction, proving the **FPSPACE**(poly)-hardness of #MAP. □

When removing the requirement of minimality the problem remains complete for **FPSPACE**(poly). However, we were only able to prove completeness using Cook reductions, which are weaker than the metric ones.

Theorem 19. *#AP is* **FPSPACE**(poly)*-complete under Cook reductions.*

Proof. Let $f(\mathcal{A},T)$ be the number of ancestors of state T for the RS $\mathcal{A}$. Then, we have $f\in$ **FPSPACE**(poly), since a polynomial-space TM can simulate $2^{|S|}$ computation steps of $\mathcal{A}$ from any possible initial state $U\subseteq S$, and count how many initial states eventually reach T.

Consider now the decision problem of establishing whether a given state T is part of a global attractor cycle in $\mathcal{A}$, that is, if all initial states $U\subseteq S$ eventually reach T. This problems is known to be **PSPACE**-complete[3]. It is easy to see that T is eventually reached by all initial states if and only if $f(\mathcal{A},T)=2^{|S|}$. This proves that f is **PSPACE**-hard under metric reductions.

If $g\in$ **FPSPACE**(poly), then the problem of computing the i-th bit of $g(x)$ is in **PSPACE**: the corresponding machine computes $g(x)$, but then only outputs its i-th bit. Hence, any **FPSPACE**(poly) function can be computed bit-by-bit in polynomial time using a polynomial number of queries to a **PSPACE** oracle. This suffices to prove the thesis. □

It is still an open problem to prove that counting the number of ancestors is **FPSPACE**(poly)-complete under metric or even parsimonious reductions. On the other hand, when looking at the forward dynamics, the problem can be shown to be **FPSPACE**(poly)-complete using only metric reductions.

Lemma 20. *Computing the length of the computation of a TM within a given space bound is* **FPSPACE**(poly)*-hard under metric reductions.*

Proof. Consider the **FPSPACE**(poly)-complete problem of counting the number of inputs of length n accepted in space m by a TM M. We can assume that all computations of M on inputs of length n consist of the same number of steps, and that this number has the form $2^{p(n)}$ for some polynomial p. Given $(M,1^n,1^m)$, let M' be the TM which enumerates all strings of length n and simulates M on each of them in turn, while keeping a counter of the number of accepted strings; finally, M' decrements the final value c of the counter down to zero one unit at a time. It turns out that M' operates in time $f(n)+g(n)\times c$, where $f(n)$ is the time required to simulate M on all strings of length n and update the counter, and $g(n)$ is the time required to decrement the counter. All updates of the counter (including leaving it unchanged when a string is rejected by M) can be performed in time $g(n)$, independent of the actual value of the counter. Since $f(n)$ and $g(n)$ can be easily computed from n, the value of c, i.e., the number of strings accepted by M, can be recovered from the number of steps performed by M' before halting. This proves that computing the length of the computation of a TM within a given space bound is **FPSPACE**(poly)-hard under metric reductions. □

Theorem 21. *#DP is* **FPSPACE**(poly)*-complete under metric reductions.*

[3] This result was proved in [6, Corollary 5] for RS allowing reactions with empty reaction or inhibitor sets. However, the proof can be easily adapted to RS with nonempty sets.

Proof. The problem is in **FPSPACE**(poly), since, given a RS $\mathcal{A} = (S, A)$ and a state $T \subseteq S$, a polynomial-space TM can check, for each state $U \subseteq S$, if U is reachable from T and return a count of such states. The problem of Lemma 20 can be parsimoniously reduced to counting the number of distinct descendants of a state T of an RS $\mathcal{A}$ by having $\mathcal{A}$ simulate a polynomial-space TM as in Theorem 13 and letting T correspond the initial configuration of the TM. Indeed, each configuration of the simulated TM corresponds to a descendant state of T. □

5. Gardens of Eden

Here we investigate two problems concerning gardens of Eden. While it is relatively easy to show the existence of them, actually producing one or counting them seems to require the presence of a **NP** oracle.

Theorem 22. GEP *is* $\mathbf{FNP}^{\mathbf{NP}}$*-complete under metric reductions.*

Proof. Given a RS $\mathcal{A} = (S, A)$, consider the following polynomial-time nondeterministic TM M with an **NP** oracle: on input $\mathcal{A}$, M guesses a state $T \subseteq S$ and asks the oracle whether T has a preimage. If it does not, then M outputs T as a garden of Eden; otherwise M rejects. This proves that the problem is in $\mathbf{FNP}^{\mathbf{NP}}$.

