PREDICTORS FOR FLAT MEMBRANE SYSTEMS

Roberto Barbuti[a], Roberta Gori[a], Paolo Milazzo[a]

[a]*Dipartimento di Informatica, Università di Pisa*
Largo Pontecorvo 3, 56127 Pisa, Italy

Abstract

In this paper we investigate dynamic causalities in membrane systems by proposing the concept of "predictor", originally defined in the context of the reaction systems by Brijder, Ehrenfeucht and Rozenberg. The goal is to characterise sufficient and necessary conditions for the presence of a multiset of molecules of interest in the configuration of a P system at a given evolution step (independently from the non-deterministic choices taken). These conditions can be used to study causal relationships between molecules and, therefore, to predict some aspects of future development of multiset rewriting systems.

To achieve this goal, we introduce the new concept of "multiset pattern" representing a logical formula on multisets. A *sufficient predictor* can be expressed as a pattern characterising initial multisets that *will surely evolve*, after the given number of evolution steps, into a multiset containing the molecules of interest. On the other hand, a *necessary predictor* models initial multisets that *may evolve* after the given number of evolution steps, into a multiset containing the molecules of interest. Necessary predictors can be used to characterise initial multisets that *will surely not evolve* (in the required number of steps) into a multiset that contains such molecules. We inductively define operators able to compute these predictors.

The patterns obtained from our operators are sound (sufficient or necessary) predictors, but, in general, they are not complete.

Keywords: Membrane Systems, Dynamic causalities, Reaction Systems, Predictors.

1. Introduction

The understanding of causal relationships among the events happening in a biological (or bio-inspired) system is an issue widely investigated in the context of both systems biology (see e.g. [1, 2, 3]) and natural computing (see e.g. [4]).

In [5] Brijder, Ehrenfeucht and Rozenberg initiate a study on *causalities* in reaction systems [6, 7]. Causalities deal with the ways entities of a reaction system influence each other. In [5], both static/structural causalities and dynamic causalities are discussed, introducing the idea of *predictors*. A predictor can be used to determine whether a molecule of interest s will be produced after k execution steps of the reaction system, without executing the system itself.

Email addresses: `barbuti@di.unipi.it` (Roberto Barbuti), `gori@di.unipi.it` (Roberta Gori), `milazzo@di.unipi.it` (Paolo Milazzo)

In reaction systems the environment is the only source of non-determinism. Knowledge about the molecules which will be provided at each step by the environment is required to determine whether a molecule s will be produced after k steps. Not all molecules are relevant for the production of s. On the basis of these two observations, a predictor is defined as the subset of molecules Q whose supply by the environment need to be observed in order to determine whether s will be produced or not after k steps.

In [8, 9, 10], the idea of predictors was enhanced by defining the notion of *formula based predictor*. A formula based predictor consists in a propositional logic formula to be satisfied by the sequence of sets of molecules that the environment provides to the reaction system. This logic formula clearly discriminates between the cases in which a particular molecule s will be produced after a given number of steps and the cases in which it will not.

P systems [11, 12] are much more powerful and complex than reaction systems. They are based on multisets rather than sets and evolution rules are applied with *maximal parallelism* and with a non-deterministic competition for reactants.

The computational behaviour of a P system is determined only by the initial multiset and by the non-deterministic choices made at each maximally parallel step. In this context, a notion of predictor may correspond to a logical formula to be satisfied by the initial multiset (representing either a *sufficient condition* or a *necessary condition*) for the molecules of interest to be present after a given number of evolution steps. Due to the intrinsically non-deterministic nature of P systems, sufficient and necessary conditions have to be dealt with separately.

Example 1.1. *For an intuition, consider the following rewriting rules:*

$$r_0 : \; A \to C \qquad r_1 : \; A \to B \qquad r_2 : \; D \to C$$

forming a maximally parallel multiset rewriting system. Assume we are interested in the presence of molecule C after one step of rewriting. If we want to be sure that molecule C will be present after one step we have two possibilities: either C or D have to belong to the initial multiset. Having molecule A in the initial multiset is not sufficient to ensure that molecule C will be produced after one evolution step. Indeed, if rule r_1 is applied the initial multiset does not evolve in one step into a multiset containing molecule C. Therefore, if we want to devise sufficient conditions for the presence of molecule C after one step, we need to characterise in some way all initial multisets containing molecule C or D. On the other hand, if a multiset after one evolution step (from the initial one) contains molecule C then we have three possible cases, either the initial multiset already contained molecule C or the initial multiset contained molecule A or molecule D. Hence, if we are interested in the necessary condition for the presence of molecule C after one step, we have to characterise all the initial multiset containing molecule A or C or D.

The previous example shows that, in general, sufficient and necessary conditions do not coincide, and therefore they have to be handled separately. Note that sufficient conditions can be used to determine the multiset of molecules that will *surely* cause the presence of the molecule of interest in the required number of evolution steps, while necessary conditions can be used to determine, by complementation, the multisets that *surely cannot cause* the presence of a molecule of interest in the required number of evolution steps. Therefore, investigating causal dependencies in the previous example, we can say that the presence of C or D in the initial multiset causes the presence of C after one evolution step but the presence of C after one step cannot be caused by an initial multiset that does not contain neither C nor A nor D.

The previous example also shows that while it is quite easy to deal with necessary conditions, dealing with sufficient condition is a complex task because competition on reactants comes into play.

In this paper we propose the notion of *multiset pattern* as a way to express logical formulas on molecules of multisets. Multiset patterns will be used to characterise sufficient and necessary conditions for a molecule (or, in general, for a multiset of molecules) to be present after a given number of steps. We first focus on the pattern that expresses sufficient conditions. If the initial multiset of the P systems satisfies (*matches*) such a pattern, then the multiset of molecules of interest will be *surely* present after k steps; nothing can be said otherwise (it is indeed a sufficient condition). Such pattern will be defined by a recursive operator. We will show that the pattern obtained from the operator will be a sound, but, in general, not a complete predictor. This means that there can be multisets that do not match the pattern, but that still always lead to the presence of the molecules of interest in k steps. However, there might exist special classes of P system for which the sufficient predictor is also complete. In this paper, we investigate the completeness of our sufficient predictor for P systems with non-cooperative rules.

We then focus on the pattern expressing necessary conditions for the presence of the molecules of interest. In this case we have that if the P system leads to a multiset that contains the molecules of interest in the required number of steps then the initial multiset *matches* the pattern expressing necessary conditions. This information can be used to characterise multisets that surely will not evolve in a multiset containing the molecules of interest in the required number of steps. We provide a recursively defined operator that computes such pattern. Once again, we will show that the pattern we compute will be a sound, but, in general, not complete. This means that there can be multisets that do model the pattern, but will not lead to the presence of the molecules of interest in k steps. However, we prove that for P systems with non-cooperative rules our necessary predictor is also complete.

Once defined, patterns expressing sufficient and necessary conditions can be used to discover non trivial causal relationships among molecules.

Related works. Causality properties have been investigated in different contexts. For example, causalities are studied in the context of concurrency theory, systems biology and natural computing. Different notions of causality have been considered such as dependencies between events or between reachable states of the considered system. Moreover, different techniques have been applied to verify causality properties on system models such as static analysis, enhanced semantics and other formal "ad hoc" techniques.

In [13, 14] formal methods for studying causality properties of concurrent systems are proposed. Concurrent systems are described by means of process algebras such as CSS or π-calculus. The proposed approaches consist in the definition of enriched semantics that allow modeling causal dependencies between events (process synchronization) and states (bindings of channel names) of the system. This is obtained by extending the semantics with information about the past execution of the considered processes.

A similar approach is followed by Busi in [15] in the context of systems biology, and in [4] in the context of membrane computing. As in the previous cases, the idea is to enrich the semantics of the modelling formalism. In particular, in Busi's approach, the semantics is extended with an explicit information about the causal dependencies between reactions.

In the context of systems biology, several approaches are based on static analysis [16, 17, 1, 2, 3]. In order to make the verification of causality properties feasible, these approaches compute an approximation of the system behaviour. Typically, the behaviour of the system is over-approximated, that is, all the possible system executions are modelled, but also some executions that are not possible are included in the model.

Methods for the analysis of systems that are based on the computation of backward evolution steps can be found in the context of *reversible systems*. Reversibility is the ability of a system to go back to the initial state and to reconstruct the trace of evolution steps leading to the final state. Reversibility has been studied in the context of multiset rewriting and membrane systems for instance in [18, 19] and [20].

In [18, 19] reversibility is defined as a relation between reachable configurations, that is dual to the notion of determinism. A system is deterministic if every configuration can evolve into a unique new configuration. Dually, a system is reversible if every configuration can be obtained by a single previous configuration. This leads the authors to study the classes of reversible and deterministic systems.

In [20] the author proposed *membrane systems with memory*, an extension of membrane systems in which objects are labeled with information about the rules that produced them in order to be able to reverse the computations. Similarly, *Dual P Systems* are proposed in [21] with the same aim. This is obtained by replacing the rules of the P system with *dual* rules that allow the computations to be executed backwards.

The idea of backward execution of a P system is somehow related to the idea of identifying the causes for the presence of some molecules after a fixed number of steps. However, in order to predict whether some molecules will be present or not, reversing a single computation is not enough. We show that all possible computations have to be considered.

This article is a revised and extended version of the paper in [22], endowed with the definition of the new concept and results on necessary predictor applied to several biological examples. More precisely, the new contributions of this paper with respect to [22] are:

- the definition of the concept of necessary predictor characterizing the necessary conditions for the presence of a multiset after a fixed number of evolution steps;
- the definition of an operator able to compute a such predictor;
- the study of a particular class of programs for which our operators on multiset patterns are also complete;
- the application of predictors to several biological examples to study causal dependencies;
- the inclusion of extended definitions, results and proofs.

The paper is organised as follows. Section 2 introduces some notions on multisets and the standard definition of membrane systems. The new notion of multiset patterns together with a discussion on their expressive power can be found in Section 3. In Section 4 we introduce, through examples of incremental complexity the notion of sufficient predictor, an operator to compute it and we state and prove all its properties. Section 5 contains the notion of necessary predictor, the definition of the operator that computes it and all its properties. Section 6 presents applications of predictors to language acceptors and to several biological systems related to gametogenesis, genetics and gene regulation network. Finally, our conclusions can be found in Section 7.

2. Preliminaries

2.1. Multisets

We denote by $\mathbb{N}$ the set of natural numbers. Let U be an arbitrary set. A *multiset* (over U) is a mapping $M : U \to \mathbb{N}$; where by $|M|_a$, for $a \in U$, we denote the *multiplicity* of a in the multiset M. The *support* of a multiset M is the set $supp(M) = \{a \mid |M|_a > 0\}$. A multiset is empty when its support is empty and it is denoted by $\emptyset$.

In order to distinguish multiset operations from standard set operations we use the following notations: $\subseteq^*$ for multiset inclusion, $\cup^*$ for multiset union, $\cap^*$ for multiset intersection. For notational convenience, we often denote multisets by strings. Given a finite set of symbols V, V^* represents the set of all multisets over V while V^+ represents V^* without the empty multiset $\emptyset$. For a set X, we denote the power set of X by $\wp(X)$ while $\wp_f(X)$ indicate all the finite sets in $\wp(X)$.

2.2. Membrane Systems

In this paper we consider *flat* P systems, namely in which the membrane structure consists only of the skin membrane. Flat P systems are defined as follows.

Definition 2.1. *A* flat P system *is a construct* $\Pi = (V, w, R)$ *where:*

- V *is a finite set of symbols, called* alphabet*, whose elements are called* molecules species*;*
- $w \in V^*$ *is the* initial multiset *of molecules;*
- R *is a finite set of* $(V^*)^2$*; the element of R are called* evolution rules *of* Π .

From [23] it follows that a P system with a standard membrane structure can be translated into an equivalent P system having a (flat) membrane structure that consists only of the skin.

In this paper we assume P systems to be closed computational devices, namely we assume that molecules cannot be sent out of the skin membrane (i.e. rules sending molecules out are not allowed in the skin membrane) and molecules cannot be received by the skin membrane from outside. `react` As a consequence, evolution rules will have the simple form $u \to v$ with $u, v \in V^+$. When $u \in V$ the rule is called non-cooperative [12].

We denote with $\mathcal{R}_V$ the set of all evolution rules on V.

Given an evolution rule $r = u \to v$, we denote with $\texttt{react}(r)$ and $\texttt{prod}(r)$ its multisets of reactants u and products v, respectively. The same notations extend naturally to finite multisets of rules: for any finite multiset of rules $M \in \mathcal{R}_V^*$, we have $\mathtt{react}(M) = \bigcup^*_{r \in \mathsf{supp}(M)} \mathtt{react}(r)^{|M|_r}$ and $\mathtt{prod}(M) = \bigcup^*_{r \in \mathsf{supp}(M)} \mathtt{prod}(r)^{|M|_r}$.

We assume evolution rules to be applied with standard maximal parallelism. Since we consider P systems which do not send/receive molecules to/from the external environment, we obtain that the behaviour of a P system is determined only by its initial multiset and by the non-deterministic choices made at each maximally parallel step.

3. Multiset Patterns

3.1. Definitions

Given an alphabet V, a multiset pattern expresses a condition on multisets in V^*. A *basic* multiset pattern is a pair $(u, \{u_1, ..., u_n\})$ where $u, u_1, ..., u_n$ are finite multisets in V^*. More complex patterns can be obtained by composing basic patterns by using propositional logic connectives $\wedge$ and $\vee$. The syntax of multiset patterns is defined as follows.

Definition 3.1 (Multiset Patterns)**.** *Given an alphabet V, $\mathcal{P}_V$ is the set of multiset patterns on V^* inductively defined as follows:*

- $\mathtt{true}, \mathtt{false} \in \mathcal{P}_V$,
- *if $p \in (V^* \times \wp_f(V^*))$ then $p \in \mathcal{P}_V$,*
- *if $p_1, p_2 \in \mathcal{P}_V$ then $p_1 \vee p_2$ and $p_1 \wedge p_2 \in \mathcal{P}_V$.*

When the reference alphabet V is clear from the context, we will denote the set $\mathcal{P}_V$ simply as $\mathcal{P}$.

Now, we formally define the notion of satisfaction of a pattern by a multiset, i.e. the semantics of multiset patterns. The idea is that a multiset w satisfies a basic pattern $(u, \{u_1, ..., u_n\})$ if by removing from w the multisets $u_1, ..., u_n$ in any maximal way (i.e. so that what remains does not include any of $u_1, ..., u_n$) we always obtain a multiset which includes u. For example, multiset A^4B^2 satisfies the pattern $(A, \{AB\})$ since after removing AB a maximal number of times (two times) we obtain A^2 which includes A.

The semantics of multiset patterns is formally defined as follows.

Definition 3.2. *Given an alphabet V, the* satisfaction relation $\models \; \subseteq V^* \times P_V$ *is the relation inductively defined as follows:*

$$\begin{array}{ll}
w \models \mathtt{true} & \\
w \models (u, \{u_1, ..., u_n\}) & \textit{iff } \forall o_1, ...o_n \in \mathbb{N} \textit{ such that } w \sqsupseteq^* u_1^{o_1} u_2^{o_2} \dots u_n^{o_n}, \textit{ it holds } w \sqsupseteq^* u u_1^{o_1} u_2^{o_2} \dots u_n^{o_n} \\
w \models p_1 \wedge p_2 & \textit{iff } w \models p_1 \textit{ and } w \models p_2 \\
w \models p_1 \vee p_2 & \textit{iff either } w \models p_1 \textit{ or } w \models p_2
\end{array}$$

For the case of a basic pattern $(u, \{u_1, ..., u_n\})$, Definition 3.2 includes a requirement on all possible values $o_1, ..., o_n \in \mathbb{N}$ used as multiplicities of the occurrences of $u_1, ..., u_n$ in w. It is easy to see that the requirement can be checked by considering only the maximal combinations of values $o_1, ..., o_n$ such that $w \sqsupseteq^* u_1^{o_1} u_2^{o_2} \dots u_n^{o_n}$.

3.2. Multiset Patterns and Multiset Languages

Multiset patterns can be used to express complex conditions on multisets.

