P Systems-Based Computing Polynomials With Integer Coefficients: Design and Formal Verification

Ming Zhu, Gexiang Zhang, *Member, IEEE*, Qiang Yang, Haina Rong, Weitao Yuan, and Mario J. Pérez-Jiménez

***Abstract* —Automatic design of mechanical procedures solving abstract problems is a relevant scientific challenge. In particular, automatic design of membranes systems performing some prefixed tasks is an important and useful research topic in the area of Natural Computing. In this context, deterministic membrane systems were designed in order to capture the values of polynomials with natural numbers coefficients. Following that work, this paper extends the previous result to polynomials with integer numbers coefficients. Specifically, a deterministic transition P system using priorities in the weak interpretation, associated with an arbitrary such kind polynomial, is presented. The configuration of the unique computation of the system will be encoded by means of two distinguished objects, the values of the polynomial for natural numbers. The descriptive computational resources required by the designed membrane system are also analyzed.**

***Index Terms*—Membrane computing, P systems, automatic design of membrane systems, polynomials with integer coefficients.**

I. Introduction

MEMBRANE computing is a rapidly growingbranch of natural computing initiated in [1], which abstracts computing models from the architecture and the functioning of living cells, as well as from the organization of cells in tissues, organs (the brain included), or other higher-degree structures. In the past twenty years, several classes of computing models (called P systems) were introduced, inspired from biological facts or motivated from mathematical or computer science points of view [2], [3]. Many P system classes are able to simulate register machines and therefore they are computationally complete, that is, they are equivalent in power to Turing machines [4]–[9]. It is well known that some P systems are efficient, in the sense that they have the ability to solve computationally hard problems by making use of an exponential workspace created in a natural way, in polynomial time [10]–[12]. Membrane computing models have been used in various applications like in the areas of approximate optimizations, systems and synthetic biology and real-life complex problems [13]–[19].

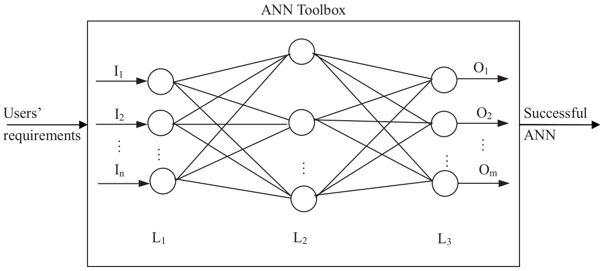

Fig. 1. Schematic graph showing the toolbox of artificial neural networks.

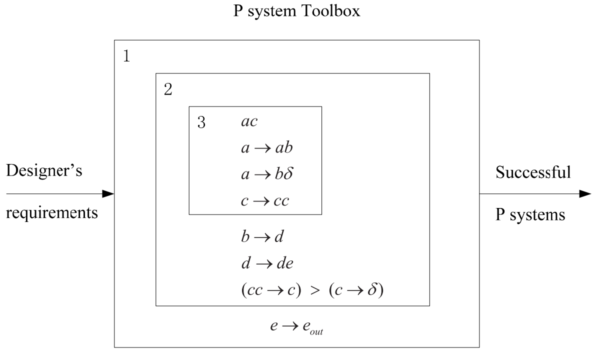

Fig. 2. Schematic graph showing the aim of automatic design of P systems.

Like the toolbox of artificial neural networks (ANN) for producing successful ANNs satisfying users' requirements, which is shown in Fig. 1, the automatic design of P systems is to develop a methodology for generating successful P systems meeting designers' requirements, as shown in Fig. 2.

This work was supported in part by the Scientific Research Fund of Sichuan Provincial Science and Technology Department under Grants 2015JY0257 and 2017FZ0010, in part by the National Nat-ural Science Foundation of China under Grants 61672437, 61702428, and 61373047, in part by the Sichuan Science and Technology Pro-gram under Grants 2018GZ0185, 2018GZ0085, and 2017GZ0159, and in part by the Fundamental Research Funds for the Central Universities under Grant A0920502051619-36. *(Corresponding author: Gexiang Zhang.)*

M. Zhu and Q. Yang are with the College of Control Engineering, Chengdu University of Information Technology, Chengdu 610225, China (e-mail: zhuming@cuit.edu.cn; yuxia2008@126.com).

G. Zhang, H. Rong, and W. Yuan are with the School of Electrical Engineering, Southwest Jiaotong University, Chengdu 610031, China (e-mail: zhgxdylan@126.com; ronghaina@126.com; 657279959@qq.com).

M. J. Pérez-Jiménez is with the Department of Computer Science and Artificial Intelligence, University of Sevilla, 41012 Sevilla, Spain (e-mail: marper@us.es).

This is a very complicated and challenging task. So far, the methods reported in the literature can be classified into two groups: heuristic and reasoning techniques [20]. The first type of methods focused on the use of heuristic algorithms, such as genetic algorthms (GAs) and quantum-inspired evolutionary algorithm (QIEAs), to make a population of P systems evolve toward a successful one [15]. This kind of methods began from the selection of an appropriate subset from a redundant set of evolution rules to design a cell-like P system, where a membrane structure and initial objects were pre-defined and fixed in the process of design [15], [21]–[24]. In [21], a genetic algorithm was employed to design a P system to calculate 4^2. In [22], a binary encoding technique was presented to denote an evolution rule set of a P system and a QIEA was used to make a population of P systems evolve toward successful ones. This method successfully solved the design of P systems to compute 4^2 and n^2 (for natural numbers $n \geq 2$). In [23], an evaluation approach considering non-determinism and halting penalty factors and a genetic algorithm with the binary encoding technique in [22] were introduced to design P systems for 4^2, n^2 and the generation of the language $\{a^{2^n}b^{3^n} | n > 1\}$. In these studies mentioned above, a specific redundant evolution rule set was designed for a specific computational task. This was developed in [15], [24] by applying one pre-defined redundant evolution rule set to design multiple different P systems, each of which executes a computation task. In [24], an automatic design method of a cell-like P system framework for performing five basic arithmetic operations (addition, subtraction, multiplication, division and power) was presented. In [15], a common redundant set of evolution rules was applied to design successful P systems for fulfilling eight computational tasks, i.e., eight computing sets of natural numbers: $2(n-1)$, $2n-1$, n^2, $\frac{1}{2}[n(n-1)]$, $n(n-1)$, $(n-1)^2+2^n+2$, $a^{2^n}b^{3^n}$ and $\frac{1}{2}(3^n-1)$, $(n > 1$ or $2)$. A significant development in this topic is the work in [25] in which a cell-like halting P system for 4^2 was designed by tuning membrane structures, initial objects and evolution rules. In that work, a genetic algorithm with a binary encoding technique was discussed to codify the three ingredients of a P system, the membrane structure, initial objects and evolution rules. Following this work, an automatic design method, *Permutation Penalty Genetic Algorithm* (PPGA), for a deterministic and non-halting membrane system by tuning membrane structures, initial objects and evolution rules was proposed in [26]. The main ideas of PPGA are the introduction of the permutation encoding technique for a membrane system, a penalty function evaluation approach for a candidate membrane system and a genetic algorithm for making a population of P systems evolve toward a successful one fulfilling a given computational task. A cell-like membrane system for computing the square of n^2 (for natural numbers $n \geq 1$) was successfully designed. In addition, the automatic design of the minimal membrane systems with respect to their membrane structures, alphabet, initial objects and evolution rules to fulfill the given task were also discussed in [26]. The second type of methods use reasoning techniques to fulfill the design of a P system. In [27], a reasoning method to design a k-degree ($k \geq 2$) polynomial P system was reported by analyzing the syntax and semantics of cell-like P systems.

