Evolution and Oscillation in P Systems: Applications to Biological Phenomena

Vincenzo Manca, Luca Bianco, and Federico Fontana

University of Verona,
Department of Computer Science,
strada Le Grazie, 15
37134 Verona, Italy
vincenzo.manca@univr.it
http://www.sci.univr/∼manca

Abstract. Some computational aspects and behavioral patterns of P systems are considered, emphasizing dynamical properties that turn useful in characterizing the behavior of biological and biochemical systems. A framework called state transition dynamics is outlined in which general dynamical concepts are formulated in completely discrete terms. A metabolic algorithm is defined which computes the evolution of P systems modeling important phenomena of biological interest once provided with the information on the initial state and reactivity parameters, or growing factors. Relationships existing between P systems and discrete linear systems are investigated. Finally, exploratory considerations are addressed about the possible use of P systems in characterizing the oscillatory behavior of biological regulatory networks described by metabolic graphs.

1 Introduction

In 1998 *P systems* were presented as a new model of computation [14]. Before their advent, some classes of rewriting systems had already shown the ability of expressing specific biological phenomena [22, 9, 10]. P systems move a step further: they have clear structural analogies with the cell, in particular they model several features of the biological membranes (for this reason they are often referred to as *membrane systems*). Moreover, the transitions happening in these systems recall certain evolution processes that take place in a living cell.

From a formal viewpoint, P systems satisfy a result of universality even in their basic definition [14]. In this sense they have all the computational power needed to capture a biomolecular process—provided that we are able to arrange it into an algorithmic procedure. In addition to this, the similarities existing between P systems and (at least some aspects of) biological cells might suggest that P systems are also able to represent the same process in a meaningful way, that is, not only to compute it as any universal machine would do, but also to provide potential insight on the biological mechanisms determining and controlling the process via the observation of the transitions of the system.

G. Mauri et al. (Eds.): WMC5 2004, LNCS 3365, pp. 63–84, 2005.

However, this is true only to some extent. Modeling specific biological activities inside a P system is not an easy task. A lot of alternative constructs derived from the basic definition of P system have been proposed, sometimes capturing crucial aspects of the biology of cells such as *thickness*, *polarity*, *catalysts*, *inhibitors*, *promoters*, *carriers*, *porters* (*symport-antiport*), *priority*, *division*, *replication*, *creation*, *dissolution*, *resources*, and *energy* [15, 17, 16, 12, 5, 1]. In other cases, powerful paradigms were imported from other formal systems having biological implications too, such as *splicing* and object-structuring (in form of strings) [15]. All these alternative constructs exhibit properties of universality, hence by all means they represent a first, necessary attempt to get P systems closer to the world of bio-molecules meanwhile preserving their computational power.

Nevertheless there are some aspects, that are crucial in almost any study of biomolecular processes, that the traditional formulations of P systems do not develop in a sufficient way from a biological point of view.

The halting of a P system tells that a computation has terminated successfully, but the terminal state is not a primary object of investigation in many biomolecular realities. Rather, we would shift the focus on the computation during its "life", in an aim to observe the living organism while surviving in the environment and, possibly, to influence his life cycle when some bio-chemical indicators tell that his physiological activity is altered (possibly harmfully). In other words, biological systems do not compute states, but rather stable behavioral pattern that satisfy some "enjoyable" conditions, and life cycles are combined and organized in very complex forms. This means that considering all the forms of periodicity in the framework of P systems is of key importance if we want to apply P systems to modeling biological processes [1]. Moreover, life, in its adaptation and evolution strategies, explores behavior spaces in a range between simple cycles and chaotic behavioral magmas where space in lost in time and *vice versa*. Therefore, chaos is very important as limit border that life try to approximate to (with the risk of falling in its destructive abyss) because "at edge of chaos" is available the dynamical richness necessary for adaptation and evolution [10].

Biomolecular mechanisms are the result of many individual local reactions, each one of those being formed by processes whose extension is limited in time and space. These processes interact with each other by means of specific communication strategies, in a way that they finally exhibit a (sometimes surprising) overall co-ordination. In this sense, and despite this co-ordination, biomolecular processes are by all means *asynchronous*.

P systems, in their classical formulation, are intended to "consume" the available resources in a maximally parallel way during the rewriting of symbols. Holding this property, then all the symbols that are present in the system at a given configuration become potential resources: they are consumed as many as possible, and new symbols are produced in consequence of that action. In other words, maximal parallelism constrains the system to consume all the available resources during a transition. Moreover, their evolution is synchronous, i.e., a

global clock triggers the production of new symbols inside all membranes. This limits their versatility in modeling biological asynchronous phenomena.

In this paper we focus on some theoretical and practical issues especially oriented to biomolecular computing. First, we consider a perspective (still in progress for many aspects) according to which P systems are cast in a discrete dynamical framework. In this perspective, we will characterize classical dynamical concepts in terms of *state transition dynamics* [11].

Next, we propose to observe rewriting rules in membranes from a different viewpoint. Membranes are intended to host "symbolic reactions", and rules apply according to some reaction parameters and substance concentration, as it normally happens in biochemical phenomena. We define a *metabolic algorithm* for computing the evolution of (deterministic) P systems when some initial state and some reaction parameters are given, such as reactivities or growing factors. This algorithm is applied to known bio-chemical oscillatory phenomena, and put in relation with differential equations.

The similarities arising between the symbolic and quantitative approach pursued by P systems and differential equation systems, respectively, stimulate the discovery of relationships existing between P systems and some widely used, simple but powerful systems of equations expressing differential problems in the discrete time, called *discrete linear systems*. Such relationships are addressed before the conclusion, where some extra considerations are made on the possibility to reproduce the behavior of biological networks expressed in terms of *metabolic graphs*: in the description of these graphs (networks) the emphasis is on the oscillatory rather than temporal behavior, so that specific mathematical tools based on a model description in the frequency domain are proposed.

2 State Transition Dynamics

One classic approach to discrete system modelling consists in first analyzing a continuous phenomenon, then producing a discrete model of it according to a given discretization method, and finally running a simulation, provided that the discrete model respects certain stability conditions.

There are cases in which a discrete model of a continuous phenomenon generates errors, but such errors can be arbitrarily reduced or, equivalently, the precision is proportional to the granularity with which the continuous phenomenon is reproduced by the discrete model. Sometimes the information needed to describe a physical phenomenon is *inherently* discrete in a way that the resulting discrete model reproduces the reality almost directly (think, for instance, to DNA replication). In this last case having a method that could compute the dynamics of the system directly from its discrete representation would be a great advantage with respect to many aspects.

The *state transition dynamics* formalism considers a system defined in a discrete domain, assuming discrete values. It studies properties such as orbits and trajectories, periodicity, eventual periodicity and divergence, fixed points, attractors and recurrence [4, 6, 3], aimed at defining analogous concepts in the

context of systems discrete in space and in time, with no metric or topological structure. It is surprising that, even assuming a very weak mathematical structure, many concepts can be defined formally in such a way that interesting facts can be deduced on the structure of attractors, on deterministic *chaos*, and on its relationship with nondeterminism [11].

To give an idea of the characterization given by state transition dynamics, here we report the most important definitions. For more details, discussions and mathematical developments we refer to [11] where we started a general approach to discrete systems dynamics that is under development.