Given a pattern p, we define the *satisfaction language* L_p of as

$$L_p = \{w \in V^* \mid w \models p\}.$$

The pattern $p_1 = (AB, \{BC, BE\})$ is satisfied by multisets that contain at least one A and one B, and where the sum of the numbers of C and of E is smaller than the number of B. The set of multisets satisfying the pattern corresponds to the multiset language

$$L_{p_1} = \{w \in V^* \mid |w|_A > 0, |w|_B > |w|_C + |w|_E\}.$$

Patterns can express also conditions that are not in the "greater than" form. For example, pattern $p_2 = (A, \{AA\})$ is satisfied by multisets with an odd number of A. Namely, it characterizes the language

$$L_{p_2} = \{w \in V^* \mid |w|_A \equiv_{mod\,2} 1\}.$$

More generally, given any $n \in \mathbb{N}$, the multiset pattern $(A, \{A^n\})$ is such that $w \models (A, \{A^n\})$ iff w satisfies $|w|_A \not\equiv_{mod\,n} 0$.

The examples we have given show that multiset patterns could be used to characterize multiset languages. It could be interesting to investigate the classes of languages characterised by multiset patterns. Although such an investigation is out of the scope of this paper, we discuss few more examples of interesting languages.

Let us look for a pattern characterizing the language consisting only of the A^3 multiset. The pattern $p_3 = (A^3, \{\})$ is not correct, since it is satisfied by any multiset with *at least* three instances of A. We could combine p_3 with a pattern satisfied by multiset containing *at most* three instances of A. However, such pattern can not be directly defined but it can be obtained only by negating the pattern $(A^4, \{\})$. Hence, characterizing the language consisting only of the A^3 multiset would be possible only by including logical negation in the syntax of patterns.

A few more examples: $(A, \{AB\})$ characterizes the language $A^m B^n$ with $m > n$. Consequently, $(A, \{AB\}) \vee (B, \{AB\})$ characterizes the language $A^m B^n$ with $m \neq n$. It seems not possible to define a pattern that characterizes the complement of the previous language, namely $A^n B^n$. However, this would be possible by including logical negation in the syntax of patterns.

3.3. Simplification of Multiset Patterns

Multiset patterns express logical conditions on multisets, hence they can be simplified using standard logic rules. For instance, a conjunction of basic patterns can be simplified to `false` every time the basic patterns implicitly express opposite constraints. Moreover, the following properties of the satisfaction relation $\models$ allow us to consider further simplification rules for multiset patterns. The proof of the two properties follows immediately from Definition 3.2.

Lemma 3.1. *Let* $(w, \{u_1, ..., u_n\})$ *be a basic multiset pattern and* $v \in V^*$. *Then* $v \models (w, \{u_1, ..., u_n\})$ *iff* $v \models (w, \{u_i \mid u_i \cap^* w \neq \emptyset\})$.

Using the previous result a basic pattern $(w, \{u_1, ..., u_n\})$ can be always simplified into $(w, \{u_i \mid u_i \cap^* w \neq \emptyset\})$.

Lemma 3.2. *Let $(w, \{u_1, ..., u_n\})$ be a basic pattern. If there exists $i \in \{1, ..., n\}$ such that $u_i \subseteq^* w$, then for all $v \in V^*$ it holds $v \not\models (w, \{u_1, ..., u_n\})$.*

Using the previous result a basic pattern $(w, \{u_1, ..., u_n\})$ such that $u_i \subseteq^* w$ for some $i \in \{1, ..., n\}$ can be simplified into `false`.

3.4. Multiset Patterns for expressing Sufficient and Necessary Conditions

We propose a methodology to compute a sufficient and necessary conditions (expressed as multiset patterns) for the presence of a multiset u after k evolution steps in a given P system. Both conditions will be expressed as patterns to be satisfied by the initial multiset w of the P system.

The idea is to define two different operators computing sufficient and necessary predictors by starting from the pattern $(u, \{\})$ and by rewriting it considering the rules of the P system under investigation.

Since establishing sufficient conditions is the most difficult task, in the next section we start by defining sufficient predictors.

Necessary predictors will be studied in Section 5.

4. Sufficient Predictors

Here we propose a methodology based on multiset patterns to compute sufficient conditions for the presence of a multiset u after k evolution steps of a given P system. The sufficient condition will be expressed as a pattern (called *sufficient predictor*) to be satisfied by the initial multiset w of the P system.

The idea is to define an operator that computes the predictor by starting from the pattern $(u, \{\})$ and rewriting it by taking into account the set of rules of the P system. The pattern will be rewritten k times, each time simulating (in an abstract way) a backward step of the P system evolution. At each step, for each rule that is assumed to be applied, the result will include information of the rules competing with the selected rule for application.

The definition of the operator that computes a predictor for a multiset u in k steps is quite complex. We start with the definition of few auxiliary functions and sets. Then, we choose to introduce the concepts by giving several examples of incremental complexity. Examples will be alternated with definitions of functions that formalise the introduced concepts. The complete definition of the operator, together with the related theoretical results and proofs will conclude the section (see Section 4.6).

4.1. Auxiliary Functions and Sets

Function `AppRules` gives the set of all possible minimal multisets of rules that lead to the production of the finite multiset w.

Definition 4.1. *Given an alphabet V, we define the function* `AppRules` $: V^* \times \wp_f(\mathcal{R}_V) \to \wp_f(\mathcal{R}_V^*)$ *as*

$$\texttt{AppRules}(w, R) = \{M \in R^* \mid w \subseteq^* \textstyle\bigcup^*_{r \in \mathsf{supp}(M)} \texttt{prod}(r)^{|M|_r}, \forall M' \subset^* M : w \not\subseteq^* \bigcup^*_{r \in \mathsf{supp}(M')} \texttt{prod}(r)^{|M'|_r}\}$$

for all finite $w \in V^*$, $R \in \wp_f(\mathcal{R}_V)$.

Example 4.1. *Consider the P system* $\Pi_0 = (\{A, B, C, D, E\}, w_0^0, R_0)$ *where evolution rules of the set* R_0 *are:*

$$r_0 : \; AB \to C \qquad r_2 : \; C \to AC \qquad r_1 : \; BD \to C \qquad r_3 : \; E \to A$$

We have $\texttt{AppRules}(CCA, R_0) = \{r_0r_0r_3,\ r_0r_1r_3,\ r_1r_1r_3,\ r_0r_2,\ r_1r_2,\ r_2r_2\}$.

In order to simulate a backward step of the P system evolution, consider that a molecule might be obtained as the product of an applied evolution rule, but it also might be obtained because it was already present and no rule used it. Of course, this is not possible if there is a rule in the P system having such a molecule as the only reactant. As a consequence, in order to simulate the backward step of a P system, we consider an extended set of evolution rules that includes also *self rules* rewriting each molecule into itself, for each molecule that is not the only reactant of a rule of the P system.

Definition 4.2. *Given an alphabet V, and a finite set of rules* $R \subseteq \mathcal{R}_V$*, the set of* self rules $Self_R$ *is defined as*

$$Self_R = \{v \to v \mid v \in V \text{ and } \forall r \in R,\ \texttt{react}(r) \neq v\}.$$

Example 4.2. *For the P system* Π_0 *of Example 4.1, we obtain*

$$Self_{R_0} = \{r_4 : A \to A,\ r_5 : B \to B,\ r_6 : D \to D\}.$$

While the self rule $A \to A$ *indicate that A can be left unchanged after one evolution step of the P system* Π_0*, the self rule* $C \to C$ *cannot be introduced, because molecule C can never be left unchanged by an evolution step of* Π_0*, since if a multiset includes molecule C then rule* r_2 *must be applied. The same holds for the self rule* $E \to E$.

4.2. Competition for Reactants

Example 4.3. *Consider the P system* $\Pi_1 = (\{A, B, C, D\}, w_0^1, R_1)$ *where the evolution rules* R_1 *are:*

$$r_0 : \; AB \to D \qquad r_1 : \; BD \to C$$

A visual representation of evolution rules in $R_1 \cup Self_{R_1}$ *is given by the graph in Fig 1. The nodes in the top of the graph represent molecules used as reactants (associated with index* 1*), while the nodes in the bottom of the graph represent products (associated with index* 2*). Reactions are represented as nodes in the middle of the graph and by the arcs connecting such nodes to reactants and products. Solid arcs represent the evolution rules in* R_1*, while dashed arcs the evolution rules in* $Self_{R_1} = \{r_2, r_3, r_4, r_5\}$.

The graph representation helps us to reason on backward steps of the P system. For instance, in order to obtain D in one step, we need to have either A and B, or D itself in the previous step. Moreover, the graph makes explicit the competition of evolution rules on common reactants. For example, the production of D by rule r_0 *competes with the application of rule* r_1 *since both rules have B as a reactant. Similarly, the self rule* $r_5 : D \to D$ *competes with rule* r_1*. Note, however, that the contrary does not hold. Indeed rule* r_1 *does not compete with the self rule* $r_5 : D \to D$ *because self rules represent molecules which are not consumed by the evolutions steps of* Π_1.

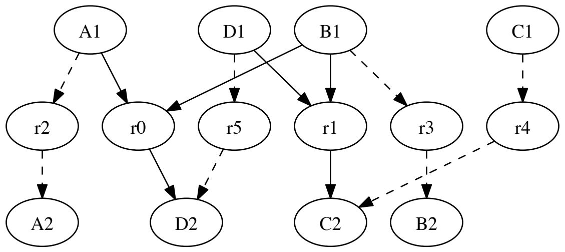

Figure 1: Rules $R_1 \cup Self_{R_1}$

The information on competition between evolution rules is essential. Indeed, in order to be sure that D is present after one step we need to be sure that either r_0 has been applied or D was already present and it has not been consumed by any other evolution rule not producing D.

We now define the function $\mathtt{competitor}_1$, which results in the set of evolution rules competing for reactants with a given evolution rule r which produces a molecule of interest s.

Definition 4.3. *Given an alphabet V, we define*

$$\mathtt{competitor}_1(r, R, s) = \{r' \in R \mid \mathtt{react}(r) \cap^* \mathtt{react}(r') \neq \emptyset \text{ and } s \notin \mathtt{prod}(r')\},$$

for any rule $r \in \mathcal{R}_V$, any finite set of rules $R \subseteq \mathcal{R}_V$ and any molecule $s \in V \cap \mathtt{supp}(\mathtt{prod}(r))$.

A pattern that characterises a sufficient condition for the presence of D after one step can be easily obtained by combining the reactants of evolution rules producing D (including self rules) with the information on the reactants of the other evolution rules that compete with them and do not produce the molecule D.

For the case of Example 4.3, we can express a pattern that characterises a sufficient condition for the production of D in one step as follows

$$\bigvee_{r \in \mathtt{AppRules}(D, R_1 \cup Self_{R_1})} (\mathtt{react}(r), \mathtt{react}(\mathtt{competitor}_1(r, R_1, D))$$

that corresponds to $(AB, \{BD\}) \vee (D, \{BD\})$. This pattern shows that D can be produced in two ways: through AB, that are the reactants of r_0, or through D itself. In both cases the only competitor is r_1, whose reactants are BD. So in both cases the pattern requires that AB or D remains after removing instances of BD in a maximal way. In other words, the pattern expresses the condition that either the multiset includes AB and the instances of B are more than the instances of D, or there is at least one D and the instances of D are more than the instances of B. Examples of multisets that satisfy the pattern (leading to the production of D) are ABB, AD, $ABBD$, etc. If w_0^1 satisfies such pattern then we can be sure that molecule D will be present after one step.

Note that rule r_2 is not considered as a competitor since it is a self rule. Such kind of rules are not considered to compete with other rules since they simply represent molecules that are not used by actual evolution rules of the P system.

In order to perform more than one backward step we will have to generalise the computation of the pattern representing the sufficient condition for the presence of a single molecule to the case of *a multiset of molecules*. For instance, in the case of Example 4.3, performing one more backward step will require to compute the sufficient condition for the production of AB or of D, that will then be used to obtain D.

In order to show how to compute a pattern for the presence of a *multiset of molecules* after one step, consider, in the case of Example 4.3, the multiset DC. The pattern representing a sufficient condition for the presence of DC in one step could be obtained by combining the already seen pattern for the presence of D with analogous pattern for the presence of C, that is $(BD, \{AB\}) \vee (C, \{\})$. Since D and C are two different molecules (the case of repeated occurrences of the same molecule is more complex and will be treated separately in Section 4.3) we can combine the two patterns by simply using conjunction of patterns, thus obtaining:

$$((AB, \{BD\}) \vee (D, \{BD\})) \wedge ((BD, \{AB\}) \vee (C, \{\}))\,.$$

Multisets satisfying the pattern are DC, ABC, ADC, $ABBD$, etc. Indeed, such multisets allow us to obtain DC after one step according to the rules in R_1. On the other hand, multisets not satisfying the pattern are for instance ABD, DCB and $ABDC$. The latter, in particular, could lead to the production of DC (actually DDC), but also to the production of ACC that does not include DC.

Example 4.4. *Consider now a P system $\Pi_2 = (\{A, B, C, D\}, w_0^2, R_2)$ where R_2 (depicted in Fig 2) is quite similar to R_1 of Example 4.3 with r_0 extended with one more product, namely*

$$r_0 : \quad AB \to DC \qquad\qquad r_1 : \quad BD \to C$$

While the sufficient conditions for molecule D to be present after one step are the same as in the previous example, the condition for the presence of C after one step has to take into account that now r_0 produces C, therefore it does not compete with r_1 for the production of C. This is correctly taken into account by the function $\texttt{competitor}_1$*, indeed* $\texttt{competitor}_1(r_0, R_2, C) = \emptyset$*. Therefore, the pattern for the presence of C in one step, defined as*

$$\bigvee_{r \in \texttt{AppRules}(C, R_2 \cup Self_{R_2})} (\texttt{react}(r), \texttt{react}(\texttt{competitor}_1(r, R_2, C)))\,,$$

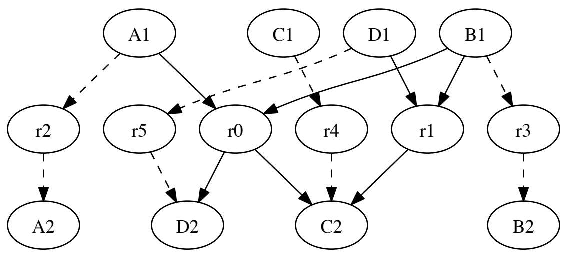

Figure 2: Rules $R_2 \cup Self_{R_2}$

turns out to be $(AB, \{\}) \vee (BD, \{\}) \vee (C, \{\})$.

4.3. Competitors Dealing with Multiple Occurrences of Molecules

When multiple occurrences of the same molecule come into the picture, things get quickly more complicated.

Example 4.5. *Consider the P system* $\Pi_3 = (\{A, B, C, D\}, w_0^3, R_3)$ *where the evolution rules* $R_3 = \{r_0, r_1\}$*, depicted in Fig. 3, are*

$$r_0 : \quad AB \rightarrow D \qquad\qquad r_1 : \quad BC \rightarrow D$$

Assume we are interested in the multiset DD. To produce DD in the P system Π_3 *we may either apply one rule twice, or the two rules together. The pattern for the presence of DD in one step cannot be obtained just as a conjunction of a pattern p expressing the sufficient condition for D with itself, since* $p \wedge p$ *is equivalent to p.*

In order to deal with multiple occurrences of molecules we have to consider, in the computation of the backward step, the possible multisets of evolution rules *that could have been applied in order to produce such molecules. These multisets of rules are given by the auxiliary function* `AppRules` *defined in Section 4.1.*

At a first glance one may think of defining the pattern for the presence of DD in one step as follows:

$$\bigvee_{n \in \mathtt{AppRules}(DD, R_3 \cup Self_{R_3})} \ \bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \mathtt{react}(\mathtt{competitor}_1(r, R_3, D))),$$

that would give the following result:

$$(ABAB, \{\}) \ \vee \ ((AB, \{\}) \wedge (BC, \{\})) \ \vee \ ((AB, \{\}) \wedge (D, \{\})) \ \vee \ ((BC, \{\}) \wedge (D, \{\})) \ \vee \ (BCBC, \{\}) \\ \vee \ (DD, \{\}))$$

This pattern is however not correct, since it is satisfied by multiset ABC (because $ABC \models (AB, \{\}) \wedge (BC, \{\})$*) that does not lead to DD in one step.*

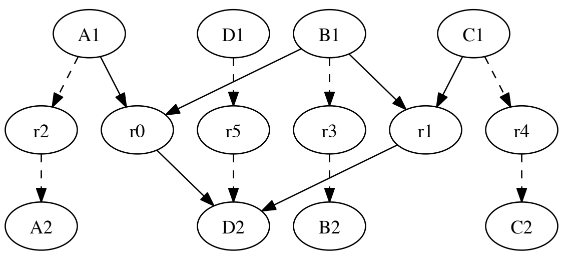

Figure 3: Rules $R_3 \cup Self_{R_3}$

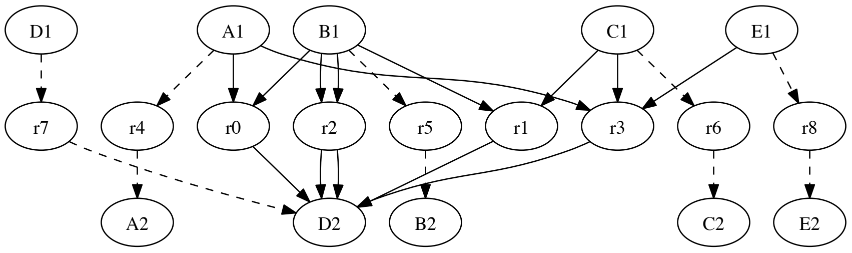

Figure 4: Rules $R_4 \cup Self_{R_4}$

The point in this case is that there are two different rules that produce the same product D competing for the same reactant B. Since more than one instance of D has to be produced, we have to take also this form of competition into account. To this purpose, we define the function $\texttt{competitor}_2$.