In the study of [27], deterministic transition P systems for computing polynomials with natural number coefficients were designed. The numerical values of such polynomials, $p(n)$, for $n \in \mathbb{N}$, are always positive and the P systems computing $p(n)$ handle only positive numbers through the multiplicity of objects in an usual manner. In this paper, the work in [27] is extended to consider the design of deterministic transition P systems for computing polynomials with integer coefficients, where the numerical values of $p(n)$, for $n \in \mathbb{N}$, may be positive or negative and, consequently, the P systems computing $p(n)$ must process integer numbers by using natural numbers in the multiplicity of objects. This task is much more challenging.

The aim of this paper is to find a "minimal" such P system computing an arbitrary polynomial with integer coefficients. Here the concept "minimal" refers to some syntactical ingredients associated with P systems: the membrane structure has only one membrane and the number of objects used is very restrictive.

The rest parts of this paper are organized as follows. Section II recalls some preliminaries needed in the following sections, including the specific variant of membrane systems considered in this work. The main concept of polynomial with integer coefficients computed by a deterministic transition P system is defined in Section III. The design and formal verification of a deterministic P system associated with an arbitrary polynomial whose coefficients are integer numbers, is presented in Section III-A. The descriptive computational resources required by the designed k-degree polynomial P system is analyzed in Section IV. The comparison with metaheuristic approaches is discussed in Section V. Finally, conclusions and future work are given in Section VI.

II. Preliminaries

In this section, some general concepts are briefly described in order to make the work self-contained.

A. Alphabet and Multisets

An *alphabet* Γ is a non-empty set and their elements are called *symbols*. A *string* u over Γ is an ordered finite sequence of symbols, that is, a mapping from a natural number $n \in \mathbb{N}$ onto Γ. The number n is called the *length* of the string u and it is denoted by $|u|$. The empty string (with length 0) is denoted by λ. A *multiset* over an alphabet Γ is a mapping f from Γ onto the set of natural numbers $\mathbb{N}$. For each symbol $a \in \Gamma$, the natural number $f(a)$ is called the *multiplicity* of symbol a in multiset f. We denote by $M(\Gamma)$ the set of all multisets over Γ.

B. Rooted Tree

An *undirected graph* G is an ordered pair (V, E), where V is a set whose elements are called *nodes* and $E = \{\{x, y\} \mid x, y \in V,\ x \neq y\}$ whose elements are called *edges*. A *path* of length $k \geq 1$ from $x \in V$ to $y \in V$ is a sequence $(x_0, \dots, x_k)$

such that $x_0 = x$ and $x_k = y$. If $x_0 = x_k$ then we say that the path is a *cycle*. An undirected graph is *connected* if every pair of nodes is connected by a path. An undirected graph with no cycle is said to be *acyclic*. A *rooted tree* is a connected, acyclic, undirected graph in which one of the vertices (called *the root of the tree*) is distinguished from the others.

C. Transition P Systems

The basic model of membrane systems was introduced by Gh. Păun in its seminal paper [1]. A *transition P system* of degree $q \geq 1$ is a tuple

$\Pi = (\Gamma, \mu, \mathcal{M}_1, \dots, \mathcal{M}_q, (\mathcal{R}_1, \rho_1), \dots, (\mathcal{R}_q, \rho_q), i_{out})$, where:

- Γ is a finite alphabet.
- μ is a rooted tree.
- $\mathcal{M}_1, \dots, \mathcal{M}_q$ are multisets over Γ.
- $\mathcal{R}_i$, $1 \leq i \leq q$, is a finite set of evolution rules of the following forms: (a) $[\, u\,]_i \rightarrow v_1 [\, v_2 [\, v_3\,]_j\,]_i$; and (b) $[\, u\,]_i \rightarrow v_1 [\, v_2 [\, v_3\,]_j\,]_i\, \delta$, where $i, j \in \{1, \dots, q\}$, $i \neq j$, $u, v_1, v_2, v_3 \in M(\Gamma)$ and δ is a distinguished symbol such that $\delta \notin \Gamma$.
- ρ_i, $1 \leq i \leq q$, is an strict partial order over $\mathcal{R}_i$.
- $i_{out} \in \{0, 1, \dots, q\}$.

A *transition P system* $\Pi = (\Gamma, \mu, \mathcal{M}_1, \dots, \mathcal{M}_q, (\mathcal{R}_1, \rho_1), \dots, (\mathcal{R}_q, \rho_q), i_{out})$, of degree $q \geq 1$ can be viewed as a set of q membranes injectively labeled by $1, \dots, q$, arranged in a hierarchical structure μ given by a rooted tree whose root is called the *skin membrane* of the system, and with an environment labeled by 0 such that: (a) $\mathcal{M}_1, \dots, \mathcal{M}_q$ are multisets over the *working alphabet* Γ representing the objects initially placed in the q membranes of the system; (b) $\mathcal{R}_i$, $1 \leq i \leq n$, is the set of rules associated with membrane i, and ρ_i provides priorities between rules in $\mathcal{R}_i$, in such a manner that if $(r_1, r_2) \in \rho_i$ we say that rule r_1 has a higher priority than r_2 and we denote it by $r_1 > r_2$; and (c) $i_{out} \in \{1, \dots, q\}$ represents a distinguished membrane (the *output membrane*).

A *configuration* at an instant t of a transition P system is described by the membrane structure at instant t and all multisets of objects over Γ associated with all the membranes present in the system. The *initial configuration* of the system is $(\mu, \mathcal{M}_1, \cdots, \mathcal{M}_q)$. Given a transition P system Π, we say that configuration $\mathcal{C}_t$ yields configuration $\mathcal{C}_{t+1}$ in one *transition step*, if we can pass from $\mathcal{C}_t$ to $\mathcal{C}_{t+1}$ by applying the rules from $\mathcal{R}_1, \dots, \mathcal{R}_q$ synchronously, in a *non-deterministic maximally parallel manner*. This means the following: the objects to evolve in a transition step and the rules by which they evolve are chosen in a non-deterministic manner, but in such a way that in each membrane we have a maximally parallel application of rules (at each transition step a multiset of rules which is maximal is applied, no further applicable rule can be added). A *computation* of Π is a (finite or infinite) sequence of configurations such that: (a) the first term of the sequence is the initial configuration of the system; (b) each non-first term of the sequence is obtained from the previous configuration by applying rules of the system in a non-deterministic maximally parallel manner; and (c) if the sequence is finite then the last term of the sequence is a configuration, where no rule of the system is applicable to it.

It is worth pointing out that in this paper the priority between rules is used in the *weak interpretation*, that is, in a transition step a rule is used always when objects exist, which were not used by a rule of a higher priority. In this paper we deal with *deterministic* transition P systems, where there is only one computation starting from an initial configuration. Besides, only rules of the type $[\, u\,]_i \rightarrow [\, v\,]_i$ will be used and they are briefly denoted by $u \rightarrow v$ when the membrane being worked with is understood.

