Membrane Computing as a Modeling Framework. Cellular Systems Case Studies

Gheorghe Păun[1] and Francisco José Romero-Campero[2]

[1] Institute of Mathematics of the Romanian Academy
PO Box 1-764, 014700 Bucureşti, Romania
george.paun@imar.ro, gpaun@us.es
[2] Automated Scheduling, Optimisation and Planning Research Group
School of Computer Science and Information Technology
University of Nottingham, Jubilee Campus, Nottingham, NG8 1BB, UK
fxc@cs.nott.ac.uk

Abstract. Membrane computing is a branch of natural computing aiming to abstract computing models from the structure and functioning of the living cell, and from the way cells cooperate in tissues, organs, or other populations of cells. This research area developed very fast, both at the theoretical level and in what concerns the applications. After a very short description of the domain, we mention here the main areas where membrane computing was used as a framework for devising models (biology and bio-medicine, linguistics, economics, computer science, etc.), then we discuss in a certain detail the possibility of using membrane computing as a high level computational modeling framework for addressing structural and dynamical aspects of cellular systems. We close with a comprehensive bibliography of membrane computing applications.

1 Introduction

Membrane computing is a branch of natural computing, the broad area of research concerned with computation taking place in nature and with human-designed computing inspired by nature. Membrane computing abstracts computing models from the architecture and the functioning of living cells, as well as from the organization of cells in tissues, organs (brain included) or other higher order structures such as colonies of cells (e.g., bacteria).

Membrane computing was initiated in 1998 (with the seminal paper published in 2000) and the literature of this area has grown very fast (already in 2003, Thompson Institute for Scientific Information, ISI, has qualified the initial paper as "fast breaking" and the domain as "emergent research front in computer science" – see http://esi-topics.com). Details, in particular, many downloadable papers, including pre-proceedings of yearly workshops and brainstorming weeks on membrane computing, can be found at http://psystems.disco.unimib.it.

The initial goal was to learn from cell biology something possibly useful to computer science, and the area quickly developed in this direction. Several classes

of computing models – called *P systems* – were defined in this context, inspired from biological facts or motivated from mathematical or computer science points of view. In the last few years, a number of applications were reported in several areas – biology, bio-medicine, linguistics, computer graphics, economics, approximate optimization, cryptography, etc. Several software products for simulating P systems and attempts of implementing P systems on a dedicated hardware were reported; also an attempt towards an implementation in bio-chemical terms is in progress.

After very briefly presenting the basic ideas of membrane computing (main types of P systems, main categories of results), we enumerate the domains where membrane computing was used as a modeling framework; then we pass to presenting P systems as a high level computational modeling framework which integrates the structural and dynamical aspects of cellular systems in a comprehensive and relevant way while providing the required formalization to perform mathematical and computational analysis. Several case studies are discussed in some detail.

The paper ends with a comprehensive bibliography, with papers clustered according to the area of applications; at the beginning of the bibliography we also provide several titles of a general interest, giving basic information about membrane computing.

2 A Quick Description of Membrane Computing

The main ingredients of a P system are (i) *the membrane structure*, delimiting compartments where (ii) *multisets of objects* evolve according to (iii) *(reaction) rules* of a bio-chemical inspiration. The rules can process both objects and membranes. Thus, membrane computing can be defined as a framework for devising cell-like or tissue-like computing models which process multisets in compartments defined by means of membranes. These models are (in general) distributed and parallel. When a P system is considered as a computing device, hence it is investigated in terms of (theoretical) computer science, the main issues studied concern the *computing power* (in comparison with standard models from computability theory, especially Turing machines/Chomsky grammars and their restrictions) and the *computing efficiency* (the possibility of using parallelism for solving computationally hard problems in a feasible time). Computationally and mathematically oriented ways of using the rules and of defining the result of a computation are considered in this case. When a P system is constructed as a model of a bio-chemical process, it is examined in terms of dynamical systems, with the evolution in time being the issue of interest, not a specific output.

At this moment, there are three main types of P systems: (i) cell-like P systems, (ii) tissue-like P systems, and (iii) neural-like P systems.

The first type imitates the (eukaryotic) cell, and its basic ingredient is the *membrane structure*, a hierarchical arrangement of membranes (understood as three dimensional vesicles), i.e., delimiting compartments where multisets of objects are placed; the objects are in general described by symbols from a given

alphabet, but also string-objects can be considered; rules for evolving these objects are provided, also localized, acting in specified compartments or on specified membranes. The most common types of rules are multiset rewriting rules (similar to chemical reactions) and transport rules, e.g., symport or antiport rules, inspired by biological processes. The objects not only evolve, but they also pass through membranes (we say that they are "communicated" among compartments). The rules can have several forms, and their use can be controlled in various ways: promoters, inhibitors, priorities, etc. Also the hierarchy of membranes can evolve, e.g., by creating and destroying membranes, by division, by bio-like operations of exocytosis, endocytosis, phagocytosis, and so on.

In tissue-like P systems, several one-membrane cells are considered as evolving in a common environment. They contain multisets of objects, while also the environment contains objects. Certain cells can communicate directly (channels are provided between them) and all cells can communicate through the environment. The channels can be given in advance or they can be dynamically established – this latter case appears in so-called *population P systems*. In the case when the cells are simple, of a limited capacity (as the number of objects they contain or of rules they can use), we obtain the notion of *P colony*.

Finally, there are two types of neural-like P systems. One is similar to tissue-like P system in that the cells (neurons) are placed in the nodes of an arbitrary graph and they contain multisets of objects, but they also have a *state* which controls the evolution. Another variant was recently introduced, under the name of *spiking neural P systems*, where one uses only one type of objects, the *spike*, and the main information one works with is the distance between consecutive spikes.

From a theoretical point of view, P systems are both powerful (most classes are Turing complete, even when using ingredients of a reduced complexity – a small number of membranes, rules of simple forms, ways of controlling the use of rules directly inspired from biology are sufficient for generating/accepting all sets of numbers or languages generated by Turing machines) and efficient (many classes of P systems, especially those with enhanced parallelism, can solve computationally hard problems – typically **NP**-complete problems, but also harder problems, e.g., **PSPACE**-complete problems – in feasible time – typically polynomial). Then, as a modeling framework, membrane computing is rather adequate for handling discrete (biological) processes, having many attractive features: easy understandability, scalability and programmability, inherent compartmentalization and non-linearity, etc.

The cell-like P systems were introduced first and their theory is now very well developed; tissue-like P systems have also attracted a considerable interest, while the neural-like systems, mainly under the form of spiking neural P systems, were only recently investigated. Correspondingly, most applications use cell-like P systems, several of them also involve tissue-like P systems, but very few research efforts were paid to using spiking neural P systems in applications (although several suggestions from "classic" neural computing are obvious – for instance, trying applications to pattern recognition).