Definition 4.4. *Given an alphabet V, we define*

$$\texttt{competitor}_2(r, R, n) = \{r' \in R \mid r' \in \texttt{supp}(n), r' \neq r, \texttt{react}(r) \cap^* \texttt{react}(r') \neq \emptyset\}$$

for any rule $r \in \mathcal{R}_V$, any finite set of rules $R \subseteq \mathcal{R}_V$ and any finite multiset $n \in R^*$.

Now, the pattern for DD can be expressed as

$$\bigvee_{n \in \texttt{AppRules}(DD, R_3 \cup Self_{R_3})} \bigwedge_{r \in \texttt{supp}(n)} (\texttt{react}(r)^{|n|_r}, \texttt{react}(C_{12}))$$

where $C_{12} = \texttt{competitor}_1(r, R_3, D) \cup \texttt{competitor}_2(r, R_3, n)$. The formula gives the following result:

$$(ABAB\{\}) \vee ((AB, \{BC\}) \wedge (BC, \{AB\})) \vee ((AB, \{\}) \wedge (D, \{\})) \vee ((BC, \{\}) \wedge (D, \{\})) \vee (BCBC, \{\}) \\ \vee\ (DD, \{\})$$

which now correctly models the required property.

4.4. Competition for Products

Example 4.6. *Consider the P system $\Pi_4 = (\{A, B, C, D, E\}, w_0^4, R_4)$ where the evolution rules $R_4 = \{r_0, r_1, r_2, r_3\}$, depicted in Fig. 4, are*

$$r_0 : \ AB \to D \qquad r_1 : \ BC \to D \qquad r_2 : \ BB \to DD \qquad r_3 : \ ACE \to D$$

Assume that, as in Example 4.5, we are interested in the presence of multiset DD after one step. The multiset DD can be produced by several combinations of rules in R_4. Rule r_2 has DD as product, but suffers from the competition of r_0 and of r_1 that, although producing the same kind of *molecule, produce only one instance of such a molecule. Indeed, by starting from multisets ABB or BBC, we may obtain DD through r_2, but we may also obtain only one D, through r_0 or r_1, respectively.*

Similarly, there are cases in which DD can be obtained by applying r_0 and r_1 together. Rule r_3, however, may compete with such a combination of rules, since in presence of E it may consume reactants necessary for the application of r_0 and r_1 giving only one D as a result.

This example suggests that the concept of competitor has to be enriched with a definition that takes into account when a rule competes with a multiset of rules $n \in R^*$ that produce more than one occurrence of a molecule. Intuitively, this occurs when the use of such a rule prevents the application of a subset of the rules in n without producing an equivalent number of occurrences of the required molecule. This form of competition is formalised by the function $\mathtt{competitor}_3$ defined as follows.

Definition 4.5. *Let $r \in \mathcal{R}_V$, $R \subseteq \mathcal{R}_V$, $n \in R^*$ and $s \in V$, we define:*

$$\begin{array}{ll} \multicolumn{2}{l}{\mathtt{competitor}_3(r, R, n, s) =} \\ \{r' \in R \mid & s \in \mathtt{prod}(r'), r' \notin \mathtt{supp}(n), \mathtt{react}(r) \cap \mathtt{react}(r') \neq \emptyset, \\ & \exists m \subseteq^* n, \{r\} \subseteq^* m, \mathtt{react}(m) \cap_* \mathtt{react}(r') = \mathtt{react}(n) \cap_* \mathtt{react}(r'), \\ & \forall m' \subset^* m, \mathtt{react}(m') \cap_* \mathtt{react}(r') \neq \mathtt{react}(n) \cap_* \mathtt{react}(r'), |\mathtt{prod}(m)|_s \supset^* |\mathtt{prod}(r')|_s\}. \end{array}$$

Assume as before that we are interested in the production of DD in one step. The pattern expressing a sufficient conditions is then

$$\bigvee_{n \in \mathtt{AppRules}(DD, R_4 \cup Self_{R_4})} \bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \mathtt{react}(C_{123}))$$

where $C_{123} = \mathtt{competitor}_1(r, R_4, D) \cup \mathtt{competitor}_2(r, R_4, n) \cup \mathtt{competitor}_3(r, R_4, n, D)$. The formula gives the following result:

$$\begin{array}{l} ((AB, \{BC, ACE\}) \wedge (BC, \{AB, ACE\})) \vee (BB, \{AB, BC\}) \vee ((AB, \{ACE, BC\}) \wedge (ACE, \{AB, BC\})) \\ \vee ((BC, \{ACE, AB\}) \wedge (ACE, \{BC, AB\})) \vee (DD, \{\}) \vee (ABAB, \{\}) \vee (BCBC, \{\}) \vee (ACEACE, \{\}) \end{array}$$

In the obtained pattern, conjunction $(AB, \{ACE, BC\}) \wedge (ACE, \{AB, BC\})$ is not satisfiable, since on the one hand it requires $ABACE$ to be included in the multiset, but at the same time it requires BC not to be included. The same holds for $(BC, \{ACE, AB\}) \wedge (ACE, \{BC, AB\})$ with $BCACE$ and AB. As a consequence, the pattern can be simplified into

$$\begin{array}{r} ((AB, \{BC, ACE\}) \wedge (BC, \{AB, ACE\})) \vee (BB, \{AB, BC\}) \vee (DD, \{\}) \vee (ABAB, \{\}) \vee (BCBC, \{\}) \\ \vee (ACEACE, \{\}) \end{array}$$

According to this pattern, neither multiset ABB nor BBC satisfy it.

This example shows the situation in which the pattern does not describe *all* multisets that actually lead to the presence of DD in one step. Indeed, the multiset $ABBCE$ leads to the presence

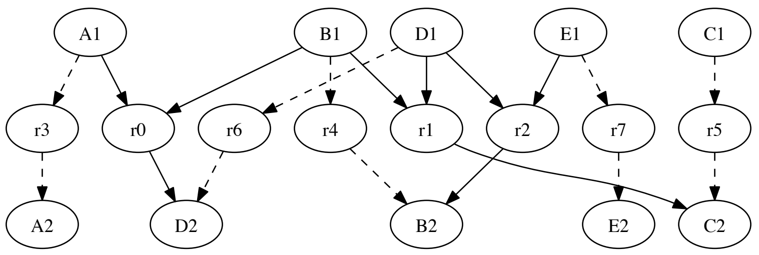

Figure 5: Rules $R_5 \cup Self_{R_5}$

of DD in one step. What happens in this case is that rule r_3 is identified as a competitor of both r_0 and r_1. However, in a multiset like $ABBCE$ the application of r_3 (that actually prevents r_0 and r_1 to be applied) causes also r_2 to be applied, obtaining DDD as result. This shows that the proposed notions of competitor are not able to characterise *all* multisets that lead to the presence of a required multiset in a given number of steps. In this case it is not able to recognise that the application of a competitor rule has as side effect the application of some other rules that actually lead to the desired result.

4.5. Multiple Backward Steps

We now describe how to obtain a pattern that expresses sufficient conditions for the presence of a molecule after two or more steps starting from patterns expressing sufficient conditions after one step.

Example 4.7. *Let us consider the P system $\Pi_5 = (\{A, B, C, D, E, F\}, w_0^5, R_5)$ where the evolution rules in R_5, depicted in Fig. 5, are*

$$r_0 : AB \to D \qquad r_1 : BD \to C \qquad r_2 : ED \to B$$

Note that rule r_0 and r_1 are the ones of Example 4.3 and the pattern for the presence of molecule D, in one step, is the same of the previous example, namely $p = (AB, \{BD\}) \vee (D, \{BD, ED\})$. Intuitively, in order to obtain the pattern expressing the sufficient condition for the presence of D after two steps, we have to consider all the ways a multiset satisfying p can be obtained in one step.

The pattern p is satisfied by multiset containing A and B, or D. Hence, we could compute the patterns that predict the presence of A, B and D in one step, and use them to construct a pattern for the satisfaction of p after one step. In addition to this, we have to pay attention to the competitors of A, B and D mentioned in p, namely the set $\{BD, ED\}$. In order to construct the pattern for the satisfiability of p in one step we have to consider also all the ways the competitors BD and ED can be obtained in one step.

We formally define an operator that considers all the possible ways a set of multisets representing competitors can be obtained in one step.

Definition 4.6. *Given an alphabet V, we define*

$$\mathtt{Cr}(R, \{u_1, ..., u_m\}) = \{\mathtt{react}(n) \mid \ n \in \mathtt{AppRules}(u_i, R \cup Self_R)) \textit{ and } u_i \in \{u_1, ..., u_m\}\}$$

for any finite set $R \subseteq \mathcal{R}_V$ and any set $\{u_1, ..., u_m\}$ such that u_i is a finite multiset in V^*, for $i \in \{1, ..., m\}$.

For computing the predictor of D in two step, our starting point is the predictor of D in one step computed in Example 4.7, that is $(AB, \{BD\}) \vee (D, \{BD, ED\})$. Now we have to devise in some way all the multisets that could lead to the previous pattern in one step. This would give all the multiset patterns that lead to D in two steps.

In order to rewrite the multiset AB in one step, we need to compute the predictors of A in one step, that is $(A, \{AB\})$, and the predictor of B in one step, that is $(ED, \{BD\}) \vee (B, \{AB, BD\})$. Moreover, we need to know how we can obtain the competitor of AB and D in one step. To this aim, we look for the possible rewritings of BD and ED using the operator $\mathtt{Cr}(.)$, that is, $\mathtt{Cr}(R_5, \{BD\}) = \{BD, EDD, ABB\} = C_5$ and $\mathtt{Cr}(R_5, \{ED\}) = \{ED, EAB\} = C_6$. Hence, we can now construct a pattern that predicts the presence of D in two steps as follows:

$$((A, \{AB\} \cup C_5) \wedge (ED, \{BD\} \cup C_5)) \vee (A, \{AB\} \cup C_5) \wedge (B, \{AB, BD\} \cup C_5)) \\ \vee (AB, \{BD\} \cup C_5 \cup C_6) \vee (D, \{BD, ED\} \cup C_5 \cup C_6)$$

Using the result of Section 3.3, the previous multiset pattern can be simplified as follows:

$$((A, \{AB\}) \wedge (ED, \{BD, EDD\}) \vee (A, \{AB\}) \wedge (B, \{AB, BD\})) \\ \vee (AB, \{BD, ABB, EAB\}) \vee (D, \{BD, ED\}),$$

where a pattern like $(A, \{AB, ABB\})$ is simplified in $(A, \{AB\})$, since the presence of ABB in addition to AB in the set of competitors does not introduce any stronger constraint on molecule A.

The initial multiset AED satisfies the pattern. Indeed, in one step we obtain AB using the only enabled rule r_2 and in two steps we obtain D applying the only enabled rule r_0. Consider $AEDD$ that does not satisfy the pattern, after one step we obtain ABD using the only enabled rule r_2 but also rule r_1 is enabled and by applying it we obtain AC that does not contain D.

4.6. Definition of the Main Operator and Theoretical Results

In the previous sections we have described the ingredients for the computation of a pattern expressing sufficient conditions for the presence of an molecule s after k steps. Now, we formally define an operator $\mathtt{Sc}_\Pi$ that performs such a computation.

The definition is given inductively on the following order on $\mathcal{P} \times \mathbb{N}$, $(p_1, n_1) \sqsubseteq (p_2, n_2)$ iff $n_1 < n_2$ or $n_1 = n_2$ and p_2 is a multiset pattern more complex than p_1, i.e. $p_2 = p_1 \vee p_3$ or $p_2 = p_1 \wedge p_3$, for some pattern p_3.

Definition 4.7. *Let $\Pi = (V, w, R)$ be a P system. We define a function $\mathtt{Sc}_\Pi : V^* \times \mathbb{N} \to \mathcal{P}$ as follows:*

$$\mathtt{Sc}_\Pi(u, k) = \mathtt{Sca}_\Pi((u, \{\}), k)$$

where the auxiliary function $\mathtt{Sca}_\Pi : \mathcal{P} \times \mathbb{N} \to \mathcal{P}$ *is recursively defined as follows:*

$$\begin{aligned}
\mathtt{Sca}_\Pi(p, 0) &= p \\
\mathtt{Sca}_\Pi(p_1 \vee p_2, k) &= \mathtt{Sca}_\Pi(p_1, k) \vee \mathtt{Sca}_\Pi(p_2, k) \\
\mathtt{Sca}_\Pi(p_1 \wedge p_2, k) &= \mathtt{Sca}_\Pi(p_1, k) \wedge \mathtt{Sca}_\Pi(p_2, k) \\
\mathtt{Sca}_\Pi((u, \{u_1, ..., u_m\}), k) &= \mathtt{Sca}_\Pi(\bigwedge_{s \in \mathtt{supp}(u)} p(s^{|u|_s}, \{u_1, ..., u_m\}), k - 1)
\end{aligned}$$

where

$$p(s^i, U) = \bigvee_{n \in \mathtt{AppRules}(s^i, R \cup Self_R)} \left(\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \bigcup_{r' \in C_{123}} \{\mathtt{react}(r')\} \cup \mathtt{Cr}(R, U)) \right)$$

and

$$C_{123} = \mathtt{competitor}_1(r, R, s) \cup \mathtt{competitor}_2(r, R, n) \cup \mathtt{competitor}_3(r, R, n, s)$$

Now we present some lemmata that, step by step, lead to the main theorem stating that the $\mathtt{Sc}_\Pi(u, k)$ operator actually computes a pattern representing a sufficient condition for the presence of u after k steps. In the lemmata and in the main theorem, given two multisets w and w' and a set of evolution rules R, we will denote with $w \to_R w'$ the fact that w' can be obtained from w by applying rules in R in a maximally parallel way.

The first lemma states that the portion of the pattern computed by the operator and defined as $p(s^i, U)$ in Def. 5.1 is a predictor for the presence of i instances of molecule s after one step of evolution of the P system.

Lemma 4.1. *Given a P system* $\Pi = (V, w_0, R)$*,* $w \in V^*$ *and* $s^i \in V^*$ *with* $s \in V$ *and* $i > 0$*, if* $w \models p(s^i, \{\})$ *then* $\forall w' \in V^*$ *such that* $w \to_R w'$*, it holds* $s^i \subseteq^* w'$*.*

Proof. By definition, $\mathtt{Cr}(R, \{\}) = \{\}$, therefore, in this case, we have $p(s^i, \{\}) = \bigvee_{n \in \mathtt{AppRules}(s^i, R \cup Self_R)} (\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \{\mathtt{react}(r') | r' \in C_{123}\}))$. Assume now, by contradiction, that $w \models p(s^i, \{\})$ but there exists w' such that $w \to_R w'$ and $s^i \not\subseteq^* w'$. This implies that $w \models \bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \{\mathtt{react}(r') | r' \in C_{123}\})$ for at least one multiset n of rules in $R \cup Self_R$ such that $s^i \in \mathtt{prod}(n)$.