Definition 1. *A state transition dynamics is a pair* (S, q) *where* S *is a set of states and* q *is a function from* S *into its power set,*

$$q : S \to \mathcal{P}(S).$$

By calling *quasi state* any subset X of S, and extending the application of q over quasi states, i.e.,

$$q(X) = \bigcup_{x \in X} q(x),$$

then we map quasi states into quasi states by means of q to form *orbits*, and characterize specific *trajectories* along these orbits by means of the following definitions.

Definition 2. *An* X-orbit *is a sequence* $\{X_i\}_{i \in \mathbf{N}}$ *of quasi states such that*

$$\begin{array}{l} X_0 = X, \\ X_i \subseteq q(X_{i-1}) \ , \ i > 0. \end{array} \tag{1}$$

A X-orbit *is complete when the previous inclusion is replaced by an equality. When* x *is a state, we write simply* x-orbit *instead of* $\{x\}$-orbit.

An s-trajectory *is a function* $\xi : \mathbf{N} \to S$ *such that*

$$\begin{array}{l} \xi(0) = s, \\ \xi(i) \in q\big(\xi(i-1)\big) \ , \ i > 0. \end{array} \tag{2}$$

By denoting with q^i the composition of q repeated i times and $q^*(s) = \bigcup_{i \in \mathbf{N}} q^i(s)$, we refer as *flights* and *blackholes* to the following special trajectories:

Definition 3. *An* s*-trajectory is an* s-flight *if it is an injective function on* $\mathbf{N}$. *An* s*-flight is an* s-blackhole *if* $q^*(s) \subseteq \xi(\mathbf{N})$ *(where* ξ *is extended to sets).*

When S is made of symbolic values then the relation $y \in q(x)$ induced by q between two states, x and y, is conveniently expressed using the notation typical to rewriting systems: $x \to y$. Note that we can easily introduce non terminating computations as long as q is total.

It is clear that the notion of dynamical system defined above is nondeterministic, because any state can transform into a set of possible states—though, an equivalently expressive deterministic system where states are the quasi states of the original system can be figured out. The nondeterministic aspect is essential for the modeling of many phenomena.

We now give a characterization of the evolution in these systems.

Definition 4. *An X-orbit is* periodic *if $q^n(X) = X$ for some $n > 0$. An orbit is* eventually periodic *if $q^{n+k}(X) = q^k(X)$ for some $k, n > 0$. In this case k is called the* transient *and n the* period.

Definition 5. *An X-orbit is $\Omega\big(f(n)\big)$-*divergent *with respect to a function $\mu : S \to \mathbf{N}$, called Ljapounov function, if $\mu\big(q^n(X)\big)$ has order $\Omega\big(f(n)\big)$. A similar definition holds for the order of divergence $O\big(f(n)\big)$.*

Definition 6. *A state s is a fixed point if the transition relation transforms it into itself, that is, $q(s) = \{s\}$.*

Periodicity and eventual periodicity are properties with a strong computational significance. It can be shown that, in a suitable computational framework where every machine finds a counterpart in a corresponding state transition dynamics, the periodicity decision problem turns out to be computationally equivalent to the termination problem [11]:

Proposition 1 *Given a computationally universal class of machines, then the (eventual) periodicity of the related dynamical systems is not decidable.*

Affine to periodicity (but weaker) is recurrence:

Definition 7. *A state x is* recurrent *if $x \in q^n(x)$ for some $n > 0$. A state x is* eternally recurrent *if $\forall n > 0 : y \in q^n(x) \Rightarrow \exists m > 0 : x \in q^m(y)$.*

A system dynamics is ultimately characterized by its *attractors*, that in very first approximation can be seen as quasi states in which the system must fall in the end. First of all, we say that an orbit is *included* in another orbit if the former sequence is contained in the latter sequence, and we say *eventually included* if it is included in the other orbit except for a finite number of quasi states.

We call *basin* a set $B \subseteq S$ such that $q(x)$ is included in B for every state $x \in B$. Inside a basin we possibly find an *attracting set* A, i.e., a subset which eventually includes the complete x-orbit of every state $x \in B$. If A is minimal under set inclusion, i.e., no subsets (even made of a single state) can be removed from A otherwise causing the lost of the attracting property, then our attracting set is an attractor.

A complete characterization of attractors requires more definitions than those reported in this paper [11]. In particular, here we have only outlined the so-called *unavoidable* attracting sets that can have three different types:

1. *periodic attractors*, that is, periodic orbits (fixed point attractors are a special case);
2. *eternally recurrent blackholes*;
3. *complex attractors*, that is, a combination of the two previous cases.

Many concepts in formal language theory can be revisited in the framework of state transition dynamics. For example, languages generated by grammars or recognized by automata are special cases of attractors. But next issues, that

are crucial in the development of state transition dynamics, are: i) the extension of its focus on more complex dynamical phenomena such as the forms and degrees of chaos, intermittency, dissipation, resonance; ii) the search for dynamical parameters useful in the qualitative analysis of dynamical patterns. In fact, both cellular automata and Kauffman networks enlighten that the relationship between the transition function and the state structure strongly determines dynamically relevant qualities [20, 21]. We put forward that several parameters that are identified in those contexts, such as *connectivity, channeling, majority, input entropy, and Derrida plot*, could inspire some analogues in P systems. The approach we present in the next section will give some hints in this direction. In fact, the metabolic viewpoint will cast P systems in the framework of dynamical networks to which both cellular automata and Kauffman networks belong.

3 Metabolic Algorithm and Oscillatory Phenomena

Our proposed algorithm is inspired by a chemical reading of the rewriting rules. Due to the biological implications of this type of reading, we called the algorithm *metabolic*.

The re-interpretation of the rewriting rules in the light of chemical reactions is not new: several researchers have applied rewriting systems to contexts different from the purely abstract one, giving alternative meanings to the rules [18, 19]. In P systems every rule can be seen as a binary relation between strings, mapping the leftward argument into the rightward one. For instance, a rule $r : AB \to CD$ containing symbols defined over an alphabet V states that every occurrence of the object $A \in V$ in the system, once paired with $B \in V$, can be substituted by the pair of objects $CD \in V^*$.

If we look at r as a *chemical reaction*, now the leftward objects A and B have the role of *reactants* whereas those on the right are *products*. Following this chemical interpretation, we propose to look at rules as descriptors of the changes in concentration of the reactants into products. In other words, r says that a number of objects of type A and B transforms into objects of type B and C. In this way we deal with populations rather than single objects.

This interpretation needs the introduction of some definitions. Consider a P system on an alphabet $V = \{A, B, C, \ldots\}$, provided with a nonempty set R of rewriting rules. Every rule $r : \alpha \to \beta$, with $\alpha, \beta \in V^*$, is associated to a *reactivity coefficient* k_r whose role will be made clear in the following.

For each membrane M we give a maximum number of objects $|M|$ that cannot be overcome. From here we agree to define a conventional *molarity unit*:

$$\mu = \nu\,|M|,$$

where ν is a molarity factor ($\nu = 0.01$ in our experiments). We denote with $|X|$ the number of elements of type X in M, and define the quantity

$$||X|| = \frac{|X|}{\mu} \tag{3}$$

as the number of *moles* of X inside M. This molar formulation for the quantities involved in a reaction leads to the *α-molar concentration*, defined as the product of the moles of every object in a string $\alpha = \alpha_1 \dots \alpha_{|\alpha|}$:

$$||\alpha|| = \prod_{i=1}^{|\alpha|} ||\alpha_i||. \tag{4}$$

It is now possible to describe an algorithm that translates the rewriting rules into a set of equations defining the *molar variation*, $\Delta||X||$, of every element X in consequence of the application of the rules.