3 Applications of Membrane Computing

As it is natural, membrane computing was first and much more intensively used as a modeling framework for addressing biological processes. In this respect, P systems can be seen as models approaching reality at the micro level, where "reactants" and "reactions" can be known individually, opposed to the macro approach (e.g., by means of differential equations), which deals with populations of reactants which can be considered infinite (large enough to be better approximated by infinity rather than by finite, discrete sets). We have mentioned above several attractive features of P systems as models of biological processes: inherent compartmentalization, easy extensibility, direct understandability (by the biologist), easy programmability, non-linear behavior. P systems are particularly suitable, if not the "obligatory", in the cases when we have to deal with a reduced number of object or with slow reactions – and this is the case in a large number of biological processes, especially related to networks of pathway controls, genetic processes, protein interactions.

It should be noted here that many of the reported applications in biology and bio-medicine are of a *postdiction* type: one takes a biological process, as described already in biological publications, one writes a membrane computing model of it, then one writes a program or one takes a program existing in the literature (for instance, at the membrane computing P page), and one simulates the model by means of this program, comparing the results with those already known from literature (based on differential equations models or on experimental results). There also are a few papers of a *prediction* type, involving biological research hypotheses and thus returning really new information to biologist, not yet known through other means. Of course, this latter direction of research is of much more interest, but the former one is still useful/necessary, because it checks the models and the programs, thus validating the tools for prediction applications.

Similar from many points of view to the bio-chemical reality is the economic reality (e.g., at the market level), where compartments can be defined where various "objects" (good, parts of goods, money, working time, contracts, and so on and so forth) "react" according to well-specified rules. There also is an important difference between bio-chemistry and economic interaction: in the latter case, the behavior of agents is not purely probabilistically controlled, e.g., depending on the multiplicity of "reactants" (stoichiometry), but the psychological factor is also important. Anyway, this direction of research needs further efforts, but it is much favored by the fact that multi-agent computer based approaches (simulations) are more and more used in economics, somewhat contrasted to the fact that "exact" methods, e.g., of the kind provided by the classic operational research, seem to be less applicable to non-trivial, complex economic phenomena.

It was also used in economics the language of membrane computing, the mathematical and the graphical one. This is much more visible in the applications to linguistics.

The applications to computer science are rather diverse. A good example is that of computer graphics: some papers are rather practical (somewhat complementing the applications of Lindenmayer systems in computer graphics, one add

membrane distribution to known approaches, with good results in terms of efficiency), many others are of a theoretical type (P systems with two-dimensional objects or generating in well specified ways picture languages, mainly arrays). Whether or not the second direction of research may be considered as dealing with "real" applications is debatable in this moment, that is why only a few titles of this kind are mentioned.

Similar discussions can be made in what concerns applications in sorting/ranking, cryptography, modelling/simulating circuits, parallel architectures, etc. They mainly show the great expressivity power of P systems, their versatility, but the results of the mentioned papers are not yet of a direct practical interest in computer science.

A promising exception to the previous remark is the use of membrane computing ideas in evolutionary computing. The so-called membrane algorithms introduced by T.Y. Nishida and much investigated by L. Huang and his collaborators, seem to be rather efficient and useful, both in terms of the convergence speed, the quality of the provided solutions, and the average and the worst solutions (which proves that such approaches are reliable and effort-saving).

We have ended with a short list of papers dealing with "other applications" (such as simulating ambient calculus or other well-know models and paradigms from computer science), but we have not included papers from the large literature dealing with polynomial (often even linear) solutions to computationally hard problems (typically, **NP**-complete, but also **PSPACE**-complete problems). The bibliography from `http://psystems.disco.unimib.it`, `http://ppage.psystems.eu` contains many papers (and PhD theses) with this subject, as well as further titles pertaining to all sections of the paper bibliography.

4 Looking for Cell Models

We pass now to discussing in more details and illustrating the issue of using P systems as models for cell systems, starting with a general presentation of the related efforts and directions of research.

The complexity and apparent messiness of interactions in cellular systems makes necessary the development of models able to provide a better understanding of their dynamics and properties. The use of models is intrinsic to any scientific activity. A model is an abstraction of the *real world* onto a mathematical/computational formalism which highlights some key features while ignoring others that are assumed to be irrelevant. Therefore, a model should not be seen or presented as a representation of the truth, but instead as a statement of our current knowledge of the phenomenon under research. A model is even useful when proved to disagree with real data, since it shows that our current hypotheses do not match the reality and it helps experimentalists to decide which experiments are necessary to advance understanding.

Although biologists are familiar with modeling, quantitative computational mathematical models have lain outside the mainstream due to the lack of

techniques from both experimental and theoretical/computational sides. Nonetheless, at the end of the last century extraordinary advances were achieved in both computer science and biology reaching the point where each one can benefit from the other one. In this respect, a new field is emerging which integrates biology, mathematics and computer science, *systems biology*. Systems biology constitutes a purely interdisciplinary field aiming to merge classical biology, computer science and mathematics. Ideally it will produce a new generation of scientists able to understand and apply concepts, techniques and sources of inspirations coming from any of the three classical fields enumerate above into any of the others.

The new advances in cellular biology have made possible the enumeration of the components of cellular systems on a large scale. Initially, a reductionist approach was taken with the aim of understanding the functioning of cells by identifying and characterizing each one of their individual constituents. This approach did not produce the expected knowledge uncovering the fact that the functioning of cellular systems arises as an emergent process from the interactions between their different components. The young field of systems biology presents a systemic methodology whose goal is to deepen the understanding of cellular level dynamics as emergent properties arising over time from the interactions between different systems made of molecular entities. Systems biology focuses on the nature of the interactions and links that connect cellular systems and the functional states of the networks that result from the assembly of such links. Due to the complexity of these connections and to the huge amount of data produced by experimentalists computational/ mathematical modeling, simulation and analysis are essential techniques in this field.

In a cell system biology model it is desirable to have at least four properties: relevance, understandability, extensibility and computability [22].

• Relevance: A model must be relevant capturing the essential features of the phenomenon investigated. It should present a unifying specification of the different components that constitute the system, the interactions between them, their dynamic behavior as well as the physical structure of the system itself.
• Understandability: The abstract formalisms used to model cellular systems should correspond well to the informal concepts and ideas from molecular biology. A model should provide a better and integrated understanding of the real cellular system instead of producing a complicated and hard to decipher formalism.
• Extensibility: In a cellular model we should be able to identify easily its different components so they can be rearranged, duplicated, composed, etc. in an easy way to produce other models. Models of cellular systems should also be extensible to higher levels of organizations, like colonies, tissues, organs, organism, etc. Our knowledge of cellular systems continues to expand and change. In order to handle this continuous supply of new discoveries a model should be adapted easily to incorporate new information.
• Computability and Mathematical tractability: It should be possible to implement a model in a computer so that we can realize it to study the dynamics

of the system by manipulating experimental conditions in the model without having to perform complex and costly experiments. The computability of the model also allows us to apply analytical techniques on it to infer qualitative and quantitative properties of the system in an automatic way. In this respect, the model should be mathematically tractable.