Let us denote the conjunction $\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \{\mathtt{react}(r') | r' \in C_{123}\})$ simply as CP. Note that $w \models CP$ implies that, for each $r \in \mathtt{supp}(n)$, $w \supseteq_* \mathtt{react}(r)^{|n|_r}$. Intuitively, this means that w could be rewritten applying each rule $r \in \mathtt{supp}(n)$ for the number of times required by the multiset n but we still are left to prove that all rules $r \in \mathtt{supp}(n)$ could be applied *simultaneously* each one for the number of times required by the multiset n. Assume, by contradiction that this is not the case, then there exists at least two rules r and r'' belonging to n such that $\mathtt{react}(r) \cap \mathtt{react}(r'') \neq \emptyset$ and such that $w \not\supseteq_* \mathtt{react}(r)^{|n|_r} \mathtt{react}(r'')^{|n|_{r''}}$. Note that at most one can be a self rule $s \to s$. Assume that if one is a self rule than it is the one called r. As a consequence, we are sure that r'' does not belong to $Self_R$. Since $r'' \in R$ and, by hypothesis, it belongs to n and it is such that $\mathtt{react}(r) \cap \mathtt{react}(r'') \neq \emptyset$ then, by definition, we have that $r'' \in \mathtt{competitor}_2(r, R, n)$. Therefore, when verifying that $w \models CP$, $\mathtt{react}(r'')$ has to be maximally matched in w before matching with $\mathtt{react}(r)^{|n|_r}$. Since $w \supseteq_* \mathtt{react}(r'')^{|n|_{r''}}$ but $w \not\supseteq_* \mathtt{react}(r)^{|n|_r}\mathtt{react}(r'')^{|r''|_n}$, this gives a contradiction. Hence, we can conclude that $w \supseteq_* \mathtt{react}(r_1)^{|n|_{r_1}} ....\mathtt{react}(r_t)^{|n|_{r_t}}$ for $r_1, ..., r_t \in \mathtt{supp}(n)$.

Now assume that w could be maximally rewritten with rules $\tilde{r}_1^{o_1}...\tilde{r}_h^{o_h}$ (of R) that, by hypothesis, give a w' satisfying $v^j \in w' \Rightarrow j < i$.

Therefore, there exist some rules in $\{r_1, ..., r_t\}$ such that they are not applied with the multiplicity required by n, when applying $\tilde{r}_1^{o_1}...\tilde{r}_h^{o_h}$. In more detail, the self rule $s \to s$ belongs to such set if the instances of s in w that are left unchanged when applying $\tilde{r}_1^{o_1}...\tilde{r}_t^{o_h}$ are less than the multiplicity of the self rule $s \to s$ in n.

Among all the rules of $\{r_1, ..., r_t\}$ satisfying the above property, let us consider the case in which there exists r that also satisfies the following property:

$$\{\tilde{r} \ \mid \tilde{r} \in \{\tilde{r}_1, ..., \tilde{r}_h\}, s \in \mathtt{prod}(\tilde{r}), \tilde{r} \notin \mathtt{supp}(n), \mathtt{react}(\tilde{r}) \cap \mathtt{react}(r) \neq \emptyset\} \subseteq \mathtt{competitor}_3(r, R, n, s) \tag{1}$$

In this case, since $\tilde{r}_1^{o_1}...\tilde{r}_h^{o_h}$ is a maximal rewriting of w such that r is not applied with the multiplicity required by n, it means that
$w \not\models (\mathtt{react}(r)^{|n|_r}, \{\mathtt{react}(\tilde{r}) \mid \tilde{r} \in \{\tilde{r}_1, ..., \tilde{r}_t\}, \tilde{r} \neq r, \mathtt{react}(r) \cap \mathtt{react}(\tilde{r}) \neq \emptyset\})$. Since we have assumed that 1 holds, we have three cases for each $\tilde{r} \in \{\tilde{r}_1, ..., \tilde{r}_h\} \subseteq R$ such that $r \neq \tilde{r}$ and $\mathtt{react}(r) \cap \mathtt{react}(\tilde{r}) \neq \emptyset$:

1. $s \notin \mathtt{prod}(\tilde{r})$ then, by definition, $\tilde{r} \in \mathtt{competitor}_1(r, R, s)$.
2. $s \in \mathtt{prod}(\tilde{r})$, $\tilde{r} \in n$, then, by definition, $\tilde{r} \in \mathtt{competitor}_2(r, R, n)$.
3. $s \in \mathtt{prod}(\tilde{r})$, $\tilde{r} \notin n$, in this case, since we have assumed that (1) holds, we can be sure that $\tilde{r} \in \mathtt{competitor}_3(r, R, n, s)$.

As a consequence, we have that $\{\mathtt{react}(\tilde{r}) \mid \tilde{r} \in \{\tilde{r}_1, ..., \tilde{r}_h\}, \tilde{r} \neq r, \mathtt{react}(r) \cap \mathtt{react}(\tilde{r}) \neq \emptyset\} \subseteq \{\mathtt{react}(r') \mid r' \in C_{123}\}$. Then, we have a contradiction since, from the last reasoning, we can conclude that $w \not\models (\mathtt{react}(r)^{|n|_r}, \{\mathtt{react}(\tilde{r}) \mid \tilde{r} \in \{\tilde{r}_1, ..., \tilde{r}_t\}, \tilde{r} \neq r, \mathtt{react}(r) \cap \mathtt{react}(\tilde{r}) \neq \emptyset\})$ but, by hypothesis, $w \models CP$ and, as a consequence, since $r \in n$, $w \models (\mathtt{react}(r)^{|n|_r}, \{\mathtt{react}(r') \mid r' \in C_{123}\})$.

Assume now that does not exist an r satisfying (1). This implies that for each r_i (with $i = 1, ..., t$) there exist (at least one) $\hat{r}_1^i...\hat{r}_{z_i}^i \in \{\tilde{r}_1, ..., \tilde{r}_h\}$ such that for $j = 1, ...z_i$, $\hat{r}_j^i \neq r_i$, $\mathtt{react}(\hat{r}_j^i) \cap \mathtt{react}(r_i) \neq \emptyset$, and $\hat{r}_j^i \notin C_{123}$. Therefore, for $i = 1, ..., t$ and $j = 1, ...z_i$, by definition, we have that it must be the case that $s \in \mathtt{prod}(\hat{r}_j^i)$ (otherwise $\hat{r}_j^i$ would belong to C_1) and $\hat{r}_j^i \notin \mathtt{supp}(n)$ (otherwise $\hat{r}_j^i$ would belong to C_2) and, for each combination m of rules of n that could not be maximally applied because we apply $\hat{r}_j^i$, we have that $\mathtt{prod}(m) \subseteq^* \mathtt{prod}(\hat{r}_j^i)$ (otherwise $\hat{r}_j^i$ would belong to C_3). For simplicity, let us say that a rule $\hat{r}$ covers a multiset $m \subseteq^* n$ iff $\mathtt{react}(m) \cap_* \mathtt{react}(\hat{r}) = \mathtt{react}(n) \cap_* \mathtt{react}(\hat{r})$ and, $\forall m' \subset^* m$, $\mathtt{react}(m') \cap_* \mathtt{react}(\hat{r}) \neq \mathtt{react}(n) \cap_* \mathtt{react}(\hat{r})$. It is worth noting that using a rule $\hat{r} \in \{\hat{r}_1^1...\hat{r}_{z_1}^1, ..., \hat{r}_1^t...\hat{r}_{z_t}^t\}$ instead of rules $r_1, ..., r_t$ to rewrite w cannot give any w' that does not contain s^i. This is because for each multiset m of rules $r_1, ..., r_t$ such that $m \subseteq^* n$, if $\hat{r}$ covers m, then $|\mathtt{prod}(\hat{r})|_s \subset^* |\mathtt{prod}(m)|_s$. Therefore we have a contradiction.

$\square$

The second lemma states that if a multiset w satisfies $p(s^i, U)$, then the multiset obtained after one evolution step will satisfy the basic pattern (s^i, U).

Lemma 4.2. *Given a P system $\Pi = (V, w_0, R)$, $w \in V^*$ and a basic pattern (s^i, U) with $s \in V$ and $i > 0$, if $w \models p(s^i, U)$ then $\forall w' \in V^*$ such that $w \to_R w'$, it holds $w' \models (s^i, U)$.*

Proof. By definition, from $w \models p(s^i, U)$, it follows $w \models p(s^i, \{\})$. By applying Lemma 4.1 we can conclude that $\forall w'$ such that $w \to_R w'$, $w' \supseteq^* s^i$. For simplicity assume that $w' \supseteq^* s^i$ but $w' \not\supseteq^* s^{i+1}$. The more general case can be obtained by applying the following reasoning more than once.

Assume now, by contradiction, that $w' \not\models (s^i, U)$. Then, there must be the case that $w' \supseteq^* u_1^{o_1}, ..., u_t^{o_t}$ for $\{u_1, ..., u_t\} = U$ but $w' \not\supseteq^* u_1^{o_1}, ..., u_t^{o_t} s^i$. This implies that there exists at least one u_j with $j \in \{1, ..., t\}$ such that $s \in u_j$ and $o_j > 0$, that is $u_j \subseteq^* w'$. Consider the multiset of rules $n = \tilde{r}_1^{\tilde{o}_1} ... \tilde{r}_k^{\tilde{o}_k}$, let us assume there is just one, used to obtain w' from w where the proper rule $\tilde{r} \in Self_R$ is used to indicate that an occurrence of a molecule is left unchanged. Then $w = pred(\tilde{r}_1)^{\tilde{o}_1}, ..., pred(\tilde{r}_k)^{\tilde{o}_k}$. Since $w' \supseteq^* u_j \supseteq^* s$ we can conclude that there exists a minimal multiset of rules $m \subseteq n$ such that $\mathtt{prod}(m) \supseteq^* u^j$. Note that $m \in \mathtt{AppRules}(u_j, R \cup Self_R)$, therefore $\mathtt{react}(m) \in \mathtt{Cr}(R, U)$ and $\mathtt{react}(m) \subseteq^* w$. Since $s \in u_j$, there exists a rule in m, let us call it $\tilde{r}_h$, such that $s \in \mathtt{prod}(\tilde{r}_h)$. By $\mathtt{react}(m) \subseteq^* w$, we derive that also $\mathtt{react}(\tilde{r}_h) \subseteq^* w$. Therefore, there exists at least one rule $\tilde{r}_h$ with $h \in \{1, ..., p\}$ such that $\tilde{r}_h \in \mathtt{supp}(n)$ such that $w \not\models (\mathtt{react}(\tilde{r})^{|n|_{\tilde{r}}}, \bigcup_{r' \in C_{123}} \{\mathtt{react}(r')\} \cup \mathtt{Cr}(R, U))$.
Hence, $w \not\models \bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \bigcup_{r' \in C_{123}} \{\mathtt{react}(r')\} \cup \mathtt{Cr}(R, U))$.

$\square$

The following result comes directly from the definition of multiset patterns

Lemma 4.3. *If $w \models (u, \{u_1, ..., u_s\})$ and $w \models (\bar{u}, \{u_1, ..., u_s\})$ with $u \cap \bar{u} = \emptyset$, then $w \models (u\bar{u}, \{u_1, ..., u_s\})$.*

Finally, the following lemma states that if a multiset w satisfies the pattern $\bigwedge_{s \in \mathtt{supp}(u)} p(s^{|u|_s}, U)$ with u a generic multiset, then the multiset obtained after one evolution step will satisfy the basic pattern (u, U).

Lemma 4.4. *Given a P system $\Pi = (V, w_0, R)$, $u \in V^*$, $w \in V^*$ and a basic pattern (u, U) with $u \in V^*$, if $w \models \bigwedge_{s \in \mathtt{supp}(u)} p(s^{|u|_s}, U)$ then $\forall w' \in V^*$ such that $w \to_R w'$, it holds $w' \models (u, U)$.*

Proof. Assume that $w \models \bigwedge_{s \in \mathtt{supp}(u)} p(s^{|u|_s}, U)$. This implies that $w \models p(s^{|u|_s}, U)$ for each $s \in \mathtt{supp}(u)$. By Lemma 4.2 we have that $\forall w'$ such that $w \to_R w'$, $w' \models (s^{|u|_s}, U)$ with $s \in \mathtt{supp}(u)$. Therefore, we can conclude that $\forall w'$ such that $w \to_R w'$, $w' \models \bigwedge_{s \in \mathtt{supp}(u)} (s^{|u|_s}, U)$. Since $s_1^{|u|_{s_1}} \cap s_2^{|u|_{s_2}} = \emptyset$ for $s_1, s_2 \in \mathtt{supp}(u)$, $s_1 \neq s_2$, by Lemma 4.3, we can conclude that $w' \models (u, U)$.

$\square$

Theorem 4.5. *Let $\Pi = (V, w_0, R)$ be a P system and let $p \in \mathcal{P}$ be a multiset pattern. If $w_0 \models \mathtt{Sca}_\Pi(p, k)$ then for any $w_1, \ldots, w_k$ such that $w_{i \in \{1, ..., k\}} \in V^*$ and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$, it holds $w_k \models p$.*

Proof. Assume that $w_0 \models \mathtt{Sca}_\Pi(p, k)$, the proof is by induction on the pair (p, k) considering the order $\sqsubseteq$ on $\mathcal{P} \times \mathbb{N}$ defined as $\mathcal{P} \times \mathbb{N}$, $(p_1, n_1) \sqsubseteq (p_2, n_2)$ iff $n_1 < n_2$ or $n_1 = n_2$ and p_2 is a multiset pattern that contains p_1.

The base case is when p is a basic multiset pattern $p = (u, U)$ and $k = 0$. In this case $\mathtt{Sca}_\Pi((u, U), 0) = (u, U)$, therefore, since by hypothesis $w_0 \models \mathtt{Sca}_\Pi(p, 0)$ we have that $w_0 \models p$.

For the inductive case, we have that either p is not a basic multiset pattern or $p = (u, U)$ and $k > 0$. We consider these cases separately.

- $p = p_1 \wedge p_2$. In this case since $\mathtt{Sca}_\Pi(p, k) = \mathtt{Sca}_\Pi(p_1, k) \wedge \mathtt{Sca}_\Pi(p_2, k)$, if $w_0 \models \mathtt{Sca}_\Pi(p, k)$ then $w_0 \models \mathtt{Sca}_\Pi(p_1, k)$ and $w_0 \models \mathtt{Sca}_\Pi(p_2, k)$. By induction hypothesis, for any $w_1, \ldots, w_k$ such that $w_{i\in\{1,\ldots,k\}} \in V^*$ and $w_0 \rightarrow_R w_1 \rightarrow_R \ldots \rightarrow_R w_k$, it holds $w_k \models p_1$ and for any $w_1, \ldots, w_k$ such that $w_{i\in\{1,\ldots,k\}} \in V^*$ and $w_0 \rightarrow_R w_1 \rightarrow_R \ldots \rightarrow_R w_k$, it holds $w_k \models p_2$. Hence, we can conclude that for any $w_1, \ldots, w_k$ such that $w_{i\in\{1,\ldots,k\}} \in V^*$ and $w_0 \rightarrow_R w_1 \rightarrow_R \ldots \rightarrow_R w_k$, it holds $w_k \models (p_1 \wedge p_2) = p$.

- $p = p_1 \vee p_2$. In this case the proof is analogous to the previous case.

- $p = (u, U)$ and $k > 0$. In this case since $\mathtt{Sca}_\Pi(p, k) =$ $\mathtt{Sca}_\Pi(\bigwedge_{s\in\mathsf{supp}(u)} p(s^{|u|_s}, \{u_1, ..., u_t\}), k - 1)$, if $w_0 \models \mathtt{Sca}_\Pi(p, k)$ then $w_0 \models \mathtt{Sca}_\Pi(\bigwedge_{s\in\mathsf{supp}(u)} p(s^{|u|_s}, \{u_1, ..., u_t\}), k - 1)$. By induction hypothesis, we have that for any $w_1, \ldots, w_{k-1}$ such that $w_{i\in\{1,\ldots,k-1\}} \in V^*$ and $w_0 \rightarrow_R w_1 \rightarrow_R \ldots \rightarrow_R w_{k-1}$, it holds $w_{k-1} \models \bigwedge_{s\in\mathsf{supp}(u)} p(s^{|u|_s}, \{u_1, ..., u_t\})$. By Lemma 4.4 we have that $\forall w_k \in V^*$ such that $w_{k-1} \rightarrow_R w_k$, it holds $w_k \models (u, U) = p$.

□

We are now ready to state the main result of this paper based on Theorem 4.5.