A rule $r : \alpha \to \beta \in R$ acts on the leftward (i.e., reactant) and rightward (i.e., product) objects: the leftward part of r diminishes the concentration of the reactants, whereas the rightward part increases the concentration of the products. Hence, the changes in the amount for an element X in M due to r are equal to the *stoichiometric coefficient*:

$$|\beta|_X - |\alpha|_X,$$

where $|\gamma|_S$ indicates the number of occurrences of S contained in γ.

In chemical terms, r affects the concentration of every element appearing in it by a similar contribution, depending on the concentration of all the reactants at the instant of application. The term $||\alpha||$ takes this aspect into account, according to equation (4). Thus, we can compute the effect $\Delta_r||X||$ of a rule $r : \alpha \to \beta$ on the concentration of X, as

$$\Delta_r||X|| = k_r \, ||\alpha|| \, (|\beta|_X - |\alpha|_X), \tag{5}$$

where k_r is the *reactivity coefficient* of the rule.

In general an object is involved in more than one rule. In order to compute the overall molar variation of an object X we have to take the contributions of all rules into account. This is made by summing up their effects on the concentration of X:

$$\Delta||X|| = \sum_{r \in R} \Delta_r||X||, \tag{6}$$

where R is the set of rules in our P system.

Hence, after the application of a set of rules our algorithm updates the number of moles of an object X according to the following assignment:

$$||X|| := ||X|| + \Delta||X||. \tag{7}$$

The multiplicity of X is updated accordingly:

$$|X| := |X| + \mu\Delta||X||. \tag{8}$$

Let us now see a concrete example of this translation from rewriting rules to *metabolic equations*. Consider the following set of rules:

$$\begin{aligned} r1 &: AC \;\; \to AB, \\ r2 &: BC \;\; \to A, \\ r3 &: BBB \to BC, \end{aligned} \tag{9}$$

each of them associated to a coefficient, respectively k_{r1}, k_{r2}, and k_{r3}. We want to calculate the variation in the multiplicity of every object in the system caused by the rules.

If we apply equation (6) to each object, then we obtain the following system of metabolic equations:

$$\begin{array}{rcl} \Delta||A|| & = & 0 \cdot k_{r1}||AC|| + 1 \cdot k_{r2}||BC|| + 0 \cdot k_{r3}||BBB||, \\ \Delta||B|| & = & +1 \cdot k_{r1}||AC|| - 1 \cdot k_{r2}||BC|| - 2 \cdot k_{r3}||BBB||, \\ \Delta||C|| & = & -1 \cdot k_{r1}||AC|| - 1 \cdot k_{r2}||BC|| + 1 \cdot k_{r3}||BBB||, \end{array} \tag{10}$$

where k_{r1}, k_{r2}, and k_{r3} can be read as "rates" of application of $r1$, $r2$ and $r3$, respectively. As we can see from (10), where all contributions (including null ones) are represented, it is always possible to figure out an equation for every object of the P system from the correspondent set of rewriting rules. Each of these equations gives the molar variation of the related element as time elapses.

By applying equation (3) we can figure out the finite *differentials* associated to the system (10):

$$\begin{array}{rcl} \Delta a & = & +\mu \cdot \frac{k_{r2}}{\mu^2} \cdot bc, \\ \Delta b & = & +\mu \cdot \frac{k_{r1}}{\mu^2} \cdot ac - \mu \cdot \frac{k_{r2}}{\mu^2} \cdot bc - 2\mu \cdot \frac{k_{r3}}{\mu^3} \cdot b^3, \\ \Delta c & = & -\mu \cdot \frac{k_{r1}}{\mu^2} \cdot ac - \mu \cdot \frac{k_{r2}}{\mu^2} \cdot bc + \mu \cdot \frac{k_{r3}}{\mu^3} \cdot b^3, \end{array} \tag{11}$$

in which we have denoted numbers of elements with a, b, c instead of $|A|, |B|, |C|$, respectively. Note that the correspondence between rewriting rules and differential equations is not bi-directional: in general there is no unique way to translate a system of differentials into a set of rewriting rules, whereas the other way round holds.

We want to emphasize an important fact about the coefficients k_r. In the molar formulation of rewriting rules they are called *reactivities*, and their role is to weight each rule's action. The reactivity of a rule takes many aspects into account: i) chemical and physical aspects of the reaction environment (pressure, temperature, PH level, catalyst activity, ...), ii) reaction speed (increasing or decreasing speed corresponds to a finer or coarser observation granularity), iii) proper features of single reactions that should account for the following aspects:

- rule activation percentage;
- synchronization and parallelism degree;
- reactants and energy partition.

If we consider all the interconnections existing between the points introduced in the previous list, then it is easy to understand that the tuning of reactivity factors is very important. We think that this aspect needs further investigation, and our future work will proceed along this line.

As previously seen, the multiplicity of X is updated according to (7) after each system transition. Unfortunately it might happen that a rule is applied too many times with respect to the reactant allowance, due to a wrong choice of the reactivity coefficients. In other words, the system in principle can consume more

reactants than those which are available at a given configuration. This violates the Principle of Mass Conservation.

To account for this, we add in our model a set of constraints that force the system to respect the Principle of Mass Conservation. One possible algorithm is the following: for every object X, before calculating its molar variation $\Delta||X||$ check if the amount $|X|$ becomes negative; if so, then stop the computation, else go on. Another possible work-around to a violation of the previously discussed constraints is to decrease each of the values of the reactivity parameters by a certain rate and, then, check again.

To clarify these ideas it is useful to calculate this set of constraints on a concrete example. Consider a P system with the set of rules (9) previously discussed; in order not to use more reactants than those available, we add the above constraints we to each reactant. In the example seen before these constraints become:

$$\begin{array}{l} \mathbf{C}_{|A|} : k_{r1}||AC|| < |A|, \\ \mathbf{C}_{|B|} : k_{r2}||BC|| + k_{r3}||BBB|| < |B|, \\ \mathbf{C}_{|C|} : k_{r1}||AC|| + k_{r2}||BC|| < |C|, \end{array} \tag{12}$$

where $\mathbf{C}_{|A|}$, $\mathbf{C}_{|B|}$ and $\mathbf{C}_{|C|}$, respectively, denote the constraints on the corresponding objects.

We want to stress that someone could think that the constraint on an object X can be equivalently calculated *after* the updating of $|X|$, by simply checking that it never assumes negative values. Once more, this is the wrong approach. In fact, even if the balance of positive and negative contributions results in an admissible variation, no one is able in this way to prevent that the amount of X consumed by all the reactions (those including it among their reactants) during a transition exceeds its real amount.

Once a constraint violation is discovered there are several ways to react. This investigation is still in progress. There are some open questions in our model, and our future work will try to give an answer to them. One of such questions deals with the temporal variation of the reactivity parameters, as independent functions in the system: we think that setting these parameters free to vary along time would have a strong impact on the system behavior, enabling it to simulate more complex reactions.

We would like to end this brief treatment outlining some of the results we get by this model implemented in a simulator *Psim*, developed in Java with an xml representation of membrane structure [2].