Our research towards using P systems as a computational/mathematical modeling framework for the specification and analysis of cell system biology aims to produce a high level formalism which integrates the structural and dynamical aspects of cellular systems in a comprehensive and relevant way while providing the required formalization to perform mathematical and computational analysis. To this aim, we introduce stochastic P systems as a modeling framework for cell systems biology models, with a detailed methodology for the specification of the components of cellular systems and of the most important molecular interactions in living cells. Then we propose the analysis of P system models using the probabilistic model checker PRISM. The lac operon regulation system is studied as a case study to illustrate this modeling approach.

5 Related and Previous Work

Modeling of cellular systems is currently subject to very intensive research. There are multiple approaches ranging from graphical representations to sophisticated computational and mathematical formalisms. Here we cannot present an exhaustive enumeration of the different modeling methodologies and will only discuss roughly those modeling approaches closely related to the work presented in this paper.

5.1 Ordinary Differential Equations

Ordinary differential equations (ODEs) constitute the most widely used approach in modeling molecular interaction networks in cellular systems. Writing and solving numerically a system of ODEs describing a reaction network can be largely automated. Each molecular species is assigned a continuous variable which represents its concentration. For each molecular species, a differential equation is written to describe its concentration change over time due to the reactions with other species in the system. The rate of each reaction is represented using a *kinetic law*, which commonly depends on one or more rate constants. *Exponential decay law*, *mass action law*, *Michaelis-Menten dynamics* and *Hill dynamics* are the most widely used kinetic laws. In this respect models based on ODEs are referred to as *macroscopic* since they do not represent mechanistic aspects of the interactions between molecules they focused on the modeling of the macroscopic effect of the molecular interactions using specific kinetic laws.

Although, ODEs have been used successfully to model kinetics of conventional macroscopic chemical reactions the realization of a reaction network as a system of ODEs is based on two assumptions:

1. First, cells are assumed to be well stirred and homogeneous volumes. Whether or not this is a good approximation depends on the time and space scales involved. In bacteria molecular diffusion is sufficiently fast to mix compounds. The time needed for a protein to diffuse throughout a bacterium size volume is a few seconds. Therefore if we are interested in transcription/translation processes (minutes), cell cycle (hours), circadian rhythms (one day), etc. the well stirred volume assumption is justified in bacteria. This is not the case in eukaryotic cells where the volume is considerably bigger and it is structured in different compartments.
2. The second basic assumption is that chemical concentrations vary continuously over time in a deterministic way. This assumption is valid if the number of molecules of each specifies in the reaction volume (the cell or the subcellular compartment) are sufficiently large and the reactions are fast.

Therefore in cellular systems with low number of molecules and slow molecular interactions the application of the classical macroscopic and deterministic approach based on ODEs is questionable. Instead *mesoscopic*, *discrete* and *stochastic* approaches are more suitable [11]. In this last approach the most relevant individual parts of the system are taken into account but details like position and momenta are neglected. One focuses on the number of individual components of the system, the statistics of the events and how often they take place. The mesoscopic approach is more tractable than the microscopic approach while keeping more relevant information than the macroscopic approach.

5.2 Computational Modeling

The complexity of mesoscopic, discrete and stochastic models makes necessary the use of computers to help to analyze them. Until recently the majority of computational models were implemented in custom programs and published as statements of the underlying mathematical model. No computational formalism was explicitly used to model and simulate cellular systems. Nevertheless, to be useful a computational model must be presented within a well defined, consistent and formal framework. Following this line, recently several formal computational frameworks has been proposed to model cellular systems. Here we will only discuss briefly Petri nets and process algebra as they are closely related to P systems.

- **Petri Nets** are a mathematical and computational tool for modeling and analysis of discrete event systems typically with a concurrent behavior. They offer a formal way to represent the structure of the interactions in a discrete event system, simulate its behavior, and prove certain properties of the system [20]. Roughly speaking a Petri net is a directed graph formed by two kinds of nodes called places and transitions. Directed edges, called arcs, connect places to transitions, and transitions to places. A non-negative integer number of tokens is assigned to each place. Tokens move from one place to another one connected to it through a transition when this transition fires.

A system of interacting molecules can be modeled using Petri nets by representing each molecular species as a different place and each biochemical transformation as a different transition. The number of tokens inside a place is used to specify the number of molecules of the corresponding molecular species [20]. Within this framework only qualitative analysis can be performed, in order to be able to develop quantitative analysis *Stochastic Petri Nets (SPN)* were introduce in [12]. In SPNs each transition is associated with a rate parameter used to compute a time delay following an negative exponential distribution. Then transitions fire according to these time delays.

- **Process algebra** is a family of formalisms for the description of interactions, communications, and synchronization between a collection of concurrent processes. Algebraic laws are provided allowing process descriptions to be manipulated and analyzed. The π-calculus is one of the most widely used process algebras in cellular modeling. It was introduced as a formal language to describe mobile concurrent processes that interact through communication channels [16]. It is now a widely accepted model for interacting systems with dynamically evolving communication topology. The π-calculus has a simple semantics and a tractable algebraic theory. Starting with atomic actions and simpler processes, complex processes can be constructed in specific ways.

  In the $\pi-$calculus formalism a system of interacting molecular entities is modeled by a system of interacting processes which communicate through complementary communication channels. Each molecular species or domain is represented by a different process. The number of copies of each process is used to specified the number of molecules. Molecular interactions are described using complementary communication channels [22].

Although these computational frameworks captures some of the information regarding cellular systems and their components, none fully integrates the dynamics and structural details of the systems. One of the main points which is neglected is the key role played by membranes and compartmentalization in the structure and functioning of living cells. There have been several attempts in specifying and simulating membranes and compartments in the process algebra [5,21]. Nevertheless, it has been discussed that the models developed using process algebra can be obscure, non intuitive and difficult to understand [22]. In this work we aim to develop a formal modeling framework based on P systems which explicitly represent in a relevant and comprehensible manner the key role played by membranes.

6 Stochastic P Systems for Cell Systems Biology Models

Let us stress once again that P systems, according to the original motivation, were not intended to provide a comprehensive and accurate model of the living cell, rather, to explore the computational nature of various features of biological membranes. Although most research in P systems concentrates on computational

powers, recently they have been used to model biological phenomena within the framework of computational systems biology presenting models of oscillatory systems [8], signal transduction [17], gene regulation control [24], quorum sensing [23] and metapopulations [18].

6.1 P System Specifications and Models

In order to develop a modeling framework based on P systems a variant has been proposed to formalize the specification of cellular systems, the parameters associated with a specification and the models obtained from specifications by instantiating their parameters with specific values [18].

In what follows the main definitions used in this work are presented. First, we introduce *P system specifications* which will constitute the main structure used to analyze particular cellular systems. A set of *parameters* is identified from the components of a P system specification. Then, the basic definition of P system specifications is extended to introduce *P system models*. Given a possible sets of values for the parameters of a P system specification, a P system model is obtained by instantiating the set of parameters using the given parameter values.