Corollary 4.6 (Sufficient Condition). *Let $\Pi = (V, w_0, R)$ be P system and $u \in V^*$. If $w_0 \models \mathtt{Sc}_\Pi(u, k)$ then for any $w_1, \ldots, w_k$ such that $w_{i\in[1,k]} \in V^*$ and $w_0 \rightarrow_R w_1 \rightarrow_R \ldots \rightarrow_R w_k$, it holds $u \subseteq^* w_k$.*

Proof. Since $\mathtt{Sc}_\Pi(u, k) = \mathtt{Sca}_\Pi((u, \{\}), k)$, if $w_0 \models \mathtt{Sc}_\Pi(u, k)$ then we have that $w_0 \models \mathtt{Sca}_\Pi((u, \{\}), k)$. By Theorem 4.5 we have that for any $w_1, \ldots, w_k$ such that $w_{i\in\{1,\ldots,k\}} \in V^*$ and $w_0 \rightarrow_R w_1 \rightarrow_R \ldots \rightarrow_R w_k$, it holds $w_k \models (u, \{\})$. By definition of multiset pattern note that $w_k \models (u, \{\})$ iff $w_k \supseteq^* u$. □

The corollary essentially states that the pattern computed by the $\mathtt{Sc}_\Pi$ operator is a *sound* sufficient predictor.

Corollary 4.6 proves a righthand implication, that is, if an initial state of a P system satisfies $\mathtt{Sc}_\Pi(u, k)$ then it will surely lead after k steps to a multiset that contains u. The left-hand implication does not hold for any P systems. Indeed, there can be multisets that do not match the pattern but still always lead to the presence of u after k steps (see, for example, the discussion of Example 4.6 at the end of Section 4.3).

However, the left-hand implication may hold in some special cases. We can prove that left-hand implication of Corollary 4.6 holds for P systems containing non-cooperative rules only either when $k = 1$ or for any k when the system has the additional property of being strong reversible [18, 19].

Lemma 4.7. *Consider a P system with non-cooperative rules $\Pi = (V, w_0, R)$. Let $w \in V^*$ be a multiset. If for all w' such that $w \rightarrow_R w'$ we have that $s^i \subseteq^* w'$, for some $s \in V$ and $i > 0$, then $w \models p(s^i, \{\})$.*

Proof. First note that in this case $v \to v \in Self_R$ if and only if there does not exists any rule $v \to u$ with $v \in V$ and $u \in V^+$.

Assume that for all w' such that $w \to_R w'$ we have that $s^i \subseteq^* w'$. Between all the multiset of rules in $R \cup Self_R$ that rewrite w in a maximal way, choose one multiset n of rules that produce s^i and that satisfies the following conditions:

(i) $\forall r \in \mathtt{supp}(n)$ there does not exists $r' \in \mathtt{supp}(n)$ such that $\mathtt{react}(r) = \mathtt{react}(r')$,

(ii) $\forall r \in \mathtt{supp}(n), r' \in R$ such that $\mathtt{react}(r) = \mathtt{react}(r')$ and $s \in \mathtt{prod}(r')$ we have that $\mathtt{prod}(r)_{|s} \leq \mathtt{prod}(r')_{|s}$,

(iii) n is minimal, i.e. $\forall n' \subseteq^* n$, n' does not produce s^i.

Note that we can always find an n satisfying (i), (ii) and (iii) since if $\mathtt{react}(r) = \mathtt{react}(r')$ implies that if $w \supseteq \mathtt{react}(r)$ then $w \supseteq \mathtt{react}(r')$. Since the multiset of rules n are used to rewrite w, we can conclude that $\mathtt{react}(r_1)^{|n|_{r_1}} \ldots\ldots \mathtt{react}(r_m)^{|n|_{r_m}} \subseteq^* w$ for $r_1, \ldots\ldots, r_m \in \mathtt{supp}(n)$.

Observe that $n \in \mathtt{AppRules}(s^i, R \cup Self_R)$, since all possible rewritings of w lead to the production of s^i. Consider now the following multiset pattern:

$$\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \bigcup_{r' \in \mathtt{competitor}_1(r,R,s) \cup \mathtt{competitor}_2(r,R,n) \cup \mathtt{competitor}_3(r,R,n,s)} \{\mathtt{react}(r')\} \cup \mathtt{Cr}(R, \{\})). \quad (2)$$

Since $\mathtt{Cr}(R, \{\}) = \{\}$, we can rewrite multiset pattern 2 as follows,

$$\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \bigcup_{r' \in \mathtt{competitor}_1(r,R,s) \cup \mathtt{competitor}_2(r,R,n) \cup \mathtt{competitor}_3(r,R,n,s)} \{\mathtt{react}(r')\}. \quad (3)$$

We now prove that $\forall r$ such that $r \in \mathtt{supp}(n)$,
$\mathtt{competitor}_1(r, R, s) \cup \mathtt{competitor}_2(r, R, n) \cup \mathtt{competitor}_3(r, R, n, s) = \emptyset$.
Indeed, $r' \notin \mathtt{competitor}_1(r, R, s)$, this is because otherwise there would exist a maximal rewriting of w using rule r' instead of rule r but not producing s^i (see condition (iii)). Moreover, condition (i) implies that $r' \notin \mathtt{competitor}_2(r, R, n)$, while condition (ii) implies that $r' \notin \mathtt{competitor}_3(r, R, n, s)$. Therefore multiset pattern (3) boils down to be

$$\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \{\}). \quad (4)$$

Since $\mathtt{react}(r_1)^{|n|_{r_1}} \ldots\ldots \mathtt{react}(r_m)^{|n|_{r_m}} \subseteq^* w$ for $r_1, \ldots\ldots, r_m \in \mathtt{supp}(n)$ and multiset pattern 3 is equivalent to 4, we can conclude that w models multiset pattern 2. The last step consists of observing that multiset pattern 2 is one of the multiset of the disjunction $p(s^i, \{\})$, therefore, since we have prove that w models 2, we have that $w \models p(s^i, \{\})$. $\square$

Theorem 4.8. *Consider a P system with only non-cooperative rules* $\Pi = (V, w_0, R)$. *Let* $w \in V^*$ *be a multiset. If for all* w' *such that* $w \to_R w'$ *we have that* $u \subseteq^* w'$, *for some multiset* $u \in V^*$, *then* $w \models \mathtt{Sc}_\Pi(u, 1)$.

Proof. Consider $u = s_1^{o_1}...s_m^{o_m}$ where $s_1, ..., s_m \in \mathtt{supp}(u)$. Since for all w_1 such that $w \rightarrow_R w_1$ we have that $s_i^{o_i} \subseteq^* w'$ for $i \in \{1, ..., m\}$ then, by Lemma 4.7, we have that $w \models p(s_i^{o_i}, \{\})$. Therefore, $w \models \bigwedge_{s \in supp(u)} p(s_i^{o_i}, \{\})$. Then, by definition, we have that $w \models \mathtt{Sc}_\Pi(u, 1)$. □

When the P system is also strong reversible, i.e. each configuration can be obtained by at most one single previous configuration, we can prove the following.

Theorem 4.9. *Consider a strong reversible P system with non-cooperative rules $\Pi = (V, w_0, R)$. Let $w \in V^*$ be a multiset. If for all $w_1, ..., w_k$ such that $w \rightarrow_R w_1 \rightarrow_R ... \rightarrow_R w_k$ we have that $u \subseteq^* w_k$, for some $u \in V^*$, then $w \models \mathtt{Sc}_\Pi(u, k)$.*

Proof. The proof is by induction on k.

Base case: The base case is when $k = 1$. In this case by applying Theorem 4.8 we have that $w \models \mathtt{Sc}_\Pi(u, k)$.

Inductive case: In this case consider the single configuration w_{k-1} such that $w_{k-1} \rightarrow_R w_k$. We have that $w \rightarrow_R w_1 \rightarrow_R ... \rightarrow_R w_{k-1}$ in $k - 1$ steps. Consider the set $C = \{\mathtt{react}(n) \mid n \in \mathtt{AppliedRules}(u, R)\}$. Since P is strong reversible and therefore each configuration can be obtained by at most one single previous configuration, we have that $\forall n, n' \in \mathtt{AppliedRules}(u, R)$, $\mathtt{react}(n) = \mathtt{react}(n')$. Moreover, $\mathtt{react}(n) \subseteq^* w_k$. By inductive hypothesis, we have that $w \models \mathtt{Sc}_\Pi(\mathtt{react}(n), k - 1)$.

We are left to prove that $\mathtt{Sc}_\Pi(react(n), k - 1)$ is equal to $\mathtt{Sc}_\Pi(u, k)$. Note that $\mathtt{Sc}_\Pi(u, k)$ is defined as
$\mathtt{Sc}_\Pi(\bigwedge_{s \in \mathtt{supp}(u)} \left(\bigvee_{n \in \mathtt{AppRules}(s^i, R \cup Self_R)} \left(\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \bigcup_{r' \in C_{123}} \{\mathtt{react}(r')\})\right)\right), k - 1)$.
The first observation is that the pattern
$\bigwedge_{s \in \mathtt{supp}(u)} \left(\bigvee_{n \in \mathtt{AppRules}(s^i, R \cup Self_R)} \left(\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \bigcup_{r' \in C_{123}} \{\mathtt{react}(r')\})\right)\right)$
can be simplified in
$\bigwedge_{s \in \mathtt{supp}(u)} \left(\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}, \bigcup_{r' \in C_{123}} \{\mathtt{react}(r')\})\right)$,
for a unique $n \in \mathtt{AppliedRules}(s^i, R \cup Self_R)$ since multiset s^i can be obtained by at most one single previous multiset. Moreover, since $\forall w_k$ such that $w_{k-1} \rightarrow_R w_k$ we have that $u \subseteq^* w_k$, together with the assumption that the P system is strong reversible, allow us to further simplify the latter pattern in
$\bigwedge_{s \in \mathtt{supp}(u)} \left(\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}\right)$.
The last step consists in observing that all rules are non cooperatives, therefore, for a rule r involved in the production of $s^i \subseteq^* u$ and a rule r' involved in the production of $v^i \subseteq^* u$ with $s \neq v$, $\mathtt{react}(r) \cap \mathtt{react}(r') \neq \emptyset$ implies that $prod(r) = prod(r') \supseteq^* sv$.

Hence, in this case $\bigwedge_{s \in \mathtt{supp}(u)} \left(\bigwedge_{r \in \mathtt{supp}(n)} (\mathtt{react}(r)^{|n|_r}\right) = react(n)$. This allows us to conclude that in this case $\mathtt{Sc}_\Pi(react(n), k - 1)$ is equal to $\mathtt{Sc}_\Pi(u, k)$.

□

5. Necessary Predictors

As before, we propose a methodology based on multiset patterns to compute necessary conditions for the presence of a multiset u after k evolution steps of a given P system. The necessary condition will be expressed as a pattern (called *necessary predictor*) to be satisfied by the initial multiset w of the P system.

To devise necessary conditions for a multiset u to be present after k steps is a much more easy task because competition between rules does not need to be taken into account when considering all the ways a multiset containing u can be obtained.

Example 5.1. *Consider again the P system $\Pi_5 = (\{A, B, C, D, E, F\}, w_0^5, R_5)$ where the evolution rules in R_5, depicted in Fig. 5, are*

$$r_0 : \;\; AB \to D \qquad\qquad r_1 : \;\; BD \to C \qquad\qquad r_2 : \;\; ED \to B$$

In order to obtain the pattern expressing the necessary conditions for the presence of DC after one step, intuitively we have to consider all the ways we can obtain the multiset DC after one step. Therefore, the necessary multiset pattern is obtained by considering all the possible combination of rules that evolves into a multiset containing molecules DC without considering competitions between rules. In this case the required pattern is given by $(DC, \{\}) \vee (ABC, \{\}) \vee (DBD, \{\}) \vee (ABBD, \{\})$. Note that since competition between rules is ignored, in order to obtain CD we do not need to consider the production of C and D separately. This will simplify the definition of the operator that computes the necessary predictor.

It is easy to see that if the initial multiset w_0^5 evolves into a multiset containing CD then the initial multiset has to contain at least one multiset between DC, ABC, DBD or ABBD, as required by pattern $(DC, \{\}) \vee (ABC, \{\}) \vee (DBD, \{\}) \vee (ABBD, \{\})$.

When interested to the pattern expressing the necessary conditions for the presence of DC after two steps, we simply have to find all ways each simple pattern can be obtained. In this case, the pattern expressing the necessary conditions for the presence of DC after two steps is $(DC, \{\}) \vee (ABC, \{\}) \vee (DBD, \{\}) \vee (ABBD, \{\}) \vee (DEDD, \{\}))$.

Note that when interested in the presence of a multiset u after a given number of steps, necessary conditions can be used to devise, through complementation, initial multisets that will *never* evolve in the required number of steps into a multiset containing u. Indeed, in the previous example we can be sure that if a multiset does not contain neither DC nor ABC nor DBD nor $ABBD$ then it will not able to evolve into a multiset containing CD.

We can now give the definition of the operator $\mathtt{Nc}_\Pi$ that computes the necessary predictor. As for the case of sufficient conditions, the definition is given inductively on the order $\sqsubseteq$ on $\mathcal{P} \times \mathbb{N}$.

Definition 5.1. *Let $\Pi = (V, w, R)$ be a P system. We define a function $\mathtt{Nc}_\Pi : V^* \times \mathbb{N} \to \mathcal{P}$ as follows:*

$$\mathtt{Nc}_\Pi(u, k) = \mathtt{Nca}_\Pi((u, \{\}), k)$$

where the auxiliary function $\mathtt{Nca}_\Pi : \mathcal{P} \times \mathbb{N} \to \mathcal{P}$ is recursively defined as follows:

$$\begin{aligned}
\texttt{Nca}_\Pi((u,\{\}),0) &= (u,\{\})\\
\texttt{Nca}_\Pi(p_1 \vee p_2, k) &= \texttt{Nca}_\Pi(p_1,k) \vee \texttt{Nca}_\Pi(p_2,k)\\
\texttt{Nca}_\Pi(p_1 \wedge p_2, k) &= \texttt{Nca}_\Pi(p_1,k) \wedge \texttt{Nca}_\Pi(p_2,k)\\
\texttt{Nca}_\Pi((u,\{\}),k) &= \texttt{Nca}_\Pi(\bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)} (\texttt{react}(n),\{\}), k-1)
\end{aligned}$$

Lemma 5.1. *Given a P system $\Pi = (V, w_0, R)$, $u \in V^*$, $w' \in V^*$ and a basic pattern $(u,\{\})$ with $u \in V^*$, if $w' \models (u,\{\})$ then $\forall w \in V^*$ such that $w \to_R w'$, it holds that $w \models \bigvee_{n \in \texttt{AppRules}(u,R \cup Self_R)}(\texttt{react}(n),\{\})$.*

Proof. We prove that $w \supseteq^* \texttt{react}(n)$ for at least one $n \in \texttt{AppRules}(u, R \cup Self_R)$. Since $w' \models (u, U)$ we have that $w' \supseteq^* u$. Now since $w \to_R w'$, consider the multiset n of rules of $R \cup Self_R$ that when applied to w gave $u \subseteq^* w'$, where if a molecule s was not produced by a rule but it was already present in w we consider the self rule $s \to s$ in n. Since n was the multiset of rules that was actually applied to w in order to obtain w', we can conclude that $\texttt{react}(n) \subseteq^* w$. Hence, $w \models \bigvee_{n \in \texttt{AppRules}(u,R \cup Self_R)}(\texttt{react}(n),\{\})$.

□

Theorem 5.2. *Let $\Pi = (V, w_0, R)$ be P system and let $p \in \mathcal{P}$ be a multiset pattern where the second element of the pair is always the empty set. If $w_k \models p$ then for any $w_1, \ldots, w_k$ such that $w_{i \in \{1,\ldots,k\}} \in V^*$ and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$, it holds $w_0 \models \texttt{Nca}_\Pi(p,k)$.*

Proof. Assume that $w_k \models p$, the prove is by induction on the pair (p,k) considering the order $\sqsubseteq$ on $\mathcal{P} \times \mathbb{N}$.

Base case: The base case is when p is a basic multiset pattern $p = (u,\{\})$ and $k = 0$. In this case, by hypothesis $w_0 \models p$, since $\texttt{Nca}_\Pi(p,0) = p$, we have that $w_0 \models \texttt{Nca}_\Pi(p,0)$.

Inductive case: Here we have that either p is not a basic multiset pattern or $p = (u,\{\})$ and $k > 0$. We handle these cases separately.