Let us consider two auxiliary functions f_+ and f_- from the set of integer numbers $\mathbb{Z}$ into the set of natural numbers $\mathbb{N}$, defined as follows:

$$f_+(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad f_-(x) = \begin{cases} 0 & \text{if } x \geq 0 \\ -x & \text{if } x < 0 \end{cases}$$

It is worth pointing out that for each integer number $x \in \mathbb{Z}$ we have $f_+(x) \geq 0$, $f_-(x) \geq 0$, $f_+(x) + f_-(x) = |x|$ and $f_+(x) - f_-(x) = x$.

III. Deterministic Transition P Systems Computing Polynomials With Integer Coefficients

In this section we define the meaning of computing a polynomial $p(n)$ whose coefficients are integer numbers, by a deterministic transition P system $\Pi_{p(n)}$ associated with it. The idea is the following: for each natural number $t \in \mathbb{N}$ the value $p(t)$ will be computed/encoded by the configuration $\mathcal{C}_{t+1}$ of the unique computation of $\Pi_{p(n)}$. For that, four distinguished objects ($\mathbf{o}_1, \mathbf{o}_2, p_1, p_2$) will be considered in the working alphabet of $\Pi_{p(n)}$, in such a manner that $\mathbf{o}_1, \mathbf{o}_2$ will be used to encode/represent integer numbers by means of their multiplicities, and p_1, p_2 will be used as their corresponding transition computing objects.

Definition 1: Let $p(n)$ be a polynomial with integer numbers coefficients. We say that $p(n)$ is computed by a deterministic transition P system

$$\Pi_{p(n)} = (\Gamma, \mu, \mathcal{M}_1, \dots, \mathcal{M}_q, (\mathcal{R}_1, \rho_1), \dots, (\mathcal{R}_q, \rho_q), i_{out})$$

if the following holds:

- The working alphabet Γ has four distinguished objects: $\mathbf{o}_1$, $\mathbf{o}_2$ (the output objects) and p_1, p_2 (transition computing objects).
- For each $t \in \mathbb{N}$, at configuration $\mathcal{C}_{t+1}$ the content of the output membrane labeled by i_{out} encodes the value $p(t)$ through the multiplicity of objects $\mathbf{o}_1$ and $\mathbf{o}_2$ as follows: (a) If $p(t) \geq 0$ then the multiplicity of $\mathbf{o}_\mathbf{1}$ is $p(t)$ and the multiplicity of $\mathbf{o}_\mathbf{2}$ is 0; and (b) if $p(t) < 0$ then the multiplicity of $\mathbf{o}_\mathbf{2}$ is $-p(t)$ and the multiplicity of $\mathbf{o}_\mathbf{1}$ is 0.

A. Design

In this section, a deterministic transition P system $\Pi_{p(n)}$ of degree 1 that computes, in the sense of Definition 1, the polynomial $p(n) = a_0 + a_1 \cdot n \cdots + a_k \cdot n^k$ of degree $k \geq 1$, with integer coefficients $a_i \in \mathbb{Z}, 0 \leq i \leq k$, is designed.

It is easy to check that for each natural number $t \in \mathbb{N}$ the following holds:

$$
\begin{aligned}
& p(t+1) - p(t) \\
& = [a_1\binom{1}{0} + a_2\binom{2}{0} + \cdots + a_{k-1}\binom{k-1}{0} + a_k\binom{k}{0}] \cdot t^0 \\
& \quad + [a_2\binom{2}{1} + \cdots + a_{k-1}\binom{k-1}{1} + a_k\binom{k}{1}] \cdot t^1 \\
& \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
& \quad + [a_{k-1}\binom{k-1}{k-2} + a_k\binom{k}{k-2}] \cdot t^{k-2} \\
& \quad + [a_k\binom{k}{k-1}] \cdot t^{k-1}
\end{aligned}
$$

Let us denote:

$$
\begin{aligned}
a_k^0 &= a_1\binom{1}{0} + a_2\binom{2}{0} + \cdots + a_{k-1}\binom{k-1}{0} + a_k\binom{k}{0} \\
a_k^1 &= a_2\binom{2}{1} + \cdots + a_{k-1}\binom{k-1}{1} + a_k\binom{k}{1} \\
& \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
a_k^{k-2} &= a_{k-1}\binom{k-1}{k-2} + a_k\binom{k}{k-2} \\
a_k^{k-1} &= a_k\binom{k}{k-1}
\end{aligned}
$$

Then, $p(t+1) - p(t) = a_k^0 + a_k^1 \cdot t + a_k^2 \cdot t^2 + \cdots + a_k^{k-2} \cdot t^{k-2} + a_k^{k-1} \cdot t^{k-1} = \sum_{i=0}^{k-1} a_k^i \cdot t^i$, that is, $p(t+1) = p(t) + \sum_{i=0}^{k-1} a_k^i \cdot t^i$.

Definition 2: Let $p(n) = a_0 + a_1 \cdot n + \cdots + a_k \cdot n^k$ be a polynomial of degree $k \geq 1$, with integer coefficients $a_i \in \mathbb{Z}, 0 \leq i \leq k$. We associate $p(n)$ with the deterministic transition P system $\Pi_{p(n)} = (\Gamma, \mu, \mathcal{M}_1, (\mathcal{R}_1, \rho_1), i_{out})$ of degree 1, defined as follows:

- $\Gamma = \{\mathbf{o}_1, \mathbf{o}_2, p_1, p_2, b_1, b_2, b_3, \cdots b_k\}$
- $\mu = [\quad]_1$
- $\mathcal{M}_1 = \left\{\mathbf{o}_1^{f_+(a_0)} \mathbf{o}_2^{f_-(a_0)} b_1\right\}$
- $\mathcal{R}_1$ is the set of the following evolution rules:

$$
\begin{aligned}
r_1 &\equiv b_1 \rightarrow p_1^{f_+(a_k^0)} p_2^{f_-(a_k^0)} b_1^{\binom{0}{0}} b_2^{\binom{1}{0}} b_3^{\binom{2}{0}} \cdots b_{k-1}^{\binom{k-2}{0}} b_k^{\binom{k-1}{0}} \\
r_2 &\equiv b_2 \rightarrow p_1^{f_+(a_k^1)} p_2^{f_-(a_k^1)} b_2^{\binom{1}{1}} b_3^{\binom{2}{1}} \cdots b_{k-1}^{\binom{k-2}{1}} b_k^{\binom{k-1}{1}} \\
r_3 &\equiv b_3 \rightarrow p_1^{f_+(a_k^2)} p_2^{f_-(a_k^2)} b_3^{\binom{2}{2}} \cdots b_{k-1}^{\binom{k-2}{2}} b_k^{\binom{k-1}{2}} \\
& \vdots \\
- \quad r_{k-1} &\equiv b_{k-1} \rightarrow p_1^{f_+(a_k^{k-2})} p_2^{f_-(a_k^{k-2})} b_{k-1}^{\binom{k-2}{k-2}} b_k^{\binom{k-1}{k-2}} \\
r_k &\equiv b_k \rightarrow p_1^{f_+(a_k^{k-1})} p_2^{f_-(a_k^{k-1})} b_k^{\binom{k-1}{k-1}} \\
r_{k+1} &\equiv p_1 p_2 \rightarrow \lambda \\
r_{k+2} &\equiv p_1 \mathbf{o}_2 \rightarrow \lambda \\
r_{k+3} &\equiv p_2 \mathbf{o}_1 \rightarrow \lambda \\
r_{k+4} &\equiv p_1 \rightarrow \mathbf{o}_1 \\
r_{k+5} &\equiv p_2 \rightarrow \mathbf{o}_2
\end{aligned}
$$

- ρ_1 is the set of priorities relation among rules in $\mathcal{R}_1$: $\{(r_{k+1}, r_{k+2}), (r_{k+1}, r_{k+3}), (r_{k+2}, r_{k+4}), (r_{k+2}, r_{k+5}), (r_{k+3}, r_{k+4}), (r_{k+3}, r_{k+5})\}$ which can be informally described as: $r_{k+1} > \{r_{k+2}, r_{k+3}\} > \{r_{k+4}, r_{k+5}\}$.
- $i_{out} = 1$.