Definition 1 (P system Specification)
A P system specification is a construct:

$$\Pi = ((\Sigma_{obj}, \Sigma_{str}), L, \mu, M_1, M_2, \ldots, M_n, (R_{l_1}^{obj}, R_{l_1}^{str}), \ldots, (R_{l_m}^{obj}, R_{l_m}^{str}))$$

where:
• $(\Sigma_{obj}, \Sigma_{str})$ *are finite alphabets of symbols. The symbols from* Σ_{obj} *represent individual objects whereas the symbols from* Σ_{str} *represent objects that can be arranged to form strings.*
• $L = \{l_1, \ldots, l_m\}$ *is a finite alphabet of symbols representing labels for the compartments and identifying compartment types*[1].
• μ *is a membrane structure containing* $n \geq 1$ *membranes identified in a one to one manner with values in* $\{1, \ldots, n\}$ *and labeled with elements from* L.
• $M_i = (l_i, w_i, s_i)$, *for each* $1 \leq i \leq n$, *is the initial configuration of membrane* i *with* $l_i \in L$, *the label of this membrane,* $w_i \in \Sigma_{obj}^*$ *a finite multiset of individual objects and* s_i *a finite set of strings over* Σ_{str}. *A multiset of objects, obj is represented as* $obj = o_1 + o_2 + \cdots + o_m$ *with* $o_1, \ldots, o_m \in \Sigma_{obj}$. *Strings are represented as follows* $\langle s_1.s_2.\cdots.s_i\rangle$ *where* $s_1, \ldots, s_i \in \Sigma_{str}$.
• $(R_{l_t}^{obj}, R_{l_t}^{str})$ *are finite sets of rewriting rules associated with compartments of the type represented by the label* $l_t \in L$. *More specifically:*

– *The rules in* $R_{l_t}^{obj} = \{r_1^{obj,l_t}, \ldots, r_{k_{obj,l_t}}^{obj,l_t}\}$, *for each* $1 \leq t \leq m$, *are multiset rewriting rules of the following form:*

$$r_j^{obj,l_t} : \ obj_1\,[\,obj_2\,]_l \overset{c_j^{obj,l_t}}{\longrightarrow} obj_1'\,[\,obj_2'\,]_l$$

[1] Compartments with the same label will be considered of the same type and thus the same set of rules will be associated with them.

with $obj_1, obj_2, obj'_1, obj'_2$ some finite multisets of objects from Σ_{obj} and l a label from L. These rules are multiset rewriting rules that operate on both sides of membranes, that is, a multiset obj_1 placed outside a membrane labeled by l and a multiset obj_2 placed inside the same membrane can be simultaneously replaced with a multiset obj'_1 and a multiset obj'_2, respectively.

– *The rules in $R_{l_t}^{str} = \{r_1^{str,l_t}, \ldots, r_{k_{str,l_t}}^{str,l_t}\}$, for each $1 \leq t \leq m$, are rewriting rules on multisets of strings and objects of the following form:*

$$r_j^{str,l_t} : [\, obj + str \,]_l \xrightarrow{c_j^{str,l_t}} [\, obj' + str'_1; str'_2 + \cdots + str'_s \,]_l$$

with obj, obj' multisets of objects over Σ_{obj} and $str, str'_1, \ldots, str'_s$ strings over Σ_{str}. These rules operate on both multisets of objects and strings. The objects obj are replaced by the objects obj'. Simultaneously a substring str is replaced by str'_1 whereas the strings $str'_2 + \cdots + str'_s$ are produced to form part of the content of the compartment.

Note that a constant, c_j^{obj,l_t} or c_j^{str,l_t}, is associated specifically with each rule. This constant will be referred to as stochastic constant and will be used to compute the propensity of the rule.

Definition 2 (Parameters)
Given a P system specification $\Pi = ((\Sigma_{obj}, \Sigma_{str}), L, \mu, M_1, \ldots, M_n, (R_{l_1}^{obj}, R_{l_1}^{str}), \ldots, (R_{l_m}^{obj}, R_{l_m}^{str}))$ the set of parameters $\mathcal{P}(\Pi) = (\mathcal{M}_0(\Pi), \mathcal{C}(\Pi))$ consists of:

1. *The initial multisets $\mathcal{M}_0(\Pi) = (M_1, \ldots, M_n)$ associated with the compartments.*
2. *The stochastic constants $\mathcal{C}(\Pi) = (c_j^{obj,l_t}, c_{j'}^{str,l_t})$ for $1 \leq j \leq k_{obj,l_t}$, $1 \leq j' \leq k_{str,l_t}$ and $1 \leq t \leq m$, associated with the rewriting rules in $(R_{l_1}^{obj}, R_{l_1}^{str}), \ldots, (R_{l_m}^{obj}, R_{l_m}^{str})$.*

Definition 3 (P system Model)
Let Π be a P system specification with parameters $\mathcal{P}(\Pi) = (\mathcal{M}_0(\Pi), \mathcal{C}(\Pi))$ and $(\mathbb{M}_0, \mathbb{C})$ a family of possible values for the initial multisets $\mathcal{M}_0(\Pi)$ and for the stochastic constants $\mathcal{C}(\Pi)$. A family of P system models, $\mathcal{F}(\Pi; \mathbb{M}_0, \mathbb{C})$, is obtained from Π and $(\mathbb{M}_0, \mathbb{C})$ by instantiating the parameters $\mathcal{P}(\Pi)$ using values from $\mathbb{M}_0$ and $\mathbb{C}$.

Hence given $(\mathbb{M}_0, \mathbb{C})$ sets of possible values for the parameters $\mathcal{P}(\Pi)$ specific values $(M_1^0, \ldots, M_n^0) \in \mathbb{M}_0$ and $(c_{j,0}^{obj,l_t}, c_{j',0}^{str,l_t}) \in \mathbb{C}$ can be selected to obtain a P system model $(\Pi; (M_1^0, \ldots, M_n^0), (c_{j,0}^{obj,l_t}, c_{j',0}^{str,l_t})) \in \mathcal{F}(\Pi; \mathbb{M}_0, \mathbb{C})$. In this way a family of P system models $\mathcal{F}(\Pi; \mathbb{M}_0, \mathbb{C})$ sharing the same P system specification can be used to study the behavior of a particular cellular system specified by Π under the different initial conditions collected in $\mathbb{M}_0$ and study the sensitivity of the system for the different rule constants in $\mathbb{C}$.

6.2 Stochastic P Systems and Gillespie's Kinetics Theory

At the microscopic level of functioning of cellular processes the interactions between molecules follow the laws of physics. A fundamental result of theoretical statistical physics is the famous $\sqrt{n}$ law, which states that the noise in a system is inversely proportional to the square root of the number of particles. Therefore, systems with a low number of molecules show high fluctuations and the application of the classical deterministic and continuous approaches is questionable. Mesoscopic, discrete and stochastic approaches are more accurate under these circumstances. In this subsection we present a stochastic extension of the original membrane computing framework using Gillespie's kinetics theory.