Consider now the $\mathbf{FNP}^{\mathbf{NP}}$-complete problem of finding an assignment to the variables from a set X such that, for all assignments to the variables from a set Y, the DNF formula φ over $X \cup Y$ is satisfied. Let $\varphi = \bigvee C$ with $C = \{\psi_1, \dots, \psi_n\}$. Build a RS having $S = X \cup Y \cup \{\diamondsuit, \heartsuit, \clubsuit\}$ such that a set of the form $V \cup \{\clubsuit\}$, with $V \subseteq X$, is a garden of Eden if and only if the assignment of the variables in X represented by V satisfies φ, irrespective of the assignment of the variables from Y.

A state of $\mathcal{A}$ of the form $T = \{\diamondsuit\} \cup V$, with $V \subseteq X \cup Y$, encodes a truth assignment $\nu \colon X \cup Y \to \{0, 1\}$ where $\nu(z) = 1$ if and only if $z \in T$. The formula φ is evaluated under ν by the reactions

$$(\{\diamondsuit\} \cup \mathrm{pos}(\psi), \{\heartsuit, \clubsuit\} \cup \mathrm{neg}(\psi), \{\heartsuit\}) \qquad \text{for all } \psi \in C \tag{19}$$

where $\mathrm{pos}(\psi)$ is the set of variables appearing in positive form in ψ, and $\mathrm{neg}(\psi)$ is the set of negated variables of ψ. These reactions produce $\heartsuit$ if and only if the assignment encoded by T satisfies φ.

The variables in $T \cap X$, i.e., the partial assignment to X, is preserved by the reactions

$$(\{\diamondsuit, x\}, \{\heartsuit, \clubsuit\}, \{x\}) \qquad \text{for all } x \in X \tag{20}$$

while the variables in $T \cap Y$ are discarded. Finally, the $\diamondsuit$ is converted into $\clubsuit$:

$$(\{\diamondsuit\}, \{\heartsuit, \clubsuit\}, \{\clubsuit\}) \tag{21}$$

The result function of $\mathcal{A}$ is thus

$$\mathrm{res}_{\mathcal{A}}(T) = \begin{cases} (T \cap X) \cup \{\heartsuit, \clubsuit\} & \text{if } T \text{ encodes a satisfying assignment for } \varphi \\ (T \cap X) \cup \{\clubsuit\} & \text{if } T \text{ encodes an assignment not satisfying } \varphi \\ \varnothing & \text{if } T \text{ does not encode an assignment} \end{cases}$$

A state of the form $T = \{\clubsuit\} \cup V$ with $V \subseteq X$ has a preimage if and only the latter has the form $\{\diamondsuit\} \cup V \cup W$ with $W \subseteq Y$, that is, if not all extensions to Y of the assignment encoded by T satisfy φ. Conversely, T is a garden of Eden if and only if all extensions to Y satisfy φ.

Therefore, a deterministic TM M can map φ to $\mathcal{A}$ in polynomial time, then ask an oracle to find a garden of Eden T for $\mathcal{A}$. If such T exists, the TM M can check whether it has the required form $T = \{\clubsuit\} \cup V$ with $V \subseteq X$; if this is the case, the corresponding assignment to X is output. In any other case, the TM M rejects. This proves that finding a garden of Eden is $\mathbf{FNP}^{\mathbf{NP}}$-hard under metric reductions. □

Corollary 23. *Given a RS* $\mathcal{A} = (S, A)$ *and a state* $T \subseteq S$*, deciding whether* T *is a garden of Eden is* **coNP***-complete.*

Proof. The problem is in **coNP**, since a polynomial time nondeterministic TM can guess a state $U \subseteq S$, reject if it is a preimage of T, and accept otherwise. This machine has only accepting computations if and only if T is a garden of Eden.

The SAT variant in the proof of Theorem 22 becomes ∀SAT (also known as VALIDITY) if we let $X = \emptyset$, and this is a **coNP**-complete problem [10]. Since we can reduce it in polynomial time to asking whether $T = \{\clubsuit\}$ is a garden of Eden for the RS $\mathcal{A}$ of the proof of Theorem 22, and this state has preimages if and only if the formula φ is not a tautology, the **coNP**-hardness of the problem follows. □

Theorem 24. *#GEP is #**P**$^{\mathbf{NP}}$-complete under metric reductions.*

Proof. Given a RS $\mathcal{A} = (S, A)$, consider the following nondeterministic polynomial-time Turing machine M with an **NP** oracle. M guesses a state $T \subseteq S$, asks the oracle whether T has a preimage, and accepts if and only if it does not. The number of accepting computations of M is thus the number of gardens of Eden of $\mathcal{A}$. This proves that the problem is in **#P**$^{\mathbf{NP}}$.