- $p = p_1 \wedge p_2$. In this case, if $w_k \models p$ this implies that $w_k \models p_1$ and $w_k \models p_2$. By inductive hypothesis, for any $w_1, \ldots, w_{k-1}$ such that $w_{i \in \{1,\ldots,k\}} \in V^*$ and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$, it holds $w_0 \models \texttt{Nca}_\Pi(p_1,k)$ and for any $w_1, \ldots, w_{k-1}$ such that $w_{i \in \{1,\ldots,k\}} \in V^*$ and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$, it holds $w_0 \models \texttt{Nca}_\Pi(p_2,k)$. Hence, since $\texttt{Nca}_\Pi(p,k) = \texttt{Nca}_\Pi(p_1,k) \wedge \texttt{Nca}_\Pi(p_2,k)$, we can conclude that for any $w_1, \ldots, w_k$ such that $w_{i \in \{1,\ldots,k\}} \in V^*$ and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$, it holds $w_0 \models \texttt{Nca}_\Pi(p_1 \wedge p_2, k) = \texttt{Nca}_\Pi(p,k)$.
- $p = p_1 \vee p_2$. In this case since $w_k \models p$ then either $w_k \models p_1$ holds or $w_k \models p_2$ holds. Without losing generality assume that $w_k \models p_1$. By inductive hypothesis, for any $w_1, \ldots, w_k$ such that $w_{i \in \{1,\ldots,k\}} \in V^*$ and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$, it holds $w_0 \models \texttt{Nca}_\Pi(p_1,k)$. Hence, we can conclude that for any $w_1, \ldots, w_k$ such that $w_{i \in \{1,\ldots,k\}} \in V^*$ and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$, it holds $w_0 \models \texttt{Nca}_\Pi((p_1 \vee p_2),k) = \texttt{Nca}_\Pi(p,k)$.

- $p = (u, \{\})$ and $k > 0$. In this case by hypothesis, $w_k \models p$. By applying Lemma 5.1 we have that $w_{k-1} \models \bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)}(\texttt{react}(n), \{\})$. By inductive hypothesis, we have that $w_0 \models \texttt{Nca}_\Pi(\bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)}(\texttt{react}(n), \{\}), k-1)$.
  By definition, $\texttt{Nca}_\Pi(\bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)}(\texttt{react}(n), \{\}), k-1) = \texttt{Nca}_\Pi((u, \{\}), k)$. This concludes the proof.

□

Corollary 5.3 (Necessary Condition). *Let $\Pi = (V, w_0, R)$ be P system and $u, w_0 \in V^*$. If there exists a w_k such that $w_k \sqsupseteq^* u$, and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$, then it holds that $w_0 \models \texttt{Nc}_\Pi(u, k)$.*

Proof. Assume that $w_k \models u$, by Theorem 5.2 we have that for any $w_1, \ldots, w_k$ such that $w_{i \in \{1,\ldots,k-1\}} \in V^*$ and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$, it holds $w_0 \models \texttt{Nca}_\Pi(u, \{\}, k)$. Since $\texttt{Nc}_\Pi(u, k) = \texttt{Nca}_\Pi((u, \{\}), k)$, we have that $w_0 \models \texttt{Sc}_\Pi(u, k)$. This concludes the proof. □

Once more, Corollary 5.3 proves a righthand implication, that is, if there exists a w_k such that $w_0 \to_R w_1 \to_R \ldots \to_R w_k$ and $w_k \models (u, \{\})$, then it holds that $w_0 \models \texttt{Nc}_\Pi(u, k)$. In general the left-hand implication does not hold, that is, there might exist w_0 modelling $\texttt{Nc}_\Pi(u, k)$ that do not lead to the production of u in k steps.

Example 5.2. *Consider a P system with just one rule $AB \to D$. By definition, $\texttt{Nc}_\Pi(A, 2) = (A, \{\})$. Consider the multiset AB and note that $AB \models \texttt{Nc}_\Pi(A, 2)$. However, there does not exists any maximal rewriting of AB than leads to the production of A in two steps.*

The left-hand implication of Corollary 5.3 holds for P systems with non cooperative rules. This time without any further restriction.

Theorem 5.4. *Let $\Pi = (V, w_0, R)$ be P system with non-cooperative rules and let $u, w_0 \in V^*$. If $w_0 \models \texttt{Nc}_\Pi(u, k)$ then there exists a w_k such that $w_k \sqsupseteq^* u$, and $w_0 \to_R w_1 \to_R \ldots \to_R w_k$.*

Proof. As we have already pointed out, in this case $v \to v \in Self_R$ if and only if there does not exists any rule $v \to u$ with $v \in V$ and $u \in V^+$.

Assume now that $w_0 \models \texttt{Nc}_\Pi(u, k)$, the prove is by induction on k.

Base case: The base case is when $k = 1$. In this case, by definition,
$\texttt{Nca}_\Pi((u, \{\}), 1) = \texttt{Nca}_\Pi(\bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)}(\texttt{react}(n), \{\}), 0) = \bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)}(\texttt{react}(n), \{\})$.
By hypothesis we have that $w_0 \models \texttt{Nc}_\Pi(u, 1) = \bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)}(\texttt{react}(n), \{\})$. Then there must exists at least one $n \in \texttt{AppRules}(u, R \cup Self_R)$ such that $w_0 \models \texttt{react}(n)$. Since a rule, in this case a self-rule $v \to v$ is introduced in $Self_R$ iff there does not exists any rule $r' \in R$ such that $\texttt{react}(r') = v$, we can conclude that $w_0 \to_R w_1$ where if a self rule $v \to v$ is used in the multiset of rules n then it means that v is left unchanged.

Inductive case: In this case, by definition,
$\texttt{Nca}_\Pi((u, \{\}), k) = \texttt{Nca}_\Pi(\bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)}(\texttt{react}(n), \{\}), k-1)$ and
$\texttt{Nca}_\Pi(\bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)}(\texttt{react}(n), \{\}), k-1) \bigvee_{n \in \texttt{AppRules}(u, R \cup Self_R)} \texttt{Nca}_\Pi(\texttt{react}(n), \{\})$. By hypothesis we have that

$w_0 \models \mathtt{Nc}_\Pi(u, 1) = \mathtt{Nca}_\Pi((u, \{\}), k) = \bigvee_{n \in \mathtt{AppRules}(u, R \cup Self_R)} \mathtt{Nca}_\Pi(\mathtt{react}(n), \{\})$. Then there must exists at least one $n \in \mathtt{AppRules}(u, R \cup Self_R)$ such that
$w_0 \models \mathtt{Nca}_\Pi(\mathtt{react}(n), k-1)$. By inductive hypothesis, we have that $w_0 \rightarrow_R \ldots\ldots \rightarrow_R w_{k-1}$. Since $n \in \mathtt{AppRules}(u, R \cup Self_R)$, we can conclude that $w_0 \rightarrow_R \ldots\ldots \rightarrow_R w_{k-1} \rightarrow_R w_k$, where if a self rule $v \rightarrow v$ is used in the last step then it means that v is left unchanged, and $w_k \supseteq^* u$.

This completes the proof. □

6. Applications of Predictors

6.1. P Systems as Language Acceptors

Let us consider again the example of the multiset language consisting only of the multiset A^3 discussed in Section 3. An acceptor for such a language can be represented by the P system $\Pi_6 = (\{O, A, T, F\}, w_0^6, R_6)$, where R_6 consists of the following rules:

$$r_0 : \ OA^3 \rightarrow T \qquad\qquad r_1 : \ TA \rightarrow F$$

The acceptor works by assuming that $|w_0^6|_O = 1$ and $|w_0^6|_T = |w_0^6|_F = 0$. If $|w_0^6|_A = 3$, then in one step T is produced and it is left unchanged in the second step (actually, the P system terminates after one step). If $|w_0^6|_A \neq 3$, then either T is not produced, or it is replaced by F in the second step. As a consequence, T will be present after two steps iff $|w_0^6|_A = 3$.

Let us compute the sufficient predictor of T in two steps for the P system Π_6 by applying the $\mathtt{Sc}_{\Pi_6}$ operator:

$$\begin{aligned}
\mathtt{Sc}_{\Pi_6}(T, 2) &= \mathtt{Sca}_{\Pi_6}((T, \{\}), 2) = \mathtt{Sca}_{\Pi_6}((OA^3, \{TA\}) \vee (T, \{TA\}), 1) \\
&= \mathtt{Sca}_{\Pi_6}((OA^3, \{TA\}), 1) \vee \mathtt{Sca}_{\Pi_6}((T, \{TA\}), 1) \\
&= \mathtt{Sca}_{\Pi_6}((OA^3, \{TA, OA^4\}), 0) \vee \mathtt{Sca}_{\Pi_6}((OA^3, \{TA, OA^4\}) \vee (T, \{TA\}), 0) \\
&= \mathtt{Sca}_{\Pi_6}((OA^3, \{TA, OA^4\}), 0) \vee \mathtt{Sca}_{\Pi_6}((OA^3, \{TA, OA^4\}), 0) \vee \mathtt{Sca}_{\Pi_6}((T, \{TA\}), 0) \\
&= (OA^3, \{TA, OA^4\}) \vee (OA^3, \{TA, OA^4\}) \vee (T, \{TA\}) \\
&= (OA^3, \{TA, OA^4\}) \vee (T, \{TA\})\,.
\end{aligned}$$

Assumptions $|w_0^6|_T = 0$ allow us to simplify the obtained pattern into $(OA^3, \{OA^4\})$. Using now the assumption $|w_0^6|_O = 1$ allows us to model the language A^3.

We now compute the necessary predictor of T in two steps by applying the $\mathtt{Nc}_{\Pi_6}$ operator:

$$\begin{aligned}
\mathtt{Nc}_{\Pi_6}(T, 2) &= \mathtt{Nca}_{\Pi_6}((T, \{\}), 2) = \mathtt{Nca}_{\Pi_6}((OA^3, \{\}) \vee (T, \{\}), 1) \\
&= \mathtt{Nca}_{\Pi_6}((OA^3, \{\}), 1) \vee \mathtt{Nca}_{\Pi_6}((T, \{\}), 1) \\
&= \mathtt{Nca}_{\Pi_6}((OA^3, \{\}), 0) \vee \mathtt{Nca}_{\Pi_6}(T, \{\}), 0) \\
&= (OA^3, \{\}) \vee (T, \{\}).
\end{aligned}$$

Such pattern can be used to devise initial multisets that will never be able to evolve into multisets containing molecule T. Indeed, if a multiset $w_0^6 \not\models \mathtt{Nc}_{\Pi_6}(T, 2)$ then we can conclude that it cannot evolve in two steps into a multiset containing molecule T. In this case, we are sure that initial multisets w_0^6 such as $\emptyset$, A or AA cannot evolve into multiset containing T, thus assuring that the presence of T depends on more than two occurrences of molecule A.

We now consider an acceptor for the language A^nB^n. As in Section 3, we start by focusing on the complement of A^nB^n, namely A^nB^m with $n \neq m$. Let $\Pi_7 = (\{O, D, A, B, C, T, F\}, w_0, R_7)$ be a P system where rules in R_7 are:

$$r_0: \; AB \to C \qquad r_1: \; AD \to T \qquad r_2: \; BD \to T \qquad r_3: \; O \to D \qquad r_4: \; FT \to T$$

Let us assume that the initial multiset w_0^7 contains exactly one F, one O and no instances of T and of D, namely $|w_0^7|_F = |w_0^7|_O = 1$ and $|w_0^7|_T = |w_0^7|_D = 0$. Under such an assumption, the evolution Π_7 is as follows: in the first step a maximal number of AB pairs are consumed by rule r_0 and, at the same time, molecule O is transformed into molecule D by rule r_3. In the second step, if either some A or some B is still present, that is if the number of A was not the same as the number of B in the initial multiset, then one instance of T is produced by either r_1 or r_2. If T is produced, it causes F to be removed in the third step due to the application of rule r_4. As a consequence, after three steps molecule T is present iff the initial multiset contained different numbers of A and B. Otherwise, molecule F is present instead of T.

Let us compute the predictor of T after three steps for the P system Π_7 by applying the $\mathtt{Sc}_{\Pi_7}$ operator:

$$\begin{aligned}
\mathtt{Sc}_{\Pi_7}(T,3) &= \mathtt{Sca}_{\Pi_7}((T,\{\}),3) \\
&= \mathtt{Sca}_{\Pi_7}((T,\{\}) \vee (FT,\{\}) \vee (BD,\{AB\}) \vee (AD,\{AB\}),2) \\
&= \mathtt{Sca}_{\Pi_7}((T,\{\}),2) \vee \mathtt{Sca}_{\Pi_7}((FT,\{\}),2) \vee \mathtt{Sca}_{\Pi_7}((BD,\{AB\}),2) \vee \mathtt{Sca}_{\Pi_7}((AD,\{AB\}),2) \\
&= \mathtt{Sca}_{\Pi_7}((T,\{\}),1) \vee \mathtt{Sca}_{\Pi_7}((FT,\{\}),1) \vee \mathtt{Sca}_{\Pi_7}((BD,\{AB\}),1) \vee \mathtt{Sca}_{\Pi_7}((AD,\{AB\}),1) \\
&\quad \vee \mathtt{Sca}_{\Pi_7}((BO,\{AB\}),1) \vee \mathtt{Sca}_{\Pi_7}((AO,\{AB\}),1) \\
&= \mathtt{Sca}_{\Pi_7}((T,\{\}),0) \vee \mathtt{Sca}_{\Pi_7}((FT,\{\}),0) \vee \mathtt{Sca}_{\Pi_7}((BD,\{AB\}),0) \vee \mathtt{Sca}_{\Pi_7}((AD,\{AB\}),0) \\
&\quad \vee \mathtt{Sca}_{\Pi_7}((BO,\{AB\}),0) \vee \mathtt{Sca}_{\Pi_7}((AO,\{AB\}),0) \\
&= (T,\{\}) \vee (FT,\{\}) \vee (BD,\{AB\}) \vee (AD,\{AB\}) \vee (BO,\{AB\}) \vee (AO,\{AB\}) \\
&= (T,\{\}) \vee (BD,\{AB\}) \vee (AD,\{AB\}) \vee (BO,\{AB\}) \vee (AO,\{AB\}) \,.
\end{aligned}$$

Note that pattern $(T,\{\}) \vee (FT,\{\})$ can be simplified to $(T,\{\})$ because $T \subseteq^* FT$:

The assumptions on the absence of T and D and on the presence of O in the initial multiset make the obtained pattern equivalent to $(B,\{AB\}) \vee (A,\{AB\})$, that is exactly the pattern we identified in Section 3 for A^nB^m with $n \neq m$. Moreover,

$$\begin{aligned}
\mathtt{Nc}_{\Pi_7}(T,3) = \mathtt{Nca}_{\Pi_7}((T,\{\}),3) &= \mathtt{Nca}_{\Pi_7}((T,\{\}) \vee (FT,\{\}) \vee (BD,\{\}) \vee (AD,\{\}),2) \\
&= (T,\{\}) \vee (FT,\{\}) \vee (BD,\{\}) \vee (AD,\{\}) \vee (BO,\{\}) \vee (AO,\{\}) \\
&= (T,\{\}) \vee (BD,\{\}) \vee (AD,\{\}) \vee (BO,\{\}) \vee (AO,\{\}) \,.
\end{aligned}$$

Again, the assumptions on the absence of T and D and on the presence of O in the initial multiset make the obtained pattern equivalent to $(B, \{\}) \vee (A, \{\})$ that allows us to say that any pattern that does not contain neither molecule A nor molecule B cannot cause the production of T.

For the same P systems Π_7, let us now compute the predictor for the presence of F after three steps. This actually should be a pattern characterising A^nB^n (that, as we have seen in Section 3, cannot be expressed by the current version of multiset patterns).

$$\begin{aligned}\mathtt{Sc}_{\Pi_7}(F,3) &= \mathtt{Sca}_{\Pi_7}((F,\{\}),3) = \mathtt{Sca}_{\Pi_7}((F,\{FT\}),2) = \mathtt{Sca}_{\Pi_7}((F,\{FT,FAD,FBD,FFT\}),1)\\ &= (F,\{FT,FAD,FBD,FFT,FAO,FBO\})\,.\end{aligned}$$

From the assumption on the absence of T and D in the initial multiset we have that the obtained pattern corresponds to $(F, \{FAO, FBO\})$. Moreover, from the assumption on the presence of F and O we can conclude that the pattern is actually satisfied only when $|w_0^7|_A = |w_0^7|_B = 0$. Hence, the pattern is a correct predictor (since A^0B^0 belongs to the A^nB^n language), but it does not capture all the initial multisets that would lead to the presence of F in three steps.