In the original approach of membrane computing P systems evolve in a non deterministic and maximally parallel manner. All the objects in every membrane that can evolve according to any rule must evolve. This produces a semi-quantitative framework that takes into account the discrete character of the molecular population and the role played by membranes in the structure and functioning of living cells. Although such coarse abstraction has been proved to achieve some success [2,3], this approach fails to model quantitative aspects that are key to the functioning of many cellular systems. Specifically the non deterministic and maximally parallel approach produces the following two inaccuracies:

1. Reactions do not occur at a correct rate.
2. All time steps are equal and do not represent the time evolution of the real cellular system.

These two problems are interdependent and must be addressed when devising a relevant modeling framework for cellular systems as it has been done in other computational approaches [12,22].

In the field of membrane computing, the discrete aspect of the different components as well as the distributed and compartmentalized character of the structure, where the computation takes place, are fundamental. This is not the case with the non deterministic and maximal parallel semantics as have been studied in different variants [7,10]. In this section the original approach will be replaced with a strategy based on Gillespie's theory of stochastic kinetics [11].

To provide P systems with a stochastic extension a constant c is associated with each rule. This constant depends only on the physical properties of the molecules involved in the reaction described by the rule and on other physical parameters of the system like temperature. It is used to compute the propensity of each rule which in turn determines the probability and time needed to apply the rule.

The starting point consists of treating each compartment, delimited by a membrane, as a well mixed and fixed volume where the classical *Gillespie algorithm* is applied. Given the state of a compartment i, $M_i = (l_i, w_i, s_i)$, and the sets of rules associated with it, $R_{l_i}^{obj}$ and $R_{l_i}^{str}$, the next rule to be applied and its waiting time is computed according to Gillespie algorithm:

1. Compute for each rule r_j associated with the compartment its propensity, $a_j(M_i)$, by multiplying the stochastic constant associated specifically with the rule by the number of distinct possible combinations of the objects and substring present on the left-side of the rule with respect to the current contents of the membranes involved in the rule.
2. Compute the sum of all propensities:

$$a_0(M_i) = \sum_{r_j \in (R_{l_i}^{obj} \cup R_{l_i}^{str})} a_j(M_i)$$

3. Draw two random numbers r_1 and r_2 from the uniform distribution in the unit-interval, and select τ_i and j_i according to

$$\tau_i = \frac{1}{a_0(M_i)} \ln\left(\frac{1}{r_1}\right)$$

$$j_i = \text{the smallest integer satisfying } \sum_{j=1}^{j_i} a_j(\mathbf{x}) > r_2 a_0(M_i)$$

This discrete-event simulation algorithm, usually referred to as Gillespie algorithm or SSA (Stochastic Simulation Algorithm), has the nice properties that it simulates every reaction event and is exact in the sense that it generates exact independent realizations of the underlying stochastic kinetic model. Nevertheless, it should be emphasized that Gillespie algorithm was developed for a single, well mixed and fixed volume or compartment. In contrast, in P systems we have a hierarchical structure defining different compartments with specific rules. In what follows we present an adaptation of the Gillespie algorithm that can be applied in the hierarchical and compartmentalized structure of a P system model. This will be referred to as *Multi-compartmental Gillespie algorithm.*

Next, a detailed specification of this algorithm is presented:

- **Initialization**
  - set time of the simulation $t = 0$;
  - for each membrane i compute a triple (τ_i, j_i, i) by using the procedure described before; construct a list containing all such triples;
  - sort this list of triples (τ_i, j_i, i) in increasing order according to τ_i;
- **Iteration**
  - extract the first triple, $(\tau_{i_0}, j_{i_0}, i_0)$ from the list;
  - set time of the simulation $t = t + \tau_{i_0}$;
  - update the waiting time for the rest of the triples in the list by subtracting τ_{i_0};
  - apply the rule $r_{j_{i_0}}$ in membrane i_0 only once changing the number of objects and sites in the membranes affected by the application of the rule;
  - for each membrane i' affected by the application of the rule remove the corresponding triple $(\tau_{i'}, j_{i'}, i')$ from the list;

- for each membrane i' affected by the application of the rule $r_{j_{i_0}}$ re-run the Gillespie algorithm for the new context in i' to obtain $(\tau'_{i'}, j'_{i'}, i')$, the next rule $r_{j'_{i'}}$, to be used inside membrane i' and its waiting time $\tau'_{i'}$;
- add the new triples $(\tau'_{i'}, j'_{i'}, i')$ in the list and sort this list according to each waiting time and iterate the process.

- **Termination**
  - Terminate simulation when time of the simulation t reaches or exceeds a preset maximal time of simulation.

It is worth noting that this is a local algorithm in the sense that all computations only consider the content and rules of a single compartment. The only remaining global computation is the location of the index of the smallest waiting time, which could be improved by keeping all reaction times in an *indexed priority queue*. The advantage of having local computations is that the algorithm is easily implemented in an event-driven object-oriented programming style, such an implementation could be multithreaded on a hyper-threading machine and would also lend itself to full message-passing implementation on a parallel computing cluster.

There exists a different well established approach to modeling cell systems in membrane computing, based on the so called *Metabolic Algorithm* [4]. This algorithm keeps maximal parallelism as the strategy for the evolution of their models. Nonetheless they use rules of the form $a \to a$, called *transparent rules*, that have no effect on the state of the system, in order to bound the number of applied rules that actually change the system. Specific functions, called *reaction maps*, defined ad hoc, are also associated with rules to represent the reactions rates. By doing this the first of the two problems presented before is somehow solved; nevertheless the real evolution time of the system is not treated in this approach. Finally, the *Metabolic Algorithm* is deterministic and so its applicability in certain cell systems suffers from the same drawbacks as other deterministic approaches like ODEs. The relationship between this approach and ODEs has been studied in [9].

Another stochastic approach in P systems has been proposed in [18], dynamical probabilistic P systems. This approach also keeps maximal parallelism and uses transparent rules to bound the number of effective rule applications. The non determinism is replaced by a probabilistic strategy which associates probabilities with the rules depending on the content of membranes. Nevertheless, this approach does not represent the real evolution time of the system as in the metabolic algorithm.

7 P System Specifications of Cellular Systems

Most modelling approaches in systems biology are formalisms coming from different sources of inspirations not related to biology. For example, the π-calculus was introduced to specify mobile concurrent processes that interact

through communication channels [16]. In contrast, P systems are inspired directly from the functioning and structure of the living cell. Therefore, the concepts in P systems are more similar to those used in molecular cell biology than the abstractions of other formalisms. This feature of P systems is key to produce relevant, comprehensive and integrative specifications of the different cellular components.

In this section, we present some principles for the specification of cellular systems in P systems. More specifically we will describe some ideas of how to describe cellular regions and compartments, protein-protein interactions and gene expression control.

7.1 Specification of Cellular Compartments

As mentioned previously, membranes play a key role in the functioning and structural organization of both prokaryotic and eukaryotic cells. The key differential feature of P systems with respect to other discrete, mesoscopic and stochastic approaches is the so called *membrane structure* which provides an explicit description of the compartmentalization in the structural organization of cells. The specification of compartmentalization has been addressed in P systems in different systems, for instance selective uptake of molecules from the environment [24], signalling at the cell surface [6] and colonies of interacting bacteria which communicate by sending and receiving diffusing signals [23].