Let #∀SAT be the following counting problem: given a formula of the form $\#X \forall Y \varphi$, where X and Y are finite sets of variables and φ a Boolean formula in disjunctive normal form, count the number of assignments of the variables in X such that each extended assignment to $X \cup Y$ satisfies φ. This problem is known to be **#P**$^{\mathbf{coNP}}$-complete and, equivalently, **#P**$^{\mathbf{NP}}$-complete under metric reductions [13].

Given a formula $\#X \forall Y \varphi$, build the RS $\mathcal{B} = (S, A \cup A')$, where (S, A) is the RS from the proof of Theorem 22, and A' contains the reactions

$$(\{\clubsuit\}, \{\heartsuit, \diamondsuit\} \cup Y, \{\heartsuit, \clubsuit\}) \tag{22}$$

$$(\{\clubsuit, x\}, \{\heartsuit, \diamondsuit\} \cup Y, \{x\}) \qquad \text{for all } x \in X \tag{23}$$

which map any state of the form $\{\clubsuit\} \cup V$, with $V \subseteq X$, into $\{\heartsuit, \clubsuit\} \cup V$, ensuring that no state of the latter form is a garden of Eden.

Therefore, the states of the RS $\mathcal{B}$ that are *not* gardens of Eden are $\emptyset$, the states of the form $\{\heartsuit, \clubsuit\} \cup V$ with $V \subseteq X$, and the states of the form $\{\clubsuit\} \cup V$ with $V \subseteq X$ representing assignments that have an extension to $X \cup Y$ not satisfying φ. The number of such states is

$$1 + 2^{|X|} + \#X \neg \forall Y \varphi \qquad = \qquad 1 + 2^{|X|} + 2^{|X|} - \#X \forall Y \varphi$$

and, as a consequence, the number of gardens of Eden is

$$2^{|S|} - 1 - 2^{|X|} - 2^{|X|} + \#X \forall Y \varphi.$$

The value of $\#X \forall Y \varphi$ can be easily recovered from the number of gardens of Eden of $\mathcal{B}$, and this shows the **#P**$^{\mathbf{NP}}$-completeness under metric reductions of counting them. □

Remark 25. The existence of at least one garden of Eden is guaranteed for RS having reactions with nonempty reactant and inhibitor sets, since the result function is never injective ($\mathrm{res}_{\mathcal{A}}(S) = \mathrm{res}_{\mathcal{A}}(\emptyset) = \emptyset$), hence never surjective. However, it is interesting to notice that the problem becomes **NP**-complete when empty reactants or inhibitors are allowed, as establishing whether the result function is surjective is **coNP**-complete in that case [6]. Even then, the problem is easier than expected, since actually outputting a garden of Eden apparently requires an **NP** oracle in addition to the use of a nondeterminstic TM.

This is related to the fact that the witnesses for the existence of a garden of Eden that we are able to exhibit consists of a pair of states having the same image under $\mathrm{res}_{\mathcal{A}}$; this kind of witness does not seem to allow us to find the actual garden of Eden in deterministic polynomial time. The garden of Eden itself does not seem to be a valid witness, since we do not know whether it can be checked in polynomial time (Corollary 23).

6. Conclusions

We investigated the problem of finding the minimal preimage of a state of a RS and proved that this problem is equivalent, under metric reductions, to finding a minimal TSP tour when the weights are expressed in unary.

We also studied the complexity of finding a minimal ancestor of a given state and showed that it is as hard as simulating a polynomial-space TM (with polynomial-length output). We have investigated the complexity of other problems related to preimages (resp., ancestors): finding the size and the number of minimal preimages (resp., ancestors). When removing the requirement of minimality, counting the number of ancestor of descendants does not decrease in complexity. Finally, we have studied problems related to gardens of Eden, showing that an **NP** oracle is required for finding one or counting them. All the problems investigated were proved to be intractable, with complexity ranging from $\mathbf{FP}^{\mathbf{NP}[\log n]}$ to **FPSPACE**(poly).