B. Formal Verification

We show in this subsection that the membrane system $\Pi_{p(n)}$ associated with the polynomial $p(n)$, designed in the previous section, computes the values $p(t)$ according to Definition 1, for each $t \in \mathbb{N}$.

Theorem 1: Let $p(n) = a_0 + a_1 \cdot n + \cdots + a_k \cdot n^k$ be a polynomial of degree $k \geq 1$ such that $a_i \in \mathbb{Z}, 0 \leq i \leq k$. Let $\Pi_{p(n)}$ be the deterministic transition P system considered in Definition 1. For each $t \geq 0$, at configuration $\mathcal{C}_{t+1}$ the content of membrane labeled by 1 is the following multiset:

$$
\{\mathbf{o}_1^{f_+(p(t))} \mathbf{o}_2^{f_-(p(t))} p_1^{\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i} p_2^{\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i} \\
b_1 \; b_2^{(t+1)} \; b_3^{(t+1)^2} \cdots \; b_k^{(t+1)^{k-1}}\}
$$

Proof: Let us prove the result by induction on t. Let us start with the base case $t = 0$. At the initial configuration $\mathcal{C}_0$, the content of membrane labeled by 1 is the multiset $\left\{\mathbf{o}_1^{f_+(a_0)} \mathbf{o}_2^{f_-(a_0)} b_1\right\}$. Then, configuration $\mathcal{C}_0$ yields configuration $\mathcal{C}_1$ by applying rule r_1 once. Thus, at configuration $\mathcal{C}_1$ the content of membrane labeled by 1 is the multiset $\left\{\mathbf{o}_1^{f_+(a_0)} \mathbf{o}_2^{f_-(a_0)} p_1^{f_+(a_k^0)} p_2^{f_-(a_k^0)} b_1 \; b_2 \; b_3 \cdots \; b_k\right\}$. Because of $p(0) = a_0 = f_+(a_0) - f_-(a_0)$, the result holds for $t = 0$.

By induction hypothesis, let us assume the result holds for $t \geq 0$, that is, at configuration $\mathcal{C}_{t+1}$ the content of membrane labeled by 1 is the multiset

$$
\{\mathbf{o}_1^{f_+(p(t))} \mathbf{o}_2^{f_-(p(t))} p_1^{\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i} p_2^{\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i} \\
b_1 \; b_2^{(t+1)} \; b_3^{(t+1)^2} \cdots \; b_k^{(t+1)^{k-1}}\}.
$$

In order to obtain the content of membrane labeled by 1 at configuration $\mathcal{C}_{t+2}$, let us analyze all the possible cases that may happen:

Case 1: $\boldsymbol{\sum_{i=0}^{k-1} a_k^i \cdot t^i \geq f_-(p(t))}$

In this case, $p(t+1) - p(t) \geq f_-(p(t))$ and the following holds:

(a) $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i \geq f_-(p(t))$. Indeed, it is sufficient to notice that $a_k^i = f_+(a_k^i) - f_-(a_k^i)$, for $0 \leq i \leq k-1$.

(b) $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i \geq \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i$. Indeed, (b) follows from (a) recalling that $f_-(p(t)) \geq 0$.

(c) $p(t+1) \geq 0$. Indeed, if $p(t) \geq 0$ then $p(t+1) \geq p(t) + f_-(p(t)) \geq 0$, and if $p(t) < 0$ then $f_-(p(t)) = -p(t)$, so $p(t+1) \geq p(t) + f_-(p(t)) = 0$.

Therefore, in this case configuration $\mathcal{C}_{t+1}$ yields configuration $\mathcal{C}_{t+2}$ as follows:

(1) From (b) we deduce that at configuration $\mathcal{C}_{t+1}$ in membrane labeled by 1 there are more copies of object p_1

than copies of object p_2, so rule $r_{k+1} \equiv \ p_1 p_2 \to \lambda$ will be applied $\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i$ times, consuming all copies of p_2 and "remaining" $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i$ copies of p_1 without evolving.

(2) From (a) we deduce that at configuration $\mathcal{C}_{t+1}$ in membrane labeled by 1 there are more copies of the "remaining" object p_1 than copies of object $\mathbf{o_2}$, so rule $r_{k+2} \equiv \ p_1\mathbf{o}_2 \to \lambda$ will be applied $f_-(p(t))$ times, consuming all copies of $\mathbf{o_2}$ and "remaining"

$$\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i - f_-(p(t))$$

copies of p_1 without evolving.

(3) Rule $r_{k+4} \equiv \ p_1 \to \mathbf{o}_1$ will be applied

$$\alpha = \sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i - f_-(p(t))$$

times, consuming all copies of p_1 and producing α copies of $\mathbf{o_1}$.

(4) For each j, $1 \leq j \leq k$, rule $r_j \equiv \ b_j \to p_1^{f_+(a_k^{j-1})} p_2^{f_-(a_k^{j-1})} b_j^{\binom{j-1}{j-1}} \ldots b_k^{\binom{k-1}{j-1}}$ will be applied $(t+1)^{j-1}$ times.

All the previous rules are applied in parallel in one transition step. Thus, in this case, at configuration $\mathcal{C}_{t+2}$ the content of membrane labeled by 1 is the multiset which contains objects $\mathbf{o}_1, \mathbf{o}_2, p_1, p_2$ and b_j, $(1 \leq j \leq k)$ with the following multiplicities:

- Multiplicity of $\mathbf{o}_1$: $f_+(p(t+1))$.
  Indeed, after execution of rules from (2), α new copies of $\mathbf{o}_1$ are produced. Thus, the total number of copies of $\mathbf{o}_1$ will be:

$$f_+(p(t)) + \sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i - f_-(p(t))$$
$$= p(t) + \sum_{i=0}^{k-1} a_k^i \cdot t^i = p(t+1) \overset{(c)}{=} f_+(p(t+1)).$$

- Multiplicity of $\mathbf{o}_2$: 0.
  Indeed, after execution of rules from (2), all copies of object $\mathbf{o}_2$ are consumed and any new copies of object $\mathbf{o}_2$ are produced for any rule applied in this transition step. Thus, at configuration $\mathcal{C}_{t+2}$ in membrane labeled by 1 the multiplicity of $\mathbf{o}_2$ is 0.
- Multiplicity of p_1: $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot (t+1)^i$.
  Indeed, after execution of rules from (1), (2) and (3), all copies of object p_1 are consumed but by applying rules from (4), the total number of copies of p_1 produced is

$$\sum_{j=1}^{k} f_+(a_k^{j-1}) \cdot (t+1)^{j-1} = \sum_{i=0}^{k-1} f_+(a_k^i) \cdot (t+1)^i.$$

- Multiplicity of p_2: $\sum_{i=0}^{k-1} f_-(a_k^i) \cdot (t+1)^i$.
  Indeed, after execution of rules from (1), (2) and (3), all copies of object p_2 are consumed but by applying rules from (4), the total number of copies of p_2 produced is

$$\sum_{j=1}^{k} f_-(a_k^{j-1}) \cdot (t+1)^{j-1} = \sum_{i=0}^{k-1} f_-(a_k^i) \cdot (t+1)^i.$$

- Multiplicity of object b_j, for each j, $1 \leq j \leq k$: $(t+2)^{j-1}$.
  Indeed, from (3) the total number of copies of object b_j produced is

$$\sum_{s=0}^{j-1} \binom{j-1}{s} (t+1)^s = [(t+1)+1]^{j-1} = (t+2)^{j-1}$$

Hence, in this case the result holds for $t+1$.