In the specification of compartments in cellular systems it is necessary to consider two distinct and relevant regions:

1. The compartment surface where a set of proteins, which control the movement of molecules and detect signals, are located.
2. The lumen or aqueous interior space where a characteristic complement of proteins interact to carry out specific functions.

In our P system modeling framework, membranes are used to define the relevant regions in cellular systems. Therefore they do not always correspond with real cell membranes although normally they do. According to this idea, in this work two different membranes will be used to specify the two relevant regions associated with a cellular compartment:

1. A first membrane will represent the compartment surface. In the region defined by this membrane the objects describing molecules associated with the compartment surface will be located. The processes involving molecular transport and cell signalling will be represented by rules which will also be associated with this region.
2. A second membrane will describe the aqueous interior of the compartment and thus it will be embedded inside the previous one. The multiset of objects and strings specifying the proteins and other molecules located in the lumen of the compartment will be placed in the region defined by this membrane. The rules describing the molecular interactions taking place inside the compartment are also associated with this membrane.

In cases where the compartment surface does not play a crucial role the first membrane is omitted and only one membrane defining the compartment interior is used.

7.2 Specification of Protein-Protein Interactions

Large and complex networks of interacting molecular entities are responsible for most of the information processing within living cells. Here we aim to provide a comprehensive and relevant P system modeling schema for the most important protein-protein interactions that take place in living cells. More specifically we will focuss on the formation and dissociation of complexes and on the fundamental processes of communication and transport between different compartments in cellular systems.

The theoretical and experimental description of protein-protein interactions is related to the field of chemical kinetics. A primary objective in this area is to determine the propensity or probability of a protein interaction, in order to describe the rate at which reactants are transformed into products. In this section every P system schema for protein-protein interactions is presented together with the propensities associated with each rule. These propensities are computed according to Gillespie's theory of stochastic kinetics [11].

• **Transformation and Degradation:** A molecule a can react to produce another molecule b or it can be degraded by the cell machinery.

In P system specifications, transformation and degradation are represented using the rewriting rules in the schema (1). In these rules the object a is replaced with the object b or is simply removed in the case of degradation. The compartment type where the molecules are transformed or degraded is also specified using square brackets with a label l. A constant c is associated with the rule so that its propensity [2] can be computed.

$$\begin{array}{l} r_1 : [\ a\ ]_l \xrightarrow{c} [\ b\ ]_l \\ r_2 : [\ a\ ]_l \xrightarrow{c} [\ \ ]_l \end{array} \qquad prop(r_i) = c \cdot |a| \quad i = 1, 2 \tag{1}$$

• **Complex Formation and Dissociation:** Two molecules, a and b, can collide and stick to produce a complex c. Once a complex has been formed it can dissociate back into its components, d and e which could have changed as a consequence of the interaction.

In biochemistry, these reactions are referred to as complex formation, more specifically heterodimer formation when $a \neq b$ and homodimer formation when $a = b$; and complex dissociation. In P system specifications, complex formation and dissociation reactions are specified using the rewriting rules in the schema (2) which take the name of the reactions they represent. In the complex formation rule, r_{cf}, the objects a and b, representing the corresponding molecules, are replaced with the object c, representing the complex. In the same manner, in the complex dissociation rule, r_{cd}, the object c is replaced with the objects d

[2] In this work $|a|$ will be used to represent the number of molecules a.

and e. The compartment type in which the reactions take place is specified using square brackets and a label l.

$$\begin{array}{ll} r_{cf} : [\ a + b\ ]_l \xrightarrow{c_{cf}} [\ c\ ]_l & prop(r_{cf}) = \begin{cases} c_{cf} \cdot |a||b| & \text{if } a \neq b \\ c_{cf} \cdot \dfrac{|a|(|a|-1)}{2} & \text{if } a = b \end{cases} \\ r_{cd} : [\ c\ ]_l \xrightarrow{c_{cd}} [\ d + e\ ]_l & prop(r_{cd}) = c_{cd} \cdot |c| \end{array} \tag{2}$$

• **Diffusion in and out:** Small molecules can readily move by simple passive diffusion across membranes without the aid of transport proteins and without the consumption of any metabolic energy.

The rewriting rules in (3) constitute a P system specification for diffusion in and out of a compartment. This compartment is represented by square brackets with a label l. For diffusion in the object a is moved from the compartment surrounding compartment l inside the region defined by it. Viceversa for the case of diffusion out from compartment l.

$$\begin{array}{ll} r_1 : a\ [\ ]_l \xrightarrow{c_{in}} [\ a\ ]_l & prop(r_1) = c_{in}|a| \\ r_2 : [\ a\ ]_l \xrightarrow{c_{out}} a\ [\ ]_l & prop(r_2) = c_{out}|a| \end{array} \tag{3}$$

• **Binding and Debinding:** One of the key steps in the process of converting signals into cellular responses, *signal transduction*, is the binding of signalling molecules to structurally complementary sites on the extracellular or membrane-spanning domains of receptors leading to their activation.

In P system specifications, the binding and debinding of a ligand to its receptor, located on the cell surface, is specified using the rewriting rules in (4). For the binding rule, the object a representing the ligand is placed outside the compartment representing the cell surface, square brackets with label l. The receptor is specified using the object b placed inside the square brackets. These two objects are replaced with the object c, the complex receptor-ligand, inside the square brackets which represent the compartment surface. The debinding reaction is specified by replacing the object c, inside the square brackets, with the object d, representing the ligand, outside the square brackets and the object e, representing the free receptor, inside them.

$$\begin{array}{ll} r_1 : a\ [\ b\ ]_l \xrightarrow{c_{lb}} [\ c\ ]_l & prop(r_1) = c_{lb}|a||b| \\ r_2 : [\ c\ ]_l \xrightarrow{c_{ld}} d\ [\ e\ ]_l & prop(r_2) = c_{ld}|c| \end{array} \tag{4}$$

The P system schema representing binding and debinding reactions has been mainly used to model signalling at the cell surface [6,17]. Nevertheless, this schema is not limited to representing receptor activation. It can also be used to specify selective uptake (binding) of certain substances from the environment and delivering of substances to the environment (debinding) by specific transport proteins located on the cell surface [24].

• **Recruitment and Releasing:** Binding of a ligand to its receptor produces a conformational change in the cytosolic domains of the receptor that triggers the recruitment of some cytoplasmic proteins. These proteins are subsequently transformed and released back into the cytoplasm which ultimately induces specific cellular responses.

The rules in (5) model recruitment and releasing in P system specifications. The compartment from where or to where the proteins are recruited or released is specified using square brackets with a label l. In the recruitment rule, r_{rt} the active receptor is represented by the object a placed outside the compartment l where the object b represents the protein that is recruited. These objects are replaced with the object c outside compartment l specifying the formation of the complex formed by the active receptor and the recruited protein.