The first dynamical system we intend to model is a well known chemical oscillator called *Brusselator*; it is a simplified model of the Belousov-Zhabotinsky reaction [13, 7, 19]. When certain reactants like sulphuric acid, malonic acid, ferroin and bromate are combined together, in presence of a cerium catalyst, the chemical compound obtained, after a period of inactivity, starts a series of sudden oscillations in color ranging from red to blue. This chemical reaction could be described by the following rewriting rules:

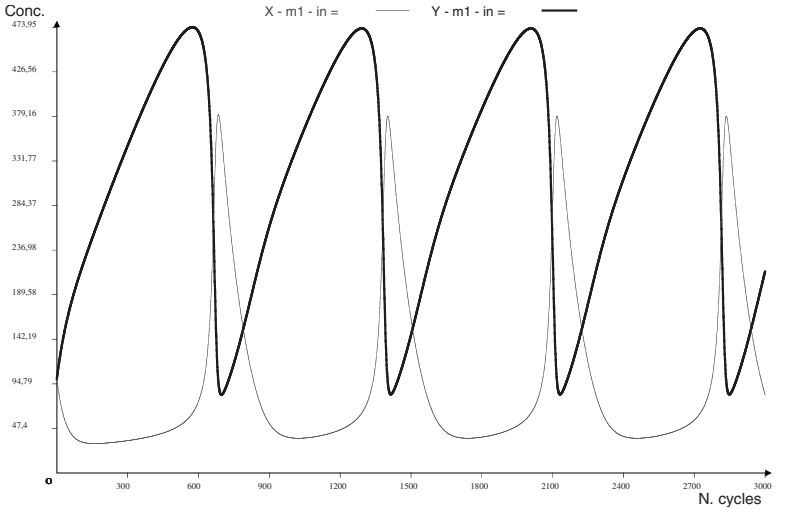

Fig. 1. Oscillations of Belousov-Zhabotinsky reaction model simulated by *Psim* with parameters $k_1 = 0.9$, $k_2 = 0.7$, $k_3 = 0.36$, $k_4 = 0.36$, $k_5 = 0.1$, $k_6 = 0.15$ and $\mu = 1000$ ($|M| = 100000$). Parameters could be rewritten in terms of k_1 in this way: $k_2 = 0.78 \cdot k_1$, $k_3 = 0.4 \cdot k_1$ and $k_4 = 0.4 \cdot k_1$

$$\begin{array}{lll} r_1: & A & \rightarrow & X, \\ r_2: & BX & \rightarrow & YD, \\ r_3: & XXY & \rightarrow & XXX, \\ r_4: & X & \rightarrow & C. \end{array} \tag{13}$$

It is usually made the assumption that the system described in this way inputs continuously reactants A and B from the outside environment; for this reason, in order to implement the reaction into our simulator, two rules have to be added at the set of rules (13):

$$\begin{array}{lll} r_5: & \lambda \rightarrow & A, \\ r_6: & \lambda \rightarrow & B, \end{array} \tag{14}$$

that are two generative rules which introduce some amount of objects A and B into the system.

It turns out that the oscillating behavior of the chemical reaction is mirrored, in the abstract system outlined by the rewriting rules, in the oscillations of the amounts of objects X and Y. We have translated this extended set of rules into our xml input file and fed it to the simulator: the trend of X and Y is visible in Figure 1, where it is possible to appreciate the perfect oscillating behavior of

the system's limit cycle. Accordingly with the assumptions made in [18] initially all objects have multiplicity equal to zero. Note that it is possible to relate all reactivity coefficients to their maximum value, as in Figure 1 k_1. This relationship is emphasized in Figure 1.

The second dynamic system we intend to investigate is a very basic predator-prey model, described, among others, in [7]. It is constituted by only two objects evolving over time: preys X and predators Y. We make the following four simplifying assumptions:

- preys grow up following a Malthusian model;
- preys' growing rate is reduced proportionally to predators' number;
- predators extinguish exponentially in absence of preys because they are predators' only sustenance;
- preys' presence make predators' growing rate increase proportionally to their number.

Under these assumptions this predator-prey model could be described by the well known Lotka-Volterra differential equations, where now $x = ||X||$ and $y = ||Y||$:

$$\begin{aligned} x' &= ax - dxy, \\ y' &= exy - by, \end{aligned} \tag{15}$$

extended by the initial conditions that $x_0 > 0$ and $y_0 > 0$.
Starting from these differential equations we have translated them into the following rewriting rules:

$$\begin{aligned} r1 &: X \rightarrow XX, \\ r2 &: XY \rightarrow YY, \\ r3 &: Y \rightarrow \lambda, \end{aligned} \tag{16}$$

with the following assignments:

$$a = k_{r1}; \quad d = \frac{k_{r2}}{\mu}; \quad b = k_{r3}; \quad e = \frac{k_{r2}}{\mu}$$

where k_{ri} and μ have the usual meaning and are input parameters of our model; in this way we get the metabolic equations:

$$\begin{aligned} \Delta||X|| &= k_{r1} \cdot ||X|| - k_{r2} \cdot ||XY|| \\ \Delta||Y|| &= -k_{r2} \cdot ||XY|| - k_{r3} \cdot ||Y|| \end{aligned} \tag{17}$$

Note that again all these rules and objects could be contained into a system with just one membrane.

We tested the system described so far starting with an initial amount of 100 preys and 20 predators. The simulation, as we can see from Figure 2, confirmed the oscillating behavior of the number of preys and predators in the predator-pray model described by the Lotka-Volterra system of equations.

The last model we discuss in this paragraph is that of an infective disease that spreads through a population and that could cause infected people's death or permanent immunity to the infection.

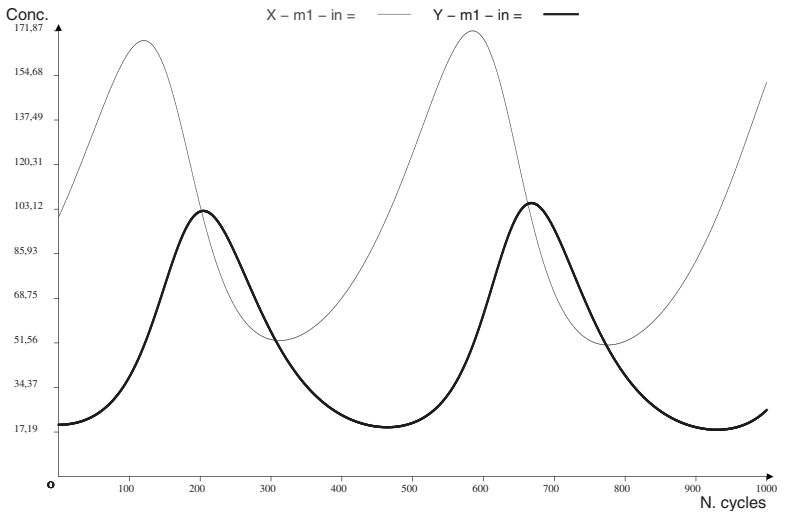

Fig. 2. Oscillations of the predator-prey model simulated by *Psim* with $k_1 = 0.01$, $k_2 = 0.02$, $k_3 = 0.02$ and $\mu = 100$ ($|M| = 10000$)

We make the simplifying assumption that the population is closed (e.g., it is made by a certain amount of people and where no births, immigration or emigration are allowed). The population of this dynamical system is partitioned into three different categories (objects of our system): healthy people C, ill people G, and immune people K. When an healthy person meets an ill one he becomes ill, with a probability depending on the reaction rate of the rule; an ill person has two possibilities: he could die, and could otherwise become immune forever to the infection. On the other hand, an healthy individual could keep his state until he gets no contact with an ill one. This pattern is common to many contagious phenomena, and could model also some forms of prion propagation that are the biomolecular basis of various infectious diseases of the nervous system (as bovine spongiform encephalopathy and Creutzfeldt-Jakob disease).