Case 2: $\mathbf{0} \leq \sum_{i=0}^{k-1} \boldsymbol{a_k^i \cdot t^i} < \boldsymbol{f_-(p(t))}$

In this case, the following holds:

(a) $p(t) < 0$ and $p(t+1) < 0$. Indeed, on the one hand, as $f_-(p(t)) > 0$ we have $f_-(p(t)) = -p(t)$. On the other hand,

$$p(t+1) = p(t) + \sum_{i=0}^{k-1} a_k^i \cdot t^i = -f_-(p(t)) + \sum_{i=0}^{k-1} a_k^i < 0.$$

(b) $0 \leq \sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i < f_-(p(t))$. Indeed, it suffices to notice that $a_k^i = f_+(a_k^i) - f_-(a_k^i)$, for $0 \leq i \leq k-1$.

(c) $p(t+1) = \left[\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i\right] - f_-(p(t))$. Indeed, as $f_-(p(t)) > 0$ we have $p(t) < 0$ and $f_-(p(t)) = -p(t)$. So,

$$p(t+1) = p(t) + \sum_{i=0}^{k-1} a_k^i \cdot t^i$$
$$= -f_-(p(t)) + \sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i.$$

Therefore, in this case configuration $\mathcal{C}_{t+1}$ yields configuration $\mathcal{C}_{t+2}$ as follows:

(1) From (b) we deduce that at configuration $\mathcal{C}_{t+1}$ in membrane labeled by 1 there are more copies of object p_1 than copies of object p_2, so rule $r_{k+1} \equiv \ p_1 p_2 \to \lambda$ will be applied $\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i$ times, consuming all copies of p_2 and "remaining" $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i$ copies of p_1 without evolving.

(2) Configuration $\mathcal{C}_{t+1}$ in membrane labeled by 1 there are more copies of object $\mathbf{o}_2$ than copies of object p_1, then rule $r_{k+2} \equiv \ p_1\mathbf{o}_2 \to \lambda$ will be applied $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i$ times,

consuming all copies of p_1 and "remaining" $f_-(p(t)) - \left[\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i\right]$ copies of $\mathbf{o}_2$ without evolving.

(3) For each j, $1 \le j \le k$, rule $r_j \equiv b_j \to p_1^{f_+(a_k^{j-1})} p_2^{f_-(a_k^{j-1})} b_j^{\binom{j-1}{j-1}} \dots b_k^{\binom{k-1}{j-1}}$ will be applied $(t+1)^{j-1}$ times.

All the previous rules are applied in parallel in one transition step. Thus, in this case, at configuration $\mathcal{C}_{t+2}$ the content of membrane labeled by 1 is the multiset which contains objects $\mathbf{o}_1, \mathbf{o}_2, p_1, p_2$ and b_j, $(1 \le j \le k)$ with the following multiplicities:

- Multiplicity of object $\mathbf{o}_1$: $f_+(p(t+1))$.
  Indeed, the applied rules do not affect to object $\mathbf{o}_1$ and from (a) we deduce that $f_+(p(t+1)) = 0 = f_+(p(t))$.
- Multiplicity of object $\mathbf{o}_2$: $f_-(p(t+1))$.
  Indeed, after execution of the cited rules, the multiplicity of $\mathbf{o}_2$ is
  $$f_-(p(t)) - \left[\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i\right] \overset{(b)}{=} -p(t+1) \overset{(c)}{=} f_-(p(t+1)).$$
- Multiplicity of object p_1: $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot (t+1)^i$.
  Indeed, after execution of the rules from (1) and (2), the multiplicity of p_1 is 0, but after the application of rules from (3) its multiplicity becomes
  $$\sum_{i=1}^{k} f_+(a_k^{i-1}) \cdot (t+1)^{i-1} = \sum_{i=0}^{k-1} f_+(a_k^i) \cdot (t+1)^i$$
- Multiplicity of object p_2: $\sum_{i=0}^{k-1} f_-(a_k^i) \cdot (t+1)^i$.
  Indeed, because after execution of the rules from (1) and (2), the multiplicity of p_2 is 0, but after the application of rules from (3) its multiplicity becomes
  $$\sum_{i=1}^{k} f_-(a_k^{i-1}) \cdot (t+1)^{i-1} = \sum_{i=0}^{k-1} f_-(a_k^i) \cdot (t+1)^i$$
- Multiplicity of object b_j, for each j, $1 \le j \le k$: $(t+2)^{j-1}$.
  Indeed, from (3) the total number of copies of object b_j produced is
  $$\sum_{s=0}^{j-1} \binom{j-1}{s} (t+1)^s = [(t+1)+1]^{j-1} = (t+2)^{j-1}$$

Hence, in this case the result holds for $t+1$.

Case 3: $\left[\boldsymbol{\sum_{i=0}^{k-1} a_k^i \cdot t^i < 0}\right] \wedge \left[\boldsymbol{f_+(p(t)) + \sum_{i=0}^{k-1} a_k^i \cdot t^i \le 0}\right]$

In this case, the following holds:

(a) $\sum_{i=0}^{k-1} f_+(a_k^i) - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i < 0$.

Indeed, it suffices to bear in mind that $\sum_{i=0}^{k-1} a_k^i \cdot t^i < 0$, and $a_k^i = f_+(a_k^i) - f_-(a_k^i)$, for $0 \le i \le k-1$.

(b) $f_+(p(t)) \le -\left[\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i\right]$

Indeed, it is enough to notice that
$$\begin{aligned} f_+(p(t)) &\le -\sum_{i=0}^{k-1} a_k^i \cdot t^i \\ &= -\left[\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i\right] \end{aligned}$$

(c) $p(t+1) \le 0$.

Indeed, if $p(t) \ge 0$ then $p(t) = f_+(p(t)) \le -\sum_{i=0}^{k-1} a_k^i \cdot t^i$ and $p(t+1) = p(t) + \sum_{i=0}^{k-1} a_k^i \cdot t^i$; if $p(t) < 0$ then $p(t+1) = p(t) + \sum_{i=0}^{k-1} a_k^i \cdot t^i < 0$.

Therefore, in this case configuration $\mathcal{C}_{t+1}$ yields configuration $\mathcal{C}_{t+2}$ as follows:

(1) From (a) we deduce that at configuration $\mathcal{C}_{t+1}$ in membrane labeled by 1 there are more copies of object p_2 than copies of object p_1, so rule $r_{k+1} \equiv p_1 p_2 \to \lambda$ will be applied $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i$ times, consuming all copies of p_1 and "remaining" $\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i$ copies of p_2 without evolving.