Conversely, in the releasing rule, r_{rl}, the object c outside compartment l is replaced with the objects d outside and the object e inside the compartment.

$$\begin{array}{ll} r_{rt} : a\ [\ b\ ]_l \xrightarrow{c_{rt}} c\ [\ \ ]_l & prop(r_{rt}) = c_{rt}|a||b| \\ r_{rl} : c\ [\ \ ]_l \xrightarrow{c_{rl}} d\ [\ e\ ]_l & prop(r_{rl}) = c_{rl}|c| \end{array} \tag{5}$$

This P system specification has been used in signal transduction systems [6,17] and to describe processes involving uptake (recruitment) of certain substances from the cytoplasm and the delivering of some substances to the cytoplasm (releasing) by specific transport proteins located on the cell surface [24].

7.3 Specification of Gene Regulation

Living cells can sense very complex environmental and internal signals through some of the molecular interactions described previously. Cells respond to these signals by producing appropriate proteins codified in specific genes. The rate of production of these proteins is regulated by special proteins called *transcription factors* which bind to genes. There are, basically, two different types of transcription factors, *activators* and *repressors*. Although both types bind to genes they have opposite effects. Activators increase the rate of transcription of genes whereas repressors produce a decrease in the rate of transcription of the genes to which they bind. Cells use transcription factors as an internal representation of the environmental and internal state of the cell.

The interaction between transcription factors and genes leading to a change in the rate of production of certain proteins are described by *transcription networks*. In this section, P system specification schemas for transcription networks in prokraryotes are presented. For simplicity only prokaryotes will be considered. In spite of the differences between gene regulation control in prokaryotes and eukaryotes the same fundamental principles and mechanisms still apply in both cases [19].

The *central dogma* of molecular cell biology states that the necessary information for the production of proteins is contained in stretches of DNA called genes. Transcription of a gene is the process by which a protein called *RNA polymerase* produces the mRNA that corresponds to a gene's coding sequence. This mRNA

is then translated into a protein or gene product, by *ribosomes*, complexes made of specific proteins and ribosomal RNA. This picture is much more complex than it first appears since *transcription factors*, which are also proteins encoded in certain genes, acts as regulators in the transcription rate of genes by binding to specific regions or sites of the DNA. These genes can codify in turn other transcription factors or other proteins produced to carry out specific tasks. This provides a feedback pathway by which genes can regulate the expression of other genes and, in this manner, the production of the proteins encoded by them. In this work two different approaches to the specification of transcription networks and gene regulation processes will be discussed.

In the first approach individual objects will be used to specify proteins, transcription factors and genes. Rewriting rules on multisets of objects will describe the interactions between the different components of transcription networks. In the second approach a much more detailed description of the interactions will be developed using objects to represent proteins and transcription factors and strings to represent genes, operons[3] and mRNA. Rewriting rules on multisets of objects and strings will provide a more mechanistic description of the processes that take place in transcription networks.

Specification of Transcription Networks Using Objects. In a simplistic approach processes like transcription and translation can be abstracted as individual reactions. In this case, genes and operons will be specified as individual objects which produce in a single step their complementary mRNA also represented by a single object. The production of a protein from the mRNA is also described in a single step. Finally, the processes involved in gene expression control, like binding and debinding of transcription factors, are also specified using rewriting rules on multisets of objects.

• **Transcription and Translation:** In the P system specification schema in (6) the objects *gene*, *rna* and *prot* specify the stretch of DNA consisting of the gene, its complementary mRNA, and its gene product or protein, respectively. The transcription of the gene into its complementary mRNA is described by the rewriting rule, r_{tc}. According to this rule in the compartments of the type represented by the label l, the object *gene* is replaced with the objects *gene* and *rna*. In this manner, when the rule is applied, the object *gene* remains in the compartment and an object *rna* representing the mRNA is produced.

$$\begin{array}{ll} r_{tc} : [\ gene\ ]_l \xrightarrow{c_{tc}} [\ gene + rna\ ]_l & prop(r_{tc}) = c_{tc}|gene| \\ r_{tl} : [\ rna\ ]_l \xrightarrow{c_{tl}} [\ rna + prot\ ]_l & prop(r_{tl}) = c_{tl}|rna| \end{array} \tag{6}$$

In a similar way translation is described by the single rewriting rule r_{tl}, according to which the object *rna* is replaced with the objects *rna* and *prot*. The application of this rule does not consume the object *rna* but it produces an object *prot* representing the translated protein.

[3] An operon is a group of genes physically linked on the chromosome and under the control of the same promoters. In an operon, the linked genes give rise to a single mRNA molecule that is translated into the different gene products.

• **Binding and Debinding of Transcription Factors to Genes:** The processes of binding and debinding of a transcription factor to a gene can be described by similar rules to the ones used to specify complex formation and dissociation. The P system specification schema in (7) constitutes the specification of these processes through rewriting rules on multisets of objects.

Rule r_{gon} describes the binding of a transcription factor, Tf, to a gene, specified by the object *gene*. According to this rule in compartments of the type l an object Tf and an object *gene* can be replaced with an object Tf–*gene*, which represents the situation when the transcription factor is bound to the gene. The reverse process, the debinding of a transcription factor from a gene, is described through the rule r_{goff}. When this rule is applied in compartments of the type l the object Tf–*gene* is replaced by the objects *gene* and Tf.

$$\begin{array}{ll} r_{gon}: \ [\, Tf + gene \,]_l \xrightarrow{c_{gon}} [\, Tf\text{–}gene \,]_l & prop(r_{gon}) = c_{gon}|Tf||gene| \\ r_{goff}: [\, Tf\text{–}gene \,]_l \xrightarrow{c_{goff}} [\, Tf + gene \,]_l & prop(r_{goff}) = c_{goff}|Tf\text{–}gene| \end{array} \tag{7}$$

Specification of Transcription Networks Using Strings. The use of individual objects to represent the complex structure of genes in the DNA and RNA and the use of single rules to describe the complex processes of transcription and translation is widely used. Nevertheless, transcription networks present some crucial features that questions the applicability of this approach. For instance, in prokaryotes, genes codifying proteins involved in similar tasks are arranged together in a piece of DNA called operon so that they are transcribed in a single strand of mRNA. The order in which these genes are placed in operons is relevant, as it determines the order in which they are transcribed, and thus the order in which their protein products become available. Therefore, it is necessary to specify genes using linear structures like strings if one wants to produce relevant models of transcription networks. Another important fact that is overlooked in approaches describing transcription and translation as individual processes is that in prokaryotes shortly after transcription has started and before it is over ribosomes can bind to the growing mRNA and start translation. Furthermore, there can be many processes of transcription and translation going on at the same time. Summing up, transcription and translation are concurrent and parallel processes that are difficult to specify using individual objects and single step rules.

Finally, another problem that arises from the use of single step rules for the description of transcription networks is the difference in the time scales of their processes. While protein-protein interactions take seconds, transcription and translation may need half an hour to complete. This difference in the time scales produces a difference of many orders of magnitude in the stochastic constants associated with the corresponding rules. When this is the case the applicability of Gillespie's theory of stochastic kinetics is questionable as the difference among the stochastic constants distorts appreciably the evolution of the system. In this section this problem is solved by decomposing the processes of transcription and translation into simpler interactions whose time scales are similar to those of protein-protein interactions.