The behavior just described could be expressed with the following set of rules:

$$\begin{array}{lll} r_1: & CG \rightarrow & GG, \\ r_2: & G \quad \rightarrow & K, \\ r_3: & G \quad \rightarrow & \lambda, \end{array} \tag{18}$$

in which all the symbols have the meaning previously discussed. The simulation of such a system with our tool has outlined results in agreement with literature. In particular, it has put into evidence the existence of a threshold of activation

for the epidemic: on the one hand, if the initial healthy population is below a certain amount, the epidemic does not start and so ill people decrease in number until its complete vanishing. On the other hand, whenever the initial healthy population is beyond that threshold the epidemic activates and the number of ill people grows up until reaching its maximum and then drops again to zero thus vanishing.

Due to our choice of the parameters, as indicated in Figure 3 and 4, it turns out that the threshold we talked about is near 2570; we find accordingly two kinds of behaviors depending of the initial amount of healthy people: in Figure 3 is depicted the case in which the epidemic doesn't activate because of the number of initial healthy people being 2000 and thus under the threshold. On the other hand, in Figure 4 the initial amount of healthy people is 7000 and the epidemic does its course reaching its maximum and then vanishing. In both cases the initial number of ill people is fixed to 300.

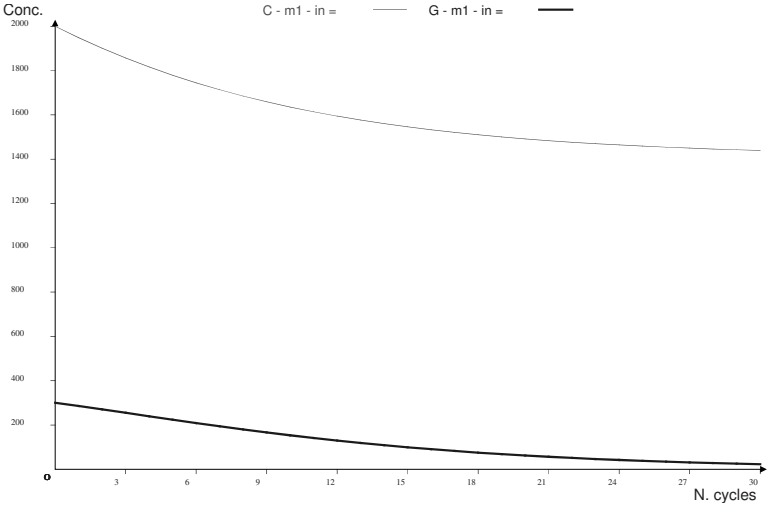

Fig. 3. Not active epidemic model simulated by *Psim* with $k_1 = 0.3$, $k_2 = 0.1$, $k_3 = 0.12$ and $\mu = 3500$

We end the section stressing once more the fact that all the examples discussed here, in spite of their extreme interest, are very simple from a topological viewpoint but their study has been very useful in order to evaluate the effectiveness of the metabolic algorithm proposed. Our work will, from now on, concentrate on the simulation of more elaborate systems.

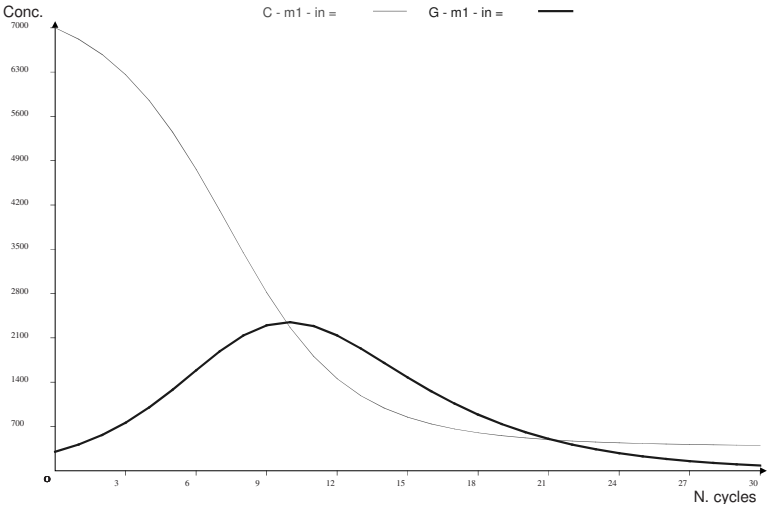

Fig. 4. Active epidemic model simulated by *Psim* with $k_1 = 0.3$, $k_2 = 0.1$, $k_3 = 0.12$ and $\mu = 3500$

4 Relationships with Linear Systems

As opposite to State Transition Dynamics, the traditional linear paradigm results in systems that are extremely simple from a formal point of view; meanwhile a lot of theoretical results exist about such systems that are useful in practice [8]. For these two reasons linear systems have found plenty of practical applications in system modeling and control, even of nonlinear phenomena.

Here we want to show that P systems can represent linear systems. Although this fact is implied by universality, nevertheless it is interesting to see how this representation can be given. We will make use of no peculiar and/or advanced properties to derive a linear restriction of P systems. In other words, we do not want to define any novel or complicate kind of construct to characterize a weaker family of P systems as those reproducing linear systems.

In its traditional formulation a *discrete linear* (DLI) system transforms, at a temporal step n, an N-dimensional state vector $\boldsymbol{v}$ according to a linear (matrix) transformation in a way that a new state will hold at the following time step. In the meantime, an output vector is produced as a linear combination of the actual state itself. It is well known by theory that for a stable system this output consists of (however many) damped sinusoids.

Coherently with the classic notion of P system here we do not consider the existence of an external input—though, this aspect is likely to be object of future research in conjunction with formulations of P systems that are capable

of accepting an input from the external environment [1]. Indeed, the external input is a crucial aspect in linear systems theory, and more in general in the modeling of real phenomena, as a factor that excites initially quiet systems and forces their evolution along time.

Apart from the input, the core structure (and, hence, the overall behavior, or *free evolution*) of a linear system depends on the structure of the transition matrix $\boldsymbol{A}$, sized $N \times N$. Thus, we can completely characterize a free-evolving linear system by the following set of equations holding at time step n:

$$\begin{cases} \boldsymbol{v}(n+1) = \boldsymbol{A}\boldsymbol{v}(n), \\ \boldsymbol{u}(n) \quad = \boldsymbol{C}\boldsymbol{v}(n), \end{cases} \tag{19}$$

in which the upper formula expresses the state transition, whereas the lower formula is responsible of the system output. Let $\boldsymbol{v}(0)$ be the initial state.

P systems cannot associate real numbers to symbols directly, as it happens for linear systems. We, then, restrict our numerical domain to rational numbers in a way that $\boldsymbol{A}$ and $\boldsymbol{C}$ can be respectively approximated by two matrices containing only rational numbers, with the desired precision. Now we collect all the matrix elements and, for each matrix, we compute the least common factor of such elements, $k_{\boldsymbol{A}}$ and $k_{\boldsymbol{C}}$ respectively. Similarly, we compute the least common factor $k_{\boldsymbol{v}}$ of the initial state elements. In this way we can move to a slightly different system:

$$\begin{cases} \tilde{\boldsymbol{v}}(n+1) = \tilde{\boldsymbol{A}}\tilde{\boldsymbol{v}}(n), \\ \tilde{\boldsymbol{u}}(n) \quad = \tilde{\boldsymbol{C}}\tilde{\boldsymbol{v}}(n), \end{cases} \tag{20}$$

in which $\tilde{\boldsymbol{A}} = k_{\boldsymbol{A}}\boldsymbol{A}$, $\tilde{\boldsymbol{C}} = k_{\boldsymbol{C}}\boldsymbol{C}$, and $\tilde{\boldsymbol{v}}(0) = k_{\boldsymbol{v}}\boldsymbol{v}(0)$. In other words we multiply both the transition and output matrix by their respective least common factors, in a way that the matrices resulting from this operation, $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{C}}$, respectively, contain only signed integers. Likewise we define $k_{\boldsymbol{v}}(0)$ as a modified initial state made of only integer values. These three rescalings lead to a modified linear system evolving within the domain of signed integers.

Since we are interested in studying the dynamic evolution of populations, then we can restrict our analysis to linear systems whose state is positive or null. The formal complications needed to account for negative values are left to further research: at the moment $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{C}}$ are made of positive numbers, and the modified initial state is positive as well.

Let us define a P system using symbols describing the state, $x_1, x_2, \ldots, x_N$, and z, and the output: $y_1, y_2, \ldots, y_N$ (output symbols are defined to add clarity to the treatment although they are not strictly necessary):

$$\begin{aligned} V &= \{x_1, x_2, \ldots, x_N, y_1, y_2, \ldots, y_N, z\}, \\ T &= \{y_1, y_2, \ldots, y_N\}\ . \end{aligned}$$

Let this system contain N membranes inside the skin:

$$\mu = [_{\text{skin}}\ [_1\ ]_1\ [_2\ ]_2\ \ldots\ [_N\ ]_N\ ]_{\text{skin}}\ .$$

The inner membrane labeled with j will contain the value of the j-th element of $\tilde{\boldsymbol{v}}(n+1)$, in terms of multiplicity of objects z contained in the membrane itself.

In the beginning, the initial state is encoded in the skin membrane as the following multiset of symbols:

$$w_{\text{skin}} = x_1^{\tilde{\boldsymbol{v}}_1(0)}\, x_2^{\tilde{\boldsymbol{v}}_2(0)}\, \ldots\, x_N^{\tilde{\boldsymbol{v}}_N(0)},$$

where $\tilde{\boldsymbol{v}}_i(n)$ denotes the value of the i-th element of the state vector $\tilde{\boldsymbol{v}}$ at time step n. Any other membrane is set to be initially empty: $w_1 = \ldots = w_N = \emptyset$.

A single-step evolution of the linear system is resolved in one transition of the corresponding P system involving both the symbols in the skin and in every inner membrane:

1. Every symbol x_j located inside the skin is distributed, once turned into z, into the i-th inner membrane with the multiplicity given by $a_{ij} \in \tilde{\boldsymbol{A}}$, for every $i = 1, \ldots, N$. In the meantime the same symbol, once turned into y_i, is sent out of the system with a multiplicity given by $c_{ij} \in \tilde{\boldsymbol{C}}$, again for each $i = 1, \ldots, N$. This happens for every component so that in the end we write

$$x_j \to z_{\text{in}_1}^{a_{1j}} \ldots z_{\text{in}_N}^{a_{Nj}}\, y_{1\,\text{out}}^{c_{1j}} \ldots y_{N\,\text{out}}^{c_{Nj}}, \quad j = 1, \ldots, N\,. \tag{21}$$

   Thus, (21) stores symbols z accounting for the new state in the inner membranes, meanwhile produces symbols $y_1, \ldots, y_N$ accounting for the system output.
2. Every inner membrane sends its symbols back to the skin, once properly renamed—say, every symbol z in the i-th membrane is sent out as x_i. These symbols form the new state and, at the end of the transition, they are ready to take part in the next one-step evolution of the linear system.

Finally, the P system output must be converted back to the original linear system output. This is made by clearing out, at temporal step n, the factors $k_{\boldsymbol{A}}$, $k_{\boldsymbol{C}}$ and $k_{\boldsymbol{v}}$ from the multiplicity value of every output symbol y_i, here denoted with $|y_i|$:

$$\boldsymbol{u}(n) = \frac{1}{k_{\boldsymbol{A}}^n k_{\boldsymbol{C}} k_{\boldsymbol{v}}^{n+1}} \left|\, |y_1|\, |y_2|\, \ldots\, |y_N| \,\right|^T \tag{22}$$

in which ^{T} denotes transposition.

Although both $\tilde{\boldsymbol{v}}(n+1)$ and $\tilde{\boldsymbol{u}}(n)$ are linear combinations of $\tilde{\boldsymbol{v}}(n)$, nevertheless we must note that the computation of the output is performed by a procedure that differs from the one used for evolving the state. In fact, the latter makes use of membranes in which to store the new state, whereas the former sends the result directly out of the skin hence avoiding the use of additional membranes. Indeed, inner membranes can be even avoided in the production of the new state provided that additional symbols, $z_1, \ldots, z_N$, are added to the P system to keep trace of it. In this case (21) is rewritten as

$$x_j \to z_1^{a_{1j}} \ldots z_N^{a_{Nj}}\, y_{1\,\text{out}}^{c_{1j}} \ldots y_{N\,\text{out}}^{c_{Nj}}, \quad j = 1, \ldots, N, \tag{23}$$

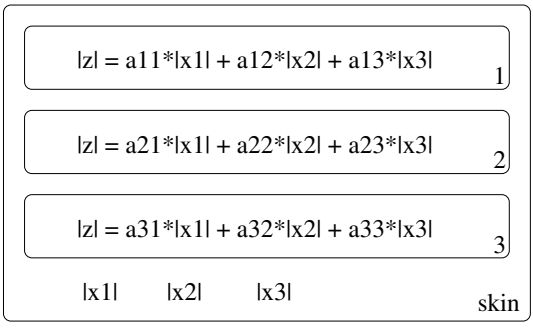

Fig. 5. Graphic representation of the P system proposed in the example. Module operators give the multiplicity of the respective surrounded symbols

meanwhile (again for each j) further rules of the type $z_j \to x_j$ update the state in parallel, by transforming the (previous) new state in the actual state of the system.

We think that the existence of inner membranes, as expressed by (21), puts more emphasis on the system's structural properties and paves the way for strategies aimed at characterizing linearity in P systems containing multiple nested membranes.

As an example, suppose to have

$$\boldsymbol{A} = \begin{vmatrix} 1 & 1/2 & 0 \\ 0 & 1 & 0 \\ 1 & 1/3 & 1 \end{vmatrix} \quad \boldsymbol{C} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{vmatrix}, \quad \boldsymbol{v}(0) = \begin{vmatrix} 0 \\ 1 \\ 0 \end{vmatrix} .$$

First, we compute $k_{\boldsymbol{A}} = 6$, $k_{\boldsymbol{C}} = 1$ and $k_{\boldsymbol{v}} = 1$ in a way that

$$\tilde{\boldsymbol{A}} = \begin{vmatrix} 6 & 3 & 0 \\ 0 & 6 & 0 \\ 6 & 2 & 6 \end{vmatrix}, \quad \tilde{\boldsymbol{C}} = \boldsymbol{C}, \quad \tilde{\boldsymbol{v}}(0) = \boldsymbol{v}(0) .$$

Then, provided V, T, μ, w_{skin}, $w_1, \ldots, w_N$ as above, the rule set is the following:

$$\begin{aligned} R_{\text{skin}} = \{ & x_1 \to z_{\text{in}_1}^6 \, z_{\text{in}_3}^6 \, y_{1\,\text{out}} \, y_{3\,\text{out}}, \\ & x_2 \to z_{\text{in}_1}^3 \, z_{\text{in}_2}^6 \, z_{\text{in}_3}^2 \, y_{2\,\text{out}}, \\ & x_3 \to z_{\text{in}_3}^6 \, y_{3\,\text{out}} \}, \\ R_1 = \{ & z \to x_{1\,\text{out}} \}, \\ R_2 = \{ & z \to x_{2\,\text{out}} \}, \\ R_3 = \{ & z \to x_{3\,\text{out}} \} . \end{aligned}$$

Figure 5 can help the reader in decoding the process.

The absence of further elements in the construct such as, for example, priorities on the rules, proves that such a construct is not universal, as it had to be expected. Despite this, the symbols which are read out of the skin give, at each temporal step n, the linear system solution once it is computed as $\boldsymbol{u}(n) = (1/6^n)\big|\,|y_1|\,|y_2|\,\ldots\,|y_N|\,\big|^T$ according to (22).

5 Conclusion and Future Research Directions

Let us call MA the metabolic algorithm. If E is a system of metabolic equations derived from a set of rewriting rules, then $MA(E,\mu)$ is the dynamics we get with a molarity unit μ. Let us call $[E]_\mu$ the "molar normalization" of equations E which is obtained by replacing every reactivity parameter k in E by $k/\mu^{(t-1)}$ where t is the degree of reactant monomial associated to k. Finally, let us call by $d(E)$ the differential form of equations E which is obtained by replacing in E the Δ finite difference operator by the differential operator d/dt, and the molar quantities by absolute quantities, that is, by putting $\mu = 1$. If $Euler$ is the Euler's approximation method for solving differential equations, the following proposition is easily proved.

Proposition 2. $MA(E,\mu) = Euler(d([E]_\mu))$.

It is very interesting that, in the case of oscillatory phenomena that we studied, especially in the Brusselator reported in [13], we get the following experimental result where $Runge–Kutta$ is a very common and reliable integration method.

Proposition 3. $MA(E,\mu) = Runge–Kutta(d([E]))$.

This result shows the relevance of molar normalization. We plan to develop further experimental and theoretical work for a better understanding of this phenomenon and for improving our metabolic algorithm by means of a more systematic and adaptive use of molar normalization. However, it is important

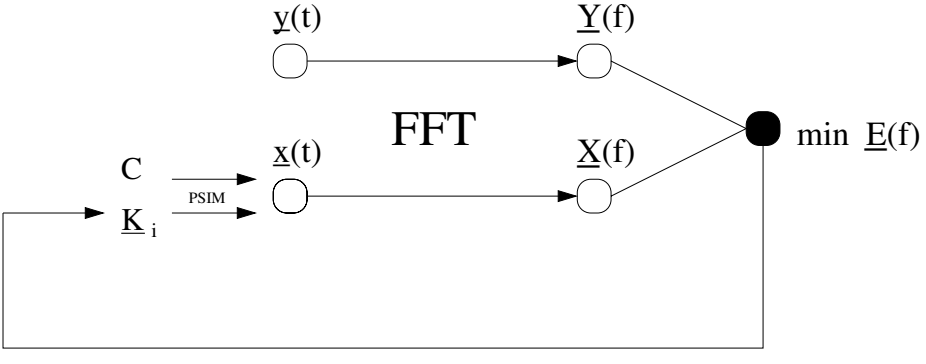

Fig. 6. Schematic of our resolution strategy of the inverse oscillating problem

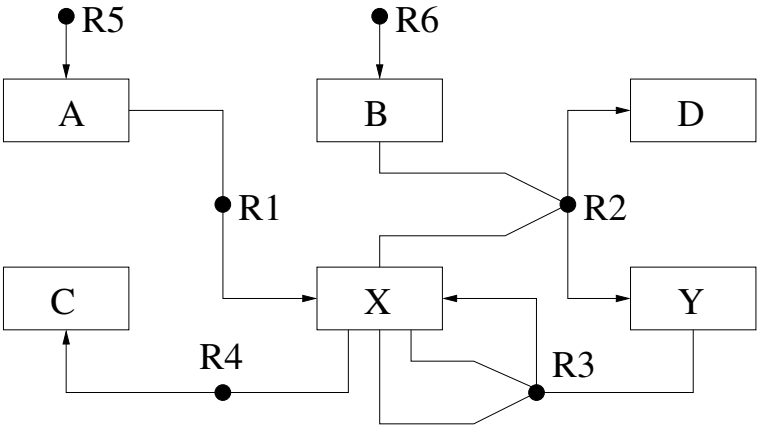

Fig. 7. The neuron-like structure of Brusselator Metabolic Graph

that this metabolic approach seems to be a basis for a reliable direct discrete tool for computing the behavior of P systems. The next step is the extension of this dynamic approach to more complex membrane topologies, and to situations where reaction parameters change in time under the influence of external factors.

Now let us express the rewriting rules of Brusselator with the graph given in Fig. 7. This formulation suggests us a new perspective in P system analysis. First, we could extend this representation to any membrane structure by a suitable use of labels. In this way any membrane system becomes a dynamical network, that is a graph where at each time nodes have a state that depend on the state of other nodes of the graph (nodes and arcs can be added and removed in time). In other words, a membrane system is always related to a sort of "neuron-like" membrane structure, according to Păun's terminology. It is easy to discover that a dynamics associated to some rewriting rules can present oscillatory phenomena only if the relative metabolic graph has (some form) of cycles. But in general finding parameters that ensure some kind of oscillations is not a simple task. In the case of the Brusselator graph, the search space is a vector space of twelve dimensions (six for initial concentrations and six for reactivities). The *Inverse Oscillation Problem* can be stated in the following way: *Given a metabolic graph, find initial concentrations and reactivity parameters that ensure an oscillation of quantities of some given types.*

We are currently working on a strategy of solution of the inverse oscillation problem[1]. As suggested by the name of the problem, we are mainly interested in dynamic oscillating behaviors. The initial data typically consist in a metabolic graph, that provides relations on the system constituents, and in a set of oscil-

[1] In fact the proposed strategy deals with a simplified instance of the problem, in which we know, even roughly, the initial concentrations and moreover we need initial estimates of the reactivity parameters. In this way the solution of the Brusselator's inverse oscillation problem lies within a six-dimensional vector space rather than a twelve one.

lating functions, usually obtained by experiments, that we intend to reproduce. Our goal is to define an automatic procedure that, in the same initial conditions (mainly in terms of reactivity coefficients) realizes the desired oscillating system behavior.

Starting from a periodic signal, in the time domain, we can describe its behavior in a more compact way by analyzing its dual representation in the frequency domain. Let $y(t)$ be the signal to be reproduced and let us denote with $x(t)$ the signal obtained from the simulation. The simulated signal will depend on the reactivity coefficients $k_1, \ldots, k_N$, whose values are our subject of investigation. For this reason, from now on we will denote the simulated signal x as $x(k_1, \ldots, k_N, t)$, where N is the number of unknown reactivity coefficients.

Note that we have to consider a pair of signals y_i, x_i for each one of the different kind of objects present in the system. Let's denote with m the number of different types of elements populating the system under investigation and with n the number of discrete-time samples forming our signals. According to the previous observation we have to deal with a couple of matrices $\underline{y}(t)$ and $\underline{x}(k_1, \ldots, k_N, t)$, sharing the same dimension, that is, $m \times n$.

Our approach starts from the observation that a periodic signal (e.g., a sinusoidal function) is described in a compact way in the frequency domain by means of the *Fourier Transform* operator (e.g. a sine in time becomes an impulse in frequency). This translation from time to frequency domain is implemented very efficiently, thanks to fast implementations of the Fourier Transform, known as *Fast Fourier Transform* or FFT [8].

For the previously described reasons we suggest to perform a Fourier Transform on our signals, $\underline{y}(t)$ and $\underline{x}(k_1, \ldots, k_N, t)$, and this lead to their dual representations $\underline{Y}(f)$ and $\underline{X}(k_1, \ldots, k_N, f)$, where the symbol f reminds that Y and X belong to the frequency domain. The target of our investigation is the minimization of the norm:

$$\underline{E}(f) = ||\underline{Y}(f) \ - \ \underline{X}(k_1, \ldots, k_N, f)||, \tag{24}$$

that is, to calculate

$$\min_{\underline{K}} ||\underline{Y}(f) \ - \ \underline{X}(f)|| = \min_{\underline{K}} \underline{E}(f), \tag{25}$$

where we have defined the norm $||M||$ of a matrix M as the maximum value of the norms of all rows of M.

In order to minimize the distance function $\underline{E}$ we can use several minimization algorithms. The important thing to point out here is that the translation to the frequency domain can be performed very efficiently by FFT algorithms, and the minimization procedure is in general carried out in a more efficient way in frequency rather than in the time domain; this efficiency is gained when $\underline{E}(f)$ turns out to be an $m' \times n'$ matrix where $m' = m$ and $n' < n$, as it happens when the maximum number of pure oscillatory behaviors under investigation (chosen to be equal to $n'/2 - 1$) is likely to be smaller than the number n of temporal samples included in the analysis window of every signal $y_1, \ldots, y_m$.

Note also that, in general, the solution obtained in the frequency domain may result in a phase-shifted time signal, but this is not limiting our point because our interest focuses on the periodical trend of the system as a whole.

In Figure 6 our approach is depicted in a schematic way. The inputs of the method are:

- $\underline{C}$, the initial concentrations of all objects;
- $\underline{K_0}$, initial estimates of reactivities;
- $\underline{y}(t)$, the matrix of signals that we want to reproduce;
- the metabolic graph (not depicted in figure), necessary to describe all relationships between objects populating the system.

Starting from objects' relationships described by the metabolic graph, and by using $\underline{K_0}$ and $\underline{C}$, an implementation of the metabolic algorithm provides the simulated behavior $\underline{x}(t)$. The FFT block translates our signals $\underline{y}(t)$ and $\underline{x}(t)$ in their frequency duals $\underline{Y}(f)$ and $\underline{X}(f)$. We perform a minimization algorithm on $\underline{Y}(f)$ and $\underline{X}(f)$ in order to adjust the vector K_0 into another one, say K_1, on which the metabolic algorithm is applied again and again until the distance between $\underline{Y}(f)$ and $\underline{X}(f)$ falls below a certain threshold τ. If this happens at the iteration cycle $i+1$ then K_i contains the result, otherwise it's desirable to specify a maximum number of cycles after that the procedure terminates without convergence.

Another research topic, related the inverse oscillation problem, is the finding of parameters, possibly defined on the metabolic graphs of P systems, that could have some dynamical relevance. Many of them are suggested by parameters introduced for cellular automata and Kauffman networks, but a lot of experimental work and theoretical analysis has to be developed along this direction.

References

1. F. Bernardini and V. Manca. Dynamical aspects of P systems. *BioSystems*, 70:85–93, 2002.
2. L. Bianco, F. Fontana, G. Franco, and V. Manca. P systems for biological dynamics. In G. Ciobanu, G. Păun, M.J. Pérez-Jiménez, eds., *Applications of Membrane Computing*, Springer-Verlag, Berlin, 2005, to appear.
3. C. Bonanno and V. Manca. Discrete dynamics in biological models. *Romanian J. of Inform. Sc. and Tech.*, 5(1–2):45–67, 2002.
4. R.L. Devaney. *Introduction to Chaotic Dynamical Systems*. Addison-Wesley, 1989.
5. R. Freund. Energy-controlled P systems. In G. Păun and C. Zandron, editors, *Proc. Int. Workshop WMC-CdeA 2002*, number 2597 in Lecture Notes in Computer Science, pages 247–260, Curtea de Arges, Romania, August 2002. Springer.
6. R. C. Hilborn. *Chaos and Nonlinear Dynamics*. Oxford University Press, 2000.
7. D.S. Jones and B.D. Sleeman. *Differential Equations and Mathematical Biology*. Chapman & Hall/CRC, London, 2003.
8. T. Kailath. *Linear Systems*. Prentice-Hall, Englewood Cliffs, 1980.
9. S.A. Kauffman. *The Origins of Order*. Oxford University Press, New York, NY, 1993.

10. C.G. Langton. Computation at the edge of chaos: phase transitions and emergent computation, *Physica D*, **42**, 12, 1990.
11. V. Manca, G. Franco, and G. Scollo. State transition dynamics: basic concepts and molecular computing perspectives. In M. Gheorghe, ed., *Molecular Computational Models: Unconventional Approaches*, Idea Group Inc., 2004.
12. C. Martin-Vide, G. Păun, and G. Rozenberg. Membrane systems with carriers. *Theoretical Computer Science*, 270:779–796, 2002.
13. G. Nicolis and I. Prigogine. *Exploring Complexity, An Introduction*. Freeman and Company, San Francisco, 1989.
14. G. Păun. Computing with membranes. *J. Comput. System Sci.*, 61(1):108–143, 2000.
15. G. Păun *Membrane Computing - An Introduction*. Springer-Verlag, Berlin, 2002.
16. G. Păun and G. Rozenberg. A guide to membrane computing. *Theoretical Computer Science*, 287:73–100, 2002.
17. G. Păun, Y. Suzuki, and H. Tanaka. P systems with energy accounting. *Int. J. Computer Math.*, 78(3):343–364, 2001.
18. Y. Suzuki and H. Tanaka. Chemical oscillation in symbolic chemical systems and its behavioral pattern. In Y. Bar-Yam, editor, *Proc. International Conference on Complex Systems*, pages 0–7, New England Complex Systems Institute, 1997.
19. Y. Suzuki, H. Tanaka. Abstract rewriting systems on multisets and their application for modelling complex behaviours, in G. Paun, M. Cavaliere, and C. Martín-Vide eds., *Brainstorming Week on Membrane Computing, Tarragona, February 5-11 2003*, Tarragona, Feb 5-11 2003.
20. A. Wuensche. Discrete dynamical networks and their Attractor Basins, online journal *Complexity International*, 1998.
21. A. Wuensche. Basins of attraction in network dynamics: A conceptual framework for biomolecular networks, in G. Schlosser and G.P. Wagner, eds., *Modularity in Development and Evolution*, Chicago University Press, 2003.
22. S. Wolfram. *Theory and Application of Cellular Automata*, Addison-Wesley, 1986.