(2) From (b) we deduce that at configuration $\mathcal{C}_{t+1}$ in membrane labeled by 1 there are more copies of object p_2 than copies of object $\mathbf{o}_1$, so rule $r_{k+3} \equiv p_2 \mathbf{o}_1 \to \lambda$ will be applied $f_+(p(t))$ times, consuming all copies of $\mathbf{o}_1$ and "remaining" $-\left[f_+(p(t)) + \sum_{i=0}^{k-1} a_k^i \cdot t^i\right]$ copies of p_2 without evolve but these copies must evolve by means of rule r_{k+5}.

(3) Rule $r_{k+5} \equiv p_2 \to \mathbf{o}_2$ will be applied $-\left[f_+(p(t)) + \sum_{i=0}^{k-1} a_k^i \cdot t^i\right]$ times, consuming all copies of p_2 and producing $-\left[f_+(p(t)) + \sum_{i=0}^{k-1} a_k^i \cdot t^i\right]$ new copies of object $\mathbf{o}_2$.

(4) For each j, $1 \le j \le k$, rule $r_j \equiv b_j \to p_1^{f_+(a_k^{j-1})} p_2^{f_-(a_k^{j-1})} b_j^{\binom{j-1}{j-1}} \dots b_k^{\binom{k-1}{j-1}}$ will be applied $(t+1)^{j-1}$ times.

All the previous rules are applied in parallel in one transition step. Thus, in this case, at configuration $\mathcal{C}_{t+2}$ the content of membrane labeled by 1 is the multiset which contains objects $\mathbf{o}_1, \mathbf{o}_2, p_1, p_2$ and b_j, $(1 \le j \le k)$ with the following multiplicities:

- Multiplicity of object $\mathbf{o}_1$: $f_+(p(t+1))$.
  Indeed, from (2) all copies of object t $\mathbf{o}_1$ are consumed, but $f_+(p(t+1)) \overset{(c)}{=} 0$.
- Multiplicity of object $\mathbf{o}_2$: $f_-(p(t+1))$.
  Indeed, after execution of the rules, its multiplicity will be

$$\begin{aligned} f_-(p(t)) - \left[f_+(p(t)) + \sum_{i=0}^{k-1} a_k^i \cdot t^i \right] \\ = -p(t) - \sum_{i=0}^{k-1} a_k^i \cdot t^i \\ = -p(t+1) \overset{(c)}{=} f_-(p(t+1)). \end{aligned}$$

- Multiplicity of object p_1: $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot (t+1)^i$.
  Indeed, after execution of the rules from (1) all copies of p_1 are consumed but from (4) the produced copies are the following:

$$\sum_{i=1}^{k} f_+(a_k^{i-1}) \cdot (t+1)^{i-1} = \sum_{i=0}^{k-1} f_+(a_k^i) \cdot (t+1)^i$$

- Multiplicity of object p_2: $\sum_{i=0}^{k-1} f_-(a_k^i) \cdot (t+1)^i$.
  Indeed, after execution of the rules from (1), (2) and (3), all copies of object p_2 are consumed but by applying rules in (4) new copies of p_2 are produced, in total the number of copies will be:

$$\sum_{i=1}^{k} f_-(a_k^{i-1}) \cdot (t+1)^{i-1} = \sum_{i=0}^{k-1} f_-(a_k^i) \cdot (t+1)^i$$

- Multiplicity of object b_j, for each j, $1 \leq j \leq k$: $(t+2)^{j-1}$.
  Indeed, from (4) the total number of copies of object b_j produced is

$$\sum_{s=0}^{j-1} \binom{j-1}{s} (t+1)^s = [(t+1)+1]^{j-1} = (t+2)^{j-1}$$

Hence, in this case the result holds for $t+1$.

Case 4: $\left[\boldsymbol{\sum_{i=0}^{k-1} a_k^i \cdot t^i < 0} \right] \wedge \left[\boldsymbol{f_+(p(t)) + \sum_{i=0}^{k-1} a_k^i \cdot t^i > 0} \right]$

In this case, the following holds:

(a) $\sum_{i=0}^{k-1} f_+(a_k^i) - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i < 0$.
Indeed, it is enough to notice that $\sum_{i=0}^{k-1} a_k^i \cdot t^i < 0$, and $a_k^i = f_+(a_k^i) - f_-(a_k^i)$, for $0 \leq i \leq k-1$.

(b) $f_+(p(t)) > -\left[\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i \right]$.
Indeed, it suffices to bear in mind that $f_+(p(t)) > -\sum_{i=0}^{k-1} a_k^i \cdot t^i = -\left[\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i \right]$ and $a_k^i = f_+(a_k^i) - f_-(a_k^i)$, for $0 \leq i \leq k-1$.

(c) $p(t) > 0$ and $p(t+1) > 0$.
Indeed, from the hypothesis in this case we have $f_+(p(t)) > -\sum_{i=0}^{k-1} a_k^i \cdot t^i > 0$. So, $p(t) > 0$ and $f_+(p(t)) = p(t)$. Thus,

$$p(t+1) = p(t) + \sum_{i=0}^{k-1} a_k^i \cdot t^i = f_+(p(t)) + \sum_{i=0}^{k-1} a_k^i \cdot t^i > 0$$

Therefore, in this case configuration $\mathcal{C}_{t+1}$ yields configuration $\mathcal{C}_{t+2}$ as follows:

(1) From (a) we deduce that at configuration $\mathcal{C}_{t+1}$ in membrane labeled by 1 there are more copies of object p_2 than copies of object p_1, so rule $r_{k+1} \equiv p_1 p_2 \rightarrow \lambda$ will be applied $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i$ times, consuming all copies of p_1 and "remaining" $\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i$ copies of p_2 without evolving.

(2) From (b) we deduce that at configuration $\mathcal{C}_{t+1}$ in membrane labeled by 1 there are more copies of object $\mathbf{o}_1$ than copies of object p_2, so rule $r_{k+3} \equiv p_1 \mathbf{o}_1 \rightarrow \lambda$ will be applied $\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i$ times, consuming all copies of p_2 and "remaining" $f_+(p(t)) - \left[\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i \right]$ copies of $\mathbf{o}_1$ without evolving.

(3) For each j, $1 \leq j \leq k$, rule $r_j \equiv b_j \rightarrow p_1^{f_+(a_k^{j-1})} p_2^{f_-(a_k^{j-1})} b_j^{\binom{j-1}{j-1}} \dots b_k^{\binom{k-1}{j-1}}$ will be applied $(t+1)^{j-1}$ times.