In what follows we propose the use of strings to represent the linear structure of strands of DNA and RNA and the use of rewriting rules on multisets of objects and strings to describe the binding and debinding of transcription factors to genes and the processes of transcription and translation as concurrent and parallel processes. A typical representation of a gene as a string in presented below where each substring represent a relevant region or site in the gene

$$\langle\ op\,.\,site_{ini}\,.\,site_{mid}\,.\,\cdots\,.\,site_{mid}\,.\,site_{ter}\ \rangle$$

• **Binding and Debinding of Transcription Factors to Specific Sites on the DNA:** Transcription factors bind to specific regions of the genes called operators. These sites are normally located around the region where the RNAP binds to start transcription. The binding of a transcription factor to an operator produces a change in the conformation of that region increasing or decreasing the rate at which RNAP starts the transcription.

The P system specification schema in (8) describes the binding and debinding of a transcription factor represented by the object Tf to an operator specified by the substring $\langle op\rangle$. Specifically, rule r_{tfb} describes the binding of the transcription factor. The effect of this rule consists of the consumption of an object Tf and the rewriting of the substring $\langle op\rangle$ representing the free operator with the substring $\langle op'\rangle$ representing the operator occupied by the transcription factor. The reverse process is described in rule r_{tfd}. An application of this rule produces an object Tf and the replacement of the substring $\langle op'\rangle$ with the substring $\langle op\rangle$.

$$\begin{array}{ll} r_{tfb} : [\ Tf + \langle op\rangle\ ]_l \xrightarrow{c_{tfb}} [\ \langle op'\rangle\ ]_l & prop(r_{tfb}) = c_{on}|Tf||\langle op\rangle| \\ r_{tfd} : [\ \langle op'\rangle\ ]_l \xrightarrow{c_{tfd}} [\ Tf + \langle op\rangle\ ]_l & prop(r_{tfd}) = c_{off}|\langle op'\rangle| \end{array} \tag{8}$$

The constants c_{tfb} and c_{tfd} represent the affinity between the transcription factor and the operator.

• **Transcription:** To carry out transcription, RNAP performs several distinct steps, namely, transcription initiation, mRNA elongation and transcription termination. In what follows a detailed description of the P system schemas used to specify these stages is presented.

– First the RNA polymerase, described by the object RNAP, recognizes and binds reversibly to a specific site at the beginning of the gene, called the *promoter*, represented by the substring $\langle prom\rangle$. The rewriting rules on multisets of objects and strings in (9) describe the binding and debinding of the RNAP to and from the promoter.

The binding of the RNAP to the promoter is described in rule r_{rb}. An application of this rule in a compartment of the type specified by the label l consumes an object RNAP and replaces $\langle prom\rangle$ with $\langle prom.\text{RNAP}\rangle$ in a string which contains $\langle prom\rangle$ as substring. This produces the insertion of the object RNAP after $\langle prom\rangle$ describing the binding of the RNAP to the promoter of the gene.

The debinding of the RNAP from the promoter is specified in rule r_{rd}. According to this rule in a compartment of type l the substring $\langle prom.\text{RNAP}\rangle$ is rewritten with the substring $\langle prom\rangle$ and an object RNAP is produced. An application of this rule produces the removal of the object RNAP from the string

where $\langle prom \rangle$ is located, representing the dropping of the RNAP from the promoter.

$$\begin{array}{c} r_{rb} : [\ \mathrm{RNAP} + \langle prom \rangle\ ]_l \xrightarrow{c_{rb}} [\ \langle prom.\mathrm{RNAP} \rangle\ ]_l \\ prop(r_{rb}) = c_{on}|\mathrm{RNAP}||\langle prom \rangle| \\ r_{rd} : [\ \langle prom.\mathrm{RNAP} \rangle\ ]_l \xrightarrow{c_{rd}} r[\ \mathrm{RNAP} + \langle prom \rangle\ ]_l \\ prop(r_{rd}) = c_{rd}|\langle prom.\mathrm{RNAP} \rangle| \end{array} \tag{9}$$

– Transcription initiation is described by the rewriting rule on strings in (10).This rule specifies the melting of the double strand of the DNA and the transcription of the first nucleotides. These nucleotides are represented by the substring $\langle site_{ini} \rangle$. The complementary ribonucleotides are represented by the substring $\langle \overline{site}_{ini} \rangle$ which mark the beginning of the nascent (growing) mRNA. The effect of an application of the rule r_{ti} in a compartment of type l consists of the replacement of the substring $\langle \mathrm{RNAP}.site_{ini} \rangle$ with the substring $\langle site_{ini}.\overline{site}_{ini}.\mathrm{RNAP} \rangle$ in the string representing the gene.

$$\begin{array}{c} r_{ti} : [\ \langle \mathrm{RNAP}.site_{ini} \rangle\ ]_l \xrightarrow{c_{ti}} [\ \langle site_{ini}.\overline{site}_{ini}.\mathrm{RNAP} \rangle\ ]_l \\ prop(r_{ti}) = c_{ini}|\langle \mathrm{RNAP}.site_{ini} \rangle| \end{array} \tag{10}$$

Note that after an application of rule r_{ti} the substring $\langle prom \rangle$ is free so another object RNAP can bind to it. In this manner we can represent the binding of an RNAP to the promoter of a gene which is currently being transcribed. That is, we can describe different processes of transcription taking place at the same time.

– During the stage of strand elongation, RNAP moves along the template DNA adding nucleotides to the nascent (growing) RNA chain. Although, the growing mRNA hangs from the RNA polymerase and is not part of the DNA; in our specification, the substring representing the growing mRNA is part of the string which represents the DNA. Nevertheless, different symbols will be used to specify DNA sites and RNA sites so the growing mRNA can be easily identified.

The rewriting rule r_{el} in (11) describes the process of mRNA elongation. The substring $\langle \overline{site}_{ini}.w.\mathrm{RNAP}.site_{mid} \rangle$ represents the situation when RNAP with a partially formed chain of mRNA, $\langle \overline{site}_{ini}.w \rangle$, is ready to transcribe the next site in the DNA, $\langle site_{mid} \rangle$.

The addition of newly transcribed nucleotides is achieved by adding the substring $\langle \overline{site}_{mid} \rangle$ to the substring representing the growing mRNA, $\langle \overline{site}_{ini}.w \rangle$. The movement of the RNA polymerase along the DNA leaving behind transcribed sites is described by moving the substring $\langle site_{mid} \rangle$ from immediately ahead of the symbol RNAP to the end of the growing mRNA represented by the substring $\langle \overline{site}_{ini} \rangle$. All this is achieved by rewriting the substring $\langle \overline{site}_{ini}.w.\mathrm{RNAP}.site_{mid} \rangle$ with the substring $\langle site_{mid}.\overline{site}_{ini}.w.\overline{site}_{mid}\mathrm{RNAP} \rangle$.

$$\begin{array}{c} r_{el} : [\ \langle \overline{site}_{ini}.w.\mathrm{