In the future we plan to continue the exploration of problems related to preimages and ancestors of RS. In the more general model [5], RSs behave as interactive processes, where new entities are introduced at every time step by means of a context sequence. Under which conditions does the presence of a context sequence increase the complexity of the problems we considered? We are also interested in questions concerning the approximability of the aforementioned problems and the complexity of finding a minimal ancestor that is not "too far" from the target state. Finally, a set of potentially interesting variants can be obtained by considering the minimality of ancestors and descendants with respect to set inclusion rather than set cardinality.

Acknowledgements

This work has been supported 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 di reazioni", by the Italian MIUR PRIN 2010-2011 grant "Automata and Formal Languages: Mathematical and Applicative Aspects" H41J12000190001, and by the French National Research Agency project EMC (ANR-09-BLAN-0164).

Bibliography

[1] Z.Z. Chen, S. Toda, On the complexity of computing optimal solutions, International Journal of Foundations of Computer Science 2 (1991) 207–220. URL: http://dx.doi.org/10.1142/S0129054191000133.

[2] A. Dennunzio, E. Formenti, L. Manzoni, Reaction systems and extremal combinatorics properties, Theoretical Computer Science (2015). To appear.

[3] A. Dennunzio, E. Formenti, L. Manzoni, A.E. Porreca, Preimage problems for reaction systems, in: A.H. Dediu, E. Formenti, C. Martín-Vide, B. Truthe (Eds.), Language and Automata Theory and Applications, 9th International Conference, LATA 2015, volume 8977 of *Lecture Notes in Computer Science*, Springer, 2015, pp. 537–548. URL: http://dx.doi.org/10.1007/978-3-319-15579-1_42.

[4] A. Ehrenfeucht, G. Rozenberg, Basic notions of reaction systems, in: C.S. Calude, E. Calude, M.J. Dinneen (Eds.), Developments in Language Theory, 8th International Conference, DLT 2004, volume 3340 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 27–29. URL: http://dx.doi.org/10.1007/978-3-540-30550-7_3.

[5] A. Ehrenfeucht, G. Rozenberg, Reaction systems, Fundamenta Informaticae 75 (2007) 263–280. URL: http://iospress.metapress.com/content/b86t11hryvwq6910/.

[6] E. Formenti, L. Manzoni, A.E. Porreca, Cycles and global attractors of reaction systems, in: H. Jürgensen, J. Karhumäki, A. Okhotin (Eds.), Descriptional Complexity of Formal Systems, 16th International Workshop, DCFS 2014, volume 8614 of *Lecture Notes in Computer Science*, Springer, 2014, pp. 114–125. URL: http://dx.doi.org/10.1007/978-3-319-09704-6_11.

[7] E. Formenti, L. Manzoni, A.E. Porreca, Fixed points and attractors of reaction systems, in: A. Beckmann, E. Csuhaj-Varjú, K. Meer (Eds.), Language, Life, Limits, 10th Conference on Computability in Europe, CiE 2014, volume 8493 of *Lecture Notes in Computer Science*, Springer, 2014, pp. 194–203. URL: http://dx.doi.org/10.1007/978-3-319-08019-2_20.

[8] M.W. Krentel, The complexity of optimization problems, Journal of Computer and System Sciences 36 (1988) 490–509. URL: http://dx.doi.org/10.1016/0022-0000(88)90039-6.

[9] R.E. Ladner, Polynomial space counting problems, SIAM Journal on Computing 18 (1989) 1087–1097. URL: http://dx.doi.org/10.1137/0218073.

[10] C.H. Papadimitriou, Computational Complexity, Addison-Wesley, 1993.

[11] A.E. Porreca, N. Murphy, M.J. Pérez-Jiménez, An optimal frontier of the efficiency of tissue P systems with cell division, in: M. García-Quismondo, L.F. Macías-Ramos, Gh. Păun, L. Valencia-Cabrera (Eds.), Tenth Brainstorming Week on Membrane Computing, volume II, Fénix Editora, 2012, pp. 141–166. URL: http://www.gcn.us.es/icdmc2012_proceedings.

[12] A. Salomaa, Minimal and almost minimal reaction systems, Natural Computing 12 (2013) 369–376. URL: http//dx.doi.org/10.1007/s11047-013-9372-y.

[13] E. Zawadzki, A. Platzer, G.J. Gordon, A generalization of SAT and #SAT for robust policy evaluation, in: F. Rossi (Ed.), Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI 13), AAAI Press, 2013, pp. 2583–2589. URL: http://ijcai.org/papers13/contents.php.