All the previous rules are applied in parallel in one transition step. Thus, in this case, at configuration $\mathcal{C}_{t+2}$ the content of membrane labeled by 1 is the multiset which contains objects $\mathbf{o}_1, \mathbf{o}_2, p_1, p_2$ and b_j, $(1 \leq j \leq k)$ with the following multiplicities:

- Multiplicity of object $\mathbf{o}_1$: $f_+(p(t+1))$.
  Indeed, from (2) we deduce that the number of copies of object $\mathbf{o}_1$ is:

$$\begin{aligned} f_+(p(t)) - \left[\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i - \sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i \right] \\ = f_+(p(t)) + \sum_{i=0}^{k-1} a_k^i \cdot t^i \overset{(c)}{=} p(t) + \sum_{i=0}^{k-1} a_k^i \cdot t^i \\ = p(t+1) \overset{(c)}{=} f_+(p(t+1)) \end{aligned}$$

  all copies of object t $\mathbf{o}_1$ are consumed, but $f_+(p(t+1)) \overset{(c)}{=} 0$.
- Multiplicity of object $\mathbf{o}_2$: $f_-(p(t+1))$.
  Indeed, object $\mathbf{o}_2$ is not involved by the application of rules to reach configuration $\mathcal{C}_{t+2}$ from configuration $\mathcal{C}_{t+1}$, so the multiplicity of $\mathbf{o}_2$ is $f_-(p(t)) \overset{(c)}{=} 0 \overset{(c)}{=} f_-(p(t+1))$.

- Multiplicity of object p_1: $\sum_{i=0}^{k-1} f_+(a_k^i) \cdot (t+1)^i$.

  Indeed, from (1) all copies of object p_1 are consumed but from (3) the total number of produced copies is

$$\sum_{j=1}^{k} f_+(a_k^{j-1}) \cdot (t+1)^{j-1} = \sum_{i=0}^{k-1} f_+(a_k^i) \cdot (t+1)^i.$$

- Multiplicity of object p_2: $\sum_{i=0}^{k-1} f_-(a_k^i) \cdot (t+1)^i$.

  Indeed, from (1) all copies of object p_2 are consumed but from (3) the total number of produced copies is

$$\sum_{j=1}^{k} f_-(a_k^{j-1}) \cdot (t+1)^{j-1} = \sum_{i=0}^{k-1} f_-(a_k^i) \cdot (t+1)^i.$$

- Multiplicity of object b_j, for each j, $1 \leq j \leq k$: $(t+2)^{j-1}$.

  Indeed, from (3) the total number of copies of object b_j produced is

$$\sum_{s=0}^{j-1} \binom{j-1}{s} (t+1)^s = [(t+1)+1]^{j-1} = (t+2)^{j-1}$$

Hence, in this case the result holds for $t+1$.

Corollary 1: Let $p(n) = a_0 + a_1 \cdot n + \cdots + a_k \cdot n^k$ be a polynomial of degree k such that $a_i \in \mathbb{Z}, i = 0, 1, \ldots, k$. Let $\Pi_{p(n)}$ be the deterministic transition P system considered in Definition 2. Then, polynomial $p(n)$ is computed by the system $\Pi_{p(n)}$ according with Definition 1.

Proof: From Theorem 1 we deduce that for each $t \in \mathbb{N}$ at configuration $\mathcal{C}_{t+1}$ the content of membrane labeled by 1 is the following multiset:

$$\{\mathbf{o}_1^{f_+(p(t))} \mathbf{o}_2^{f_-(p(t))} p_1^{\sum_{i=0}^{k-1} f_+(a_k^i) \cdot t^i} p_2^{\sum_{i=0}^{k-1} f_-(a_k^i) \cdot t^i} \\ b_1 \; b_2^{(t+1)} \; b_3^{(t+1)^2} \cdots \; b_k^{(t+1)^{k-1}}\}$$

In order to know the multiplicity of objects $\mathbf{o}_1$ and $\mathbf{o}_2$ in membrane labeled by 1 at configuration $\mathcal{C}_{t+1}$, two cases are distinguished:

- If $p(t) \geq 0$ then $f_+(p(t)) = p(t)$ and $f_-(p(t)) = 0$. So, the multiplicity of $\mathbf{o}_1$ is $f_+(p(t)) = p(t)$ and the multiplicity of $\mathbf{o}_2$ is $f_-(p(t)) = 0$.
- If $p(t) < 0$ then $f_+(p(t)) = 0$ and $f_-(p(t)) = -p(t)$. Thus, the multiplicity of $\mathbf{o}_1$ is $f_+(p(t)) = 0$ and the multiplicity of $\mathbf{o}_2$ is $f_-(p(t)) = -p(t)$.

IV. Descriptive Computational Resources

In this section, the descriptive computational resources required by the deterministic transition P system $\Pi_{p(n)}$ considered in Definition 2 which computes polynomial $p(n)$ with integer numbers coefficients, is depicted.

- The size of the working alphabet: k+4.
- The initial number of objects: $1 + |a_0|$.
- The number of rules: k+5.
- The total number of objects involved in the rules is $2^k + k + 9 + \sum_{i=0}^{k-1} |a_k^i|$.

Hence, the total amount of descriptive computational resources is exponential in the size of the polynomial.

V. Discussions

Until now, two kinds of methods have been reported in literature to implement automatic design of membrane systems. One is the reasoning way presented in this paper and [27], which is called REASON. The other is the metaheuristic approaches (META) already used in membrane systems design, such as genetic algorithms [21], [23], [25], in particular Permutation Penalty Genetic Algorithms (PPGAs) [26], and quantum-inspired evolutionary algorithm (QIEAs) [22], [24].

REASON and META have the following differences:

- *Concept:* META uses a metaheuristic approach to evolve a population of candidate P systems (feasible or infeasible) toward the successful P systems, while REASON uses inductive method (from simple to complex P systems) to obtain the successful P systems. A metaheuristic approach may be a genetic algorithm, a quantum-inspired evolutionary algorithm or others.
- *Usage:* META is quite easy to understand and master for a beginner, while REASON sounds a quite complex technique to a beginner.
- *Generation:* META is a more general technique than REASON and therefore it is possible to use META to design different P systems. While in REASON, different P systems are designed by using different specific reasoning techniques.
- *Resource:* In REASON, it is possible to calculate the resource required by a P system with respect to computing time and workspace. While in META, it is quite hard to summarize the resource.
- *Software:* The evaluation of a successful P system in META is performed by using the well-known P system simulator, P-Lingua [28]. REASON does not need any software.
- *Extendibility:* REASON can be easily extended from a specific to a general P system, e.g., from a low-degree to high-degree polynomial P system, for a kind of membrane system. This extendibility is not suitable for META.

VI. Conclusion

This paper extends the work in [27] from the automatic design of deterministic transition P systems for computing polynomials with natural number coefficients to the automatic design of such kind of membrane systems for computing polynomials with integer coefficients, by analyzing the syntactical and semantics ingredients of cell-like membrane systems. This is a significant step for the programmability of membrane systems, namely how to automatically design a P system by using programs so as to develop a useful toolbox for the community of membrane computing.

As future work we plan to extend this method in order to design new variants of membrane systems with the capability of performing more complex tasks like finding the minimal membrane system, with respect to the number of used objects, for a given assignment or like practical applications such as membrane controllers for mobile robots. On the other hand, we propose: (a) to develop software platforms to simulate transition P systems using a weak interpretation of the priorities as well as FPGA (Field Programmable Gate Array) based hardware to implement them; and (b) the use membrane-inspired evolutionary algorithms [15], [29], [30] or optimization spiking neural P systems [31] to implement the automatic design of a membrane systems (including spiking neural P systems) for solving computationally hard problems.

References

[1] Gh. Păun, "Computing with membranes," *J. Comput. Syst. Sci.*, vol. 61, no. 1, pp. 108–143, Aug. 2000.

[2] Gh. Păun, G. Rozenberg, and A. Salomaa, *The Oxford Handbook of Membrane Computing*. New York, NY, USA: Oxford Univ. Press, 2010.

[3] M. Gheorghe, Gh. Păun, M. J. Pérez-Jiménez, and G. Rozenberg, "Research frontiers of membrane computing: Open problems and research topics," *Int. J. Found. Comput. Sci.*, vol. 24, no. 5, pp. 547–624, 2013.

[4] Gh. Păun, Y. Suzuki, and H. Tanaka, "On the power of membrane division in P systems," *Theor. Comput. Sci.*, vol. 324, no. 1, pp. 61–85, 2004.

[5] C. Martín-Vide, Gh. Păun, J. Pazos, and A. Rodríguez-Patón, "Tissue P systems," *Theor. Comput. Sci.*, vol. 296, no. 2, pp. 295–326, 2003.

[6] M. Ionescu, Gh. Păun, and T. Yokomori, "Spiking neural P systems," *Fundam. Inf.*, vol. 71, no. 2, pp. 279–308, 2006.

[7] L. Pan and X. Zeng, "Small universal spiking neural P systems working in exhaustive mode," *IEEE Trans. Nanobiosci.*, vol. 10, no. 2, pp. 99–105, Jun. 2011.

[8] L. Pan, J. Wang, and H. J. Hoogeboom, "Spiking neural P systems with astrocytes," *Neural Comput.*, vol. 24, no. 3, pp. 805–825, 2012.

[9] L. Pan, Gh. Păun, G. Zhang, and F. Neri, "Spiking neural P systems with communication on request," *Int. J. Neural Syst.*, vol. 27, no. 8, 2017, Art. no. 1750042.

[10] A. Alhazov, C. Martín-Vide, and L. Pan, "Solving a PSPACE-complete problem by recognizing P systems with restricted active membranes," *Fundamenta Informaticae*, vol. 58, no. 2, pp. 66–77, 2003.

[11] L. Pan and C. Martin-Vide, "Solving multidimensional 0–1 knapsack problem by P systems with input and active membranes," *J. Parallel Distrib. Comput.*, vol. 65, no. 12, pp. 1578–1584, 2005.

[12] B. Song, T. Song, and L. Pan, "Time-free solution to sat problem by P systems with active membranes and standard cell division rules," *Natural Comput.*, vol. 14, no. 4, pp. 673–681, 2015.

[13] G. Ciobanu, M. J. Pérez-Jiménez, and Gh. Păun, Eds., *Applications of Membrane Computing* (Natural Computing Series). Berlin, Germany: Springer, 2006.

[14] P. Frisco, M. Gheorghe, M. J. Pérez-Jiménez, Eds., *Applications of Membrane Computing in Systems and Synthetic Biology* (Emergence, Complexity and Computation). Berlin, Germany: Springer, 2014.

[15] G. Zhang, M. Gheorghe, L. Pan, and M. J. Pérez-Jiménez, "Evolutionary membrane computing: A comprehensive survey and new results," *Inf. Sci.*, vol. 279, pp. 528–551, Sep. 2014.

[16] G. Zhang, M. J. Pérez-Jiménez, and M. Gheorghe, *Real-life Applications with Membrane Computing* (Emergence, Complexity and Computation). Berlin, Germany: Springer, 2017.

[17] H. Peng, J. Wang, M. J. Pérez-Jiménez, H. Wang, J. Shao, and T. Wang, "Fuzzy reasoning spiking neural P system for fault diagnosis," *Inf. Sci.*, vol. 235, pp. 106–116, Jun. 2013.

[18] C. Buiu, C. Vasile, and O. Arsene, "Development of membrane controllers for mobile robots," *Inf. Sci.*, vol. 187, no. 1, pp. 33–51, 2012.

[19] X. Wang *et al.*, "Design and implementation of membrane controllers for trajectory tracking of nonholonomic wheeled mobile robots," *Integr. Comput.-Aided Eng.*, vol. 23, no. 1, pp. 15–30, 2016.

[20] G. Zhang, J. Cheng, T. Wang, X. Wang, and J. Zhu, Eds., *Membrane Computing: Theory and Applications*. Beijing, China: Science Press, 2015.

[21] G. Escuela and M. Á. G. Naranjo, "An application of genetic algorithms to membrane computing," in *Proc. 8th Brainstorming Week Membrane Comput.*, 2010, pp. 101–108.

[22] X. Huang, G. Zhang, H. Rong, and F. Ipate, "Evolutionary design of a simple membrane system," in *Membrane Computing* (Lecture Notes in Computer Science), vol. 7184, M. Gheorghe, Gh. Păun, G. Rozenberg, A. Salomaa, and S. Verlan, Eds. Berlin, Germany: Springer, 2012, pp. 203–214.

[23] C. Tudose, R. Lefticaru, and F. Ipate, "Using genetic algorithms and model checking for P systems automatic design," in *Nature Inspired Cooperative Strategies for Optimization* (Studies in Computational Intelligence), vol. 387, D. A. Pelta, N. Krasnogor, D. Dumitrescu, C. Chira, and R. Lung, Eds. Berlin, Germany: Springer, 2011, pp. 285–302.

[24] Y. Chen, G. Zhang, T. Wang, and X. Huang, "Automatic design of a P system for basic arithmetic operations," *Chin. J. Electron.*, vol. 23, no. 2, pp. 302–304, 2014.

[25] Z. Ou, G. Zhang, T. Wang, and X. Huang, "Automatic design of cell-like P systems through tuning membrane structures, initial objects and evolution rules," *Int. J. Unconventional Comput.*, vol. 9, nos. 5–6, pp. 425–443, 2013.

[26] G. Zhang, H. Rong, Z. Ou, M. J. Pérez-Jiménez, and M. Gheorghe, "Automatic design of deterministic and non-halting membrane systems by tuning syntactical ingredients," *IEEE Trans. Nanobiosci.*, vol. 13, no. 3, pp. 363–371, Sep. 2014.

[27] W. Yuan, G. Zhang, M. J. Pérez-Jiménez, T. Wang, and X. Huang, "P systems based computing polynomials: Design and formal verification," *Natural Comput.*, vol. 15, no. 4, pp. 591–596, 2016.

[28] M. García-Quismondo, R. Gutiérrez-Escudero, I. Pérez-Hurtado, M. J. Pérez-Jiménez, and A. Riscos-Núñez, "An overview of P-lingua 2.0," in *Workshop Membrane Computing* (Lecture Notes in Computer Science), vol. 5957, Gh. Păun, M. J. Pérez-Jiménez, A. Riscos-Núñez, G. Rozenberg, and A. Salomaa, Eds. Berlin, Germany: Springer, 2010, pp. 264–288.

[29] G. Zhang, J. Cheng, M. Gheorghe, and Q. Meng, "A hybrid approach based on differential evolution and tissue membrane systems for solving constrained manufacturing parameter optimization problems," *Appl. Soft Comput.*, vol. 13, no. 3, pp. 1528–1542, 2013.

[30] J. Xiao, Y. Huang, Z. Cheng, J. He, and Y. Niu, "A hybrid membrane evolutionary algorithm for solving constrained optimization problems," *Optik*, vol. 125, no. 2, pp. 897–902, 2014.

[31] G. Zhang, H. Rong, F. Neri, and M. J. Pérez-Jiménez, "An optimization spiking neural P system for approximately solving combinatorial optimization problems," *Int. J. Neural Syst.*, vol. 24, no. 5, pp. 1–16, 2014.