The pattern obtained by the proposed operator represents a sufficient condition for the presence of some molecules after a given number of steps. The last example shows that there are cases in which a complete condition (without false negatives) cannot be expressed with the current definition of multiset patterns. However, the limited expressiveness of multiset patterns is not the only reason for the incompleteness of the $\mathtt{Sc}_\Pi$ operator. Indeed, there are also cases in which the operator fails in computing a complete pattern, even if such a pattern could be expressed. We have shown this kind of situations in Example 4.6.

6.2. An Application to Gametogenesis

Lake frog (*Pelophylax ridibundus* Pallas, 1771) and pool frog (*Pelophylax lessonae* Camerano, 1882) can mate producing the hybrid edible frog (*Pelophylax esculentus* Linneus, 1758). The edible frog can coexist with one or both of the parental species giving rise to mixed populations. Usually the genotypes of *P. ridibundus*, *P. lessonae* and *P. esculentus* are indicated by *RR*, *LL*, and *LR*, respectively. In Europe there are mainly mixed populations containing *P. lessonae* and *P. esculentus*, called L-E systems. Hybrids in these populations reproduce in a particular way, called *hybridogenesis*. Hybridogenesis consists in a particular gametogenetic process in which the hybrids exclude one of their parental genomes, and transmit the other one, clonally, to eggs and sperm. This particular way of reproduction requires that hybrids live sympatrically with the parental species the genome of which is eliminated. In this way hybrids in a L-E system eliminate the L genome thus producing *P. esculentus* when mating with *P. lessonae*, and generating *P. ridibundus* when mating with other hybrids. Usually *P. ridibundus* generated in L-E complexes are inviable due to deleterious mutations accumulated in the clonally transmitted R genome. Because of inviability of *P. esculentus* $\times$ *P. esculentus* offspring, edible frog populations cannot survive alone, but they must act as a sexual parasite of the parental species *P. lessonae*. In L-E complexes, the reproductive pattern is the one described in the following table, where the subscribed y indicates the male sexual chromosome.

	LL	LR
L_yL	L_yL LL	L_yR LR
L_yR	LR	RR not viable

Note that the y chromosome, determining the sex of males, can occur only in the L genome, due to primary hybridization which involved, for size constraints, *P. lessonae* males and *P. ridibundus* females. The table shows that only one of the three possible matings resulting in viable offspring produce LL genotypes.

In [24, 25, 26] we studied the dynamics of frog populations by varying a set of biological parameters. Here we propose a simple example inspired by hybridogenesis in L-E complexes. We assume to have *P. esculentus* males and females, represented by $P_E^{\male}$ and $P_E^{\female}$, respectively, and *P. lessonae* males and females represented by $P_L^{\male}$ and $P_L^{\female}$. For the sake of simplicity we assume that each frog produces a single gamete. In particular, *P. esculentus* males produce $R^{\male}$ gametes, while *P. esculentus* females produce $R^{\female}$ gametes; *P. lessonae* males produce either $L^{\male}$ or $L_y^{\male}$ gametes, and *P. lessonae* females produce only $L^{\female}$ gametes. Gametes combine for producing the next generation of frogs.

The example is formalised by the P system $\Pi_8 = (V_{Frogs}, w_0^8, R_{Frogs})$ where V_{Frogs} contains all the molecules representing individual frogs and gametes, and the set R_{Frogs} consists of the following evolution rules:

$$\begin{array}{lllll}
P_E^{\male} \to R^{\male} & P_E^{\female} \to R^{\female} & P_L^{\male} \to L^{\male} & P_L^{\male} \to L_y^{\male} & P_L^{\female} \to L^{\female} \\
L^{\male} L^{\female} \to P_L^{\female} & L^{\male} R^{\female} \to P_E^{\female} & R^{\male} L^{\female} \to P_E^{\female} & L_y^{\male} L^{\female} \to P_L^{\male} & \\
L_y^{\male} R^{\female} \to P_E^{\male} & R^{\male} R^{\female} \to \emptyset & & &
\end{array}$$

We want to check the possibility to obtain a particular frog in two steps. We assume to start from an initial configuration containing only frogs. This will allow us to discard all patterns which consider the possibility to have gametes in the initial situation.

First we check whether there is an initial configuration which gives the certainty to obtain a *P. esculentus* male, $P_E^{\male}$, in two steps. We have:

$$\begin{aligned}
\mathtt{Sc}_{\Pi_8}(P_E^{\male}, 2) &= \mathtt{Sca}_{\Pi_8}((P_E^{\male}, \{\}), 2) = \mathtt{Sca}_{\Pi_8}((L_y^{\male} R^{\female}, \{L_y^{\male} L^{\female},\ R^{\male} R^{\female}\}), 1) \\
&= \mathtt{Sca}_{\Pi_8}((P_L^{\male}, \{P_L^{\male},\ P_L^{\male} P_L^{\female}, L_y^{\male} L^{\female},\ R^{\male} R^{\female}, P_E^{\male} P_E^{\female}\}) \wedge (P_E^{\female}, \{\ P_L^{\male} P_L^{\female}, L_y^{\male} L^{\female},\ R^{\male} R^{\female}, P_E^{\male} P_E^{\female}\}), 0) \\
&= \mathtt{false} \wedge (P_E^{\female}, \{\ P_E^{\male} P_E^{\female}\}) = \mathtt{false}
\end{aligned}$$

In the last step, the first operand of the $\wedge$ operator is always `false`. Thus there are no initial multiset of frogs which can ensure the production of a *P. esculentus* male. Recall that a *P. esculentus* male can be obtained only by a *P. lessonae* male producing the gamete $L_y^{\male}$. Unfortunately *P. lessonae* males could all produce the $L^{\male}$ gamete and no $L_y^{\male}$.

Then we compute the necessary predictor to detect the conditions that cannot surely allow us to obtain a *P. esculentus* male in two steps.

$$\mathtt{Nc}_{\Pi_8}(P_E^{\male},2) = \mathtt{Nca}_{\Pi_8}((P_E^{\male},\{\}),2) = \mathtt{Nca}_{\Pi_8}((L_y^{\male}R^{\female},\{\}),1) = \mathtt{Nca}_{\Pi_8}((P_L^{\male}P_E^{\female},\{\}),0) = (P_L^{\male}P_E^{\female},\{\})$$

This tells us that the only way a *P. esculentus* male can be obtained is starting with a *P. esculentus* female and *P. lessonae* male. Even if this will not ensure the production of a *P. esculentus* male in two steps, it is the only possibility: any other combination will surely not lead to the required product.

Let us now consider the production of a *P. esculentus* female, $P_E^{\female}$, in two steps:

$$\begin{aligned}
\mathtt{Sc}_{\Pi_8}(P_E^{\female},2) &= \mathtt{Sca}_{\Pi_8}((P_E^{\female},\{\}),2)\\
&= \mathtt{Sca}_{\Pi_8}((L^{\male}R^{\female},\{L^{\male}L^{\female},\ L_y^{\male}R^{\female},R^{\male}R^{\female}\}) \vee (R^{\male}L^{\female},\{L_y^{\male}L^{\female},\ L^{\male}L^{\female}\ R^{\male}R^{\female}\}),1)\\
&= \mathtt{Sca}_{\Pi_8}((L^{\male}R^{\female},\{L^{\male}L^{\female},\ L_y^{\male}R^{\female},R^{\male}R^{\female}\}),1) \vee \mathtt{Sca}_{\Pi_8}((R^{\male}L^{\female},\{L_y^{\male}L^{\female},L^{\male}L^{\female},R^{\male}R^{\female}\}),1)\\
&= \mathtt{Sca}_{\Pi_8}(((P_L^{\male},\{P_L^{\male},...\}) \wedge (P_E^{\female},\{\ ...\})),0)\\
&\quad \vee (\mathtt{Sca}_{\Pi_8}((P_E^{\male},\{P_L^{\male}P_L^{\female},P_L^{\male}P_E^{\female},P_E^{\male}P_E^{\female},L_y^{\male}L^{\female},L^{\male}L^{\female},\ R^{\male}R^{\female}\}),0)\\
&\qquad \wedge \mathtt{Sca}_{\Pi_8}((P_L^{\female},\{P_L^{\male}P_L^{\female},P_L^{\male}P_E^{\female},P_E^{\male}P_E^{\female},L_y^{\male}L^{\female},L^{\male}L^{\female},\ R^{\male}R^{\female}\}),0))\\
&= \ \mathtt{false} \vee (P_E^{\male},\{P_E^{\male}P_E^{\female}\}) \wedge (P_L^{\female},\{P_L^{\male}P_L^{\female}\})
\end{aligned}$$

The pattern $(P_E^{\male},\{P_E^{\male}P_E^{\female}\}) \wedge (P_L^{\female},\{P_L^{\male}P_L^{\female}\})$ is satisfied by multisets containing both more *P. esculentus* males than *P. esculentus* females and more *P. lessonae* females than *P. lessonae* males. For example an initial multiset containing two *P. esculentus* males, one *P. esculentus* female and two *P. lessonae* females will produce at least one *P. esculentus* female in two steps.

By computing the sufficient predictor we discover the conditions that cannot surely allow us to obtain a *P. esculentus* female in two steps:

$$\begin{aligned}
\mathtt{Nc}_{\Pi_8}(P_E^{\female},2) &= \mathtt{Nca}_{\Pi_8}((P_E^{\female},\{\}),2) = \mathtt{Nca}_{\Pi_8}(((L^{\male}R^{\female},\{\}) \vee (R^{\male}L^{\female},\{\})),1)\\
&= \mathtt{Nca}_{\Pi_8}((L^{\male}R^{\female},\{\}),1) \vee \mathtt{Nca}_{\Pi_8}((R^{\male}L^{\female},\{\}),1) = \mathtt{Nca}_{\Pi_8}((P_L^{\male}P_E^{\female},\{\ \}),0) \vee \mathtt{Nca}_{\Pi_8}((P_E^{\male}P_L^{\female},\{\}),0)\\
&= (P_L^{\male}P_E^{\female},\{\ \}) \vee (P_E^{\male}P_L^{\female},\{\})
\end{aligned}$$

This tells us that the different ways in which we could produce an *P. esculentus* male. If we do not have at least one *P. esculentus* male and one *P. lessonae* females or one *P. esculentus* male and one *P. lessonae* females we can not have any *P. esculentus* male in two steps.

6.3. An Application to Genetics

In cats, the red variants (red or cream) of coat color is due to a gene called O. The O allele of such a gene induces phaeomelanin (red pigment) to completely replaces eumelanin (black or brown pigment). The gene is a mutation of the wild o allele which codes for the non-red color (brown, black, blue, ...).

The O gene is sex-linked being located on the X chromosome. Males have only one X chromosome, so they have only one allele of this gene: O results in red color, and o results in non-red coat. Females have two X chromosomes, thus they have two alleles of the gene. OO results in either red or cream females, oo results in non-red females, and Oo results in tortoiseshell (or blue-cream when the color is diluted) cats, in which part of the coat color is determined by the O allele and part by the o one. The reason for this expression of alleles is $X -$ *inactivation*, a process by which one of the copies of the X chromosome is inactivated in cells. The choice of which X chromosome will be inactivated is random during the embryos development, but once an X chromosome is inactivated it will remain inactive throughout the lifetime of the cell and its descendants in the organism. Therefore, since Y is the males sex chromosome, X_O is a X chromosome with the O allele, and X_o is a X chromosome with the o allele, we have the following kinds of cats.

(YX_O)	red male
(YX_o)	non-red male
(X_OX_O)	red female
(X_OX_o)	tortoiseshell female
(X_oX_o)	non-red female

In the following table are depicted the possible offspring of cats with different genetics.

	(X_OX_O)	(X_OX_o)	(X_oX_o)
(YX_O)	(YX_O) (X_OX_O)	(YX_O) (YX_o) (X_OX_O) (X_OX_o)	(YX_o) (X_OX_o)
(YX_o)	(YX_O) (X_oX_O)	(YX_O) (YX_o) (X_OX_o) (X_oX_o)	(YX_o) (X_oX_o)

For example, a red male and a tortoiseshell female can produce either a red or non-red male or a red or tortoiseshell female.

The example is formalised by the P system $\Pi_9 = (V_{Cats}, w_0^9, R_{Cats})$ where V_{Cats} contains male and female cats with different colors, represented by the combination of the Y chromosome and the two different version of X chromosome, and the set R_{Cats} consists of the following evolution rules representing the different possibilities of the previous table:

$$\begin{array}{llll}
(YX_O)\ (X_OX_O) \to (YX_O) & (YX_O)\ (X_OX_O) \to (X_OX_O) & (YX_O)\ (X_OX_o) \to (YX_O) \\
(YX_O)\ (X_OX_o) \to (YX_o) & (YX_O)\ (X_OX_o) \to (X_OX_O) & (YX_O)\ (X_OX_o) \to (X_OX_o) \\
(YX_O)\ (X_oX_o) \to (YX_o) & (YX_O)\ (X_oX_o) \to (X_OX_o) & \\
\\
(YX_o)\ (X_OX_O) \to (YX_O) & (YX_o)\ (X_OX_O) \to (X_OX_o) & (YX_o)\ (X_OX_o) \to (YX_o) \\
(YX_o)\ (X_OX_o) \to (YX_O) & (YX_o)\ (X_OX_o) \to (X_OX_o) & (YX_o)\ (X_OX_o) \to (X_oX_o) \\
(YX_o)\ (X_oX_o) \to (YX_o) & (YX_o)\ (X_oX_o) \to (X_oX_o) &
\end{array}$$

Assume we are interested in having a red female after one, two or three generation. By computing the necessary predictor, we discover the conditions that may allow us to have a red female in one generation:

$$\begin{aligned}
\mathtt{Nc}_{\Pi_9}((X_OX_O),1) &= \mathtt{Nca}_{\Pi_9}(((X_OX_O),\{\}),1) = \mathtt{Nca}_{\Pi_9}((((YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_O),\{\})),0)\\
&= \mathtt{Nca}_{\Pi_9}((((YX_O)(X_OX_o),\{\}),0) \vee \mathtt{Nca}_{\Pi_9}((((YX_O)(X_OX_O),\{\})),0)\\
&= ((YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_O),\{\}).
\end{aligned}$$

The predictor tells us that the only possibility for having a red female after one generation is to start with a red male and either a tortoiseshell or red female. No other combination of parents can lead to the generation of a red female. Indeed, it is not possible to generate a red female starting either with a non-red male or with a female not having at least one O allele.

Computing the sufficient predictor for allowing us to have a red female in two generations, we obtain:

$$\begin{aligned}
\mathtt{Nc}_{\Pi_9}((X_OX_O),2) &= \mathtt{Nca}_{\Pi_9}((X_OX_O),\{\}),2) = \mathtt{Nca}_{\Pi_9}((((YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_O),\{\}),1)\\
&= \mathtt{Nca}_{\Pi_9}((((YX_O)(X_OX_o),\{\}),1) \vee \mathtt{Nca}_{\Pi_9}((((YX_O)(X_OX_O),\{\}),1)\\
&= \mathtt{Nca}_{\Pi_9}((((YX_O)(X_OX_O)(YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_O)(YX_O)(X_oX_o),\{\})\\
&\qquad \vee ((YX_O)(X_OX_O)(YX_o)(X_OX_O),\{\}) \vee ((YX_O)(X_OX_O)(YX_o)(X_OX_o),\{\})\\
&\qquad \vee ((YX_O)(X_OX_o)(YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_o)(YX_O)(X_oX_o),\{\})\\
&\qquad \vee ((YX_O)(X_OX_o)(YX_o)(X_OX_O),\{\}) \vee ((YX_O)(X_OX_o)(YX_o)(X_OX_o),\{\})\\
&\qquad \vee ((YX_o)(X_OX_o)(YX_O)(X_OX_o),\{\}) \vee ((YX_o)(X_OX_o)(YX_O)(X_oX_o),\{\})\\
&\qquad \vee ((YX_o)(X_OX_o)(YX_o)(X_OX_O),\{\}) \vee ((YX_o)(X_OX_o)(YX_o)(X_OX_o),\{\}),0) \vee\\
&\quad \mathtt{Nca}_{\Pi_9}((((YX_O)(X_OX_O)(YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_O)(YX_O)(X_OX_O),\{\})\\
&\qquad \vee (((YX_O)(X_OX_o)(YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_o)(YX_O)(X_OX_O),\{\})\\
&\qquad \vee ((YX_o)(X_OX_o)(YX_O)(X_OX_o),\{\}) \vee ((YX_o)(X_OX_o)(YX_O)(X_OX_O),\{\})\\
&\qquad \vee ((YX_O)(X_OX_O)(YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_O)(YX_O)(X_OX_O),\{\}),0)\\
&\quad = \quad ((YX_O)(X_OX_O)(YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_O)(YX_O)(X_oX_o),\{\})\\
&\qquad \vee ((YX_O)(X_OX_O)(YX_o)(X_OX_O),\{\}) \vee ((YX_O)(X_OX_O)(YX_o)(X_OX_o),\{\})\\
&\qquad \vee ((YX_O)(X_OX_o)(YX_O)(X_OX_o),\{\}) \vee ((YX_O)(X_OX_o)YX_O(X_oX_o),\{\})\\
&\qquad \vee ((YX_O)(X_OX_o)(YX_o)(X_OX_O),\{\}) \vee ((YX_O)(X_OX_o)(YX_o)(X_OX_o),\{\})\\
&\qquad \vee ((YX_o)(X_OX_o)(YX_O)(X_OX_o),\{\}) \vee ((YX_o)(X_OX_o)(YX_O)(X_oX_o),\{\})\\
&\qquad \vee ((YX_o)(X_OX_o)(YX_o)(X_OX_O),\{\}) \vee ((YX_o)(X_OX_o)(YX_o)(X_OX_o),\{\})\\
&\qquad \vee ((YX_O)(X_OX_O)(YX_O)(X_OX_O),\{\}) \vee ((YX_O)(X_OX_o)(YX_O)(X_OX_O),\{\})\\
&\qquad \vee ((YX_o)(X_OX_o)(YX_O)(X_OX_o),\{\}) \vee ((YX_o)(X_OX_o)(YX_O)(X_OX_O),\{\})
\end{aligned}$$

This new predictor tells us all the ways in which four cats can generate in two generations a red female. Since we have modelled a situation where each pair generates just one kitten, the conditions that we obtain says that four distinct (grandparents) cats are necessary. Actually, it is not mandatory that the cats are all distinct. For example, in the multiset $(YX_o)(X_OX_o)(YX_o)(X_OX_o)$ the cats involved can be just two, because one pair of cats can generate more than one kitten.

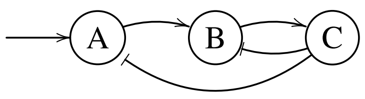

Figure 6: An example of gene regulatory network.

Moreover, the multiset $(YX_O)(X_OX_O)(YX_O)(X_oX_o)$ can represent only three cats, since the red male can be involved in two different matings.

If we observe two generations of cats, it is possible that non-red males generate a red female if they mate with red or tortoiseshell females (see the pattern $((YX_o)(X_OX_o)(YX_o)(X_OX_o), \{\})$). Indeed, a black male that mate with a tortoiseshell female can generate either a red male or a tortoiseshell female, which, in the next generation may generate a red female. Note that, it is not possible for a red male and a non-red female to generate a red female in two generations; in the predictor there is not a pattern of the form $((YX_O)(X_oX_o)(YX_O)(X_oX_o), \{\})$, but all patterns involving a pair of red male and a non-red female $((YX_O)(X_oX_o))$ also involve a pair of either black or red male and either tortoiseshell or red female (see for example the pattern $((YX_o)(X_OX_o)(YX_O)(X_oX_o), \{\})$).

If we ask for the sufficient predictor for having a red female in three generations we obtain a disjunction of multiset patterns expressing conditions on eight cats. It is easy to see that one member of the disjunction $\mathtt{Nc}_{\Pi_9}((X_OX_O), 3)$ is the pattern $((YX_O)(X_oX_o)(YX_O)(X_oX_o)(YX_O)(X_oX_o)(YX_O)(X_oX_o), \{\})$. Such pattern tells us that, finally after three generation, a red female can be obtained even starting from a pair of cats formed by a red male and a non-red female. It is worth noting that no sufficient predictor can be useful in this case. Due to the non-determinism, there are no conditions that assure us that a particular kitten is generated. Indeed, $\mathtt{Sc}_{\Pi_9}(cat, k) = \mathtt{false}$ for $cat \in \{(YX_o), (YX_O), (X_OX_O), (X_OX_o), (X_oX_o)\}$ and for all k. Intuitively this is because the same pair of cats can generate several kinds of kittens.

6.4. Application to Gene Regulatory Networks

Gene regulatory networks are networks of interactions among genes of the same cell aimed at regulating the activation of specific cell functionalities. More specifically, in a regulatory network each active gene can either stimulate or inhibit the activation of a number of other genes. Moreover, the activation of some genes is also usually influenced by other factors such as the availability of some substances in the environment or the reception of a signal form neighbour cells. As a consequence, gene regulatory networks can be seen as the mechanisms that allow a cell to react to external stimuli. When a stimulus is received, it usually changes the activation state of few genes. This causes, through the regulatory network, a chain of changes in the activation states of other genes, allowing a new configuration of active genes to be reached.

A gene regulatory network is often represented as a graph like the one in Fig. 6. Nodes of the graph represent genes and edges represent interaction. Arrows represent the way genes influence each other. An arrow form a node X to a node Y means that the gene represented by X influences the activation of the gene represented by Y. If the arrow is normal, it means that X stimulates the activation of Y. On the other hand, if the arrow is T-shaped, it means that X inhibits the activation of Y. What happens when the same activation of a gene is concurrently stimulated and inhibited by other genes can be different for different genes.

Gene regulatory networks can be modelled in many ways (see [27] for a survey on this topic). Modelling techniques can either deal with only the qualitative aspects of such networks (treating them essentially as logic circuits), or can describe also the quantitative aspects, such as the rates of the interactions. The latter approach is for sure more precise, but requires many additional details of the networks dynamics to be taken into account, such as the rates of transcription into RNA and the translation into proteins of each involved gene. Qualitative models are often sufficient to reason on the behaviour of the regulatory networks, although with some degree of approximation.

In the qualitative modelling setting, one of the most successful modelling frameworks for gene regulatory networks are boolean networks. The idea is that each gene is described as a boolean variable the value of which represents the gene's activation state. Moreover, each gene is associated with a boolean function used to compute the value of the variable in terms of the values of the variables associated with the other genes.

The same interpretation of gene regulatory networks used to model them as boolean networks may be used to construct models based on P systems. The gene network shown in Fig. 6 can be formalised by the P system $\Pi_{10} = (V_{10}, w_0^{10}, R_{10})$ where
$V_{10} = \{A, B, C, A_B, B_c, C_A, C_B, \overline{A}, \overline{B}, \overline{C}, \overline{A}_B, \overline{B}_c, \overline{C}_A, \overline{C}_B\}$ and the set R_{10} consists of the following evolution rules:

$$\begin{array}{lllll}
A \to A_B & \overline{A} \to \overline{A}_B & B \to B_C & \overline{B} \to \overline{B}_C & C \to C_B C_A \\
\overline{C} \to \overline{C}_B \overline{C}_A & \overline{C}_B A_B \to B & \overline{C}_B \overline{A}_B \to \overline{B} & C_B A_B \to \overline{B} & C_B \overline{A}_B \to \overline{B} \\
C_B \to \overline{A} \quad \overline{C}_B \to A & & B_C \to C & \overline{B}_C \to \overline{C} &
\end{array}$$

For any gene $X \in \{A, B, C\}$ with X we model the fact that the gene is active while $\overline{X}$ denote the fact that the gene X is inactive. Moreover, for any $X \in \{A, B, C, \overline{A}, \overline{B}, \overline{C}\}$ and $Y \in \{A, B, C\}$ we model the effect the active or inactive gene X has on gene Y. Note that, every real evolution step of the gene network corresponds to two steps of our P system. Moreover, multisets where $\overline{X}$ and X ($X \in \{A, B, C\}$) are both presents do not describe any real possible multiset. If we are interested in all initial multisets that can lead in one real step to the situation where all three genes are inactive we have to compute $\mathtt{Nc}_{\Pi_{10}}(\overline{ABC}, 2)$ as follows.

$$\begin{aligned}
\mathtt{Nc}_{\Pi_{10}}(\overline{ABC}, 2) &= \mathtt{Nca}_{\Pi_{10}}((\overline{ABC}\{\}), 2) = \mathtt{Nca}_{\Pi_{10}}(C_B C_B A_B \overline{B}_C, \{\}) \vee (C_B C_B \overline{A}_B \overline{B}_C\{\})), 1) \\
&= \mathtt{Nca}_{\Pi_{10}}(C_B C_B A_B \overline{B}_C, \{\}), 1) \vee \mathtt{Nca}_{\Pi_{10}}((C_B C_B \overline{A}_B \overline{B}_C\{\}), 1) \\
&= \mathtt{Nca}_{\Pi_{10}}(CCA\overline{B}, \{\}), 0) \vee \mathtt{Nca}_{\Pi_{10}}((CC\overline{AB}, \{\}), 0) \\
&= (CCA\overline{B}, \{\}) \vee (CC\overline{AB}, \{\})
\end{aligned}$$

The qualitative reading of these results tells us that for having all three genes inactive in one step we have to start with a configuration having gene C active and gene B inactive.

If we are interested in all initial multisets w_0^{10} that can lead in two real step to the situation where all three genes are active we have to compute $\mathtt{Nc}_{\Pi_{10}}(ABC, 4)$ as follows.

$$\begin{aligned}\mathtt{Nc}_{\Pi_{10}}(ABC, 4) &= \mathtt{Nca}_{\Pi_{10}}((ABC\{\}), 4) = \mathtt{Nca}_{\Pi_{10}}((\overline{C}_B\overline{C}_B A_B B_C, \{\}), 3)\\ &= \mathtt{Nca}_{\Pi_{10}}((AB\overline{CC}, \{\}), 2) = \mathtt{Nca}_{\Pi_{10}}((\overline{C}_B\overline{C}_B A_B\overline{B}_C\overline{B}_C, \{\}), 1)\\ &= \mathtt{Nca}_{\Pi_{10}}((A\overline{BBCC}, \{\})), 0) = (A\overline{BBCC}, \{\})\end{aligned}$$

Again, the qualitative reading of the result tells us that for having all three genes active after four steps we have to start with a initial configuration having gene A active and gene B and C inactive. Finally, note that since we are interested in a qualitative reading of this last biological example the sufficient predictor in this case does not model any interesting behaviour of the gene regulation network.

7. Conclusions and Further Developments

In this paper we studied dynamic causalities in membrane systems by defining the new notions of sufficient and necessary predictors. These are multiset patterns intended to characterise initial multisets that surely evolve in multiset containing molecules of interest in a given number of steps and initial multisets that surely will not evolve in multiset containing molecules of interest in a given number of steps. This information can be easily exploited to define causal relations. Indeed, in this paper we presented several biological examples that can be studied and better understood using our predictors. Moreover, another main contribution of this paper is the definition of multiset patterns. Multiset patterns have a finite representation but can characterise infinite sets of multisets such as languages. Multiset patterns seem, at this stage, the right issue to express predictors that can have several other applications e.g., to define a subclass of P systems with some important decidable properties.

Further developments of our work include the investigation of multiset patterns under the viewpoint of the multiset languages they characterise. Moreover, extensions of multiset patterns could be studied in order to enrich their expressiveness, this would be useful also to allow a new notion of predictors to be proposed which satisfies the completeness property (absence of false negatives).

References

[1] R. Gori, F. Levi, Abstract interpretation based verification of temporal properties for bioambients, Inf. Comput. 208 (8) (2010) 869–921.

[2] C. Bodei, R. Gori, F. Levi, An analysis for causal properties of membrane interactions, Electr. Notes Theor. Comput. Sci. 299 (2013) 15–31.

[3] C. Bodei, R. Gori, F. Levi, Causal static analysis for brane calculi, Theor. Comput. Sci. 587 (2015) 73–103.

[4] N. Busi, Causality in membrane systems, in: Membrane Computing, 8th International Workshop, WMC 2007, Thessaloniki, Greece, June 25-28, 2007 Revised Selected and Invited Papers, 2007, pp. 160–171. `doi:10.1007/978-3-540-77312-2_10`.

[5] R. Brijder, A. Ehrenfeucht, G. Rozenberg, A note on causalities in reaction systems, ECEASST 30.

[6] A. Ehrenfeucht, G. Rozenberg, Reaction systems, Fundamenta informaticae 75 (1-4) (2007) 263–280.

[7] R. Brijder, A. Ehrenfeucht, M. G. Main, G. Rozenberg, A tour of reaction systems, Int. J. Found. Comput. Sci. 22 (7) (2011) 1499–1517.

[8] R. Barbuti, R. Gori, F. Levi, P. Milazzo, Investigating dynamic causalities in reaction systems, Theoretical Computer Science 623 (2016) 114–145.

[9] R. Barbuti, R. Gori, F. Levi, P. Milazzo, Specialized predictor for reaction systems with context properties, Fundamenta Informaticae 147 (2-3) (2016) 173–191.

[10] R. Barbuti, R. Gori, F. Levi, P. Milazzo, Generalized contexts for reaction systems: definition and study of dynamic causalities, Acta Informatica (2017) 1–41In Press.

[11] G. Păun, Computing with membranes, Journal of Computer and System Sciences 61 (2000) 108–143.

[12] G. Păun, Membrane Computing: An Introduction, Natural Computing Series, Springer-Verlag GmbH, 2002.

[13] M. Boreale, D. Sangiorgi, A fully abstract semantics for causality in the pi-calculus, Acta Informatica 35 (5) (1998) 353–400.

[14] P. Degano, C. Priami, Enhanced operational semantics: a tool for describing and analyzing concurrent systems, ACM Computing Surveys (CSUR) 33 (2) (2001) 135–176.

[15] N. Busi, Towards a causal semantics for brane calculi, Proceedings of the Fifth Brainstorming Week on Membrane Computing, 97-111. Sevilla, ETS de Ingeniería Informática, 29 de Enero-2 de Febrero, 2007.

[16] H. R. Nielson, F. Nielson, H. Pilegaard, Spatial analysis of bioambients, in: International Static Analysis Symposium, Springer, 2004, pp. 69–83.

[17] F. Nielson, H. R. Nielson, C. Priami, D. Rosa, Control flow analysis for bioambients, Electronic Notes in Theoretical Computer Science 180 (3) (2007) 65–79.

[18] A. Alhazov, K. Morita, On reversibility and determinism in p systems, in: International Workshop on Membrane Computing, Springer, 2009, pp. 158–168.

[19] A. Alhazov, R. Freund, K. Morita, Sequential and maximally parallel multiset rewriting: reversibility and determinism, Natural Computing 11 (1) (2012) 95–106.

[20] G. M. Pinna, Reversing steps in membrane systems computations, in: International Conference on Membrane Computing, LNCS 10725, Springer, 2017, pp. 245–261.

[21] O. Agrigoroaiei, G. Ciobanu, Dual p systems, in: International Workshop on Membrane Computing, LNCS 5391, Springer, 2008, pp. 95–107.

[22] R. Barbuti, R. Gori, P. Milazzo, Multiset patterns and their application to dynamic causalities in membrane systems, in: Membrane Computing - 18th International Conference, CMC 2017, Bradford, UK, July 25-28, 2017, Revised Selected Papers, 2017, pp. 54–73.

[23] R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, S. Tini, Flat form preserving step-by-step behaviour, Fundamenta Informaticae 87 (2008) 1–34.

[24] P. Bove, P. Milazzo, R. Barbuti, The role of deleterious mutations in the stability of hybridogenetic water frog complexes, BMC evolutionary biology 14 (1) (2014) 1.

[25] R. Barbuti, P. Bove, P. Milazzo, G. Pardini, Minimal probabilistic P systems for modelling ecological systems, Theoretical Computer Science 608 (2015) 36–56.

[26] R. Barbuti, P. Bove, P. Milazzo, G. Pardini, Applications of P systems in population biology and ecology: The cases of MPP and APP systems, in: 17th International Conference on Membrane Computing (CMC17), LNCS 10105, Springer, 2016, pp. 28–48.

[27] T. Schlitt, A. Brazma, Current approaches to gene regulatory network modelling, BMC bioinformatics 8 (6) (2007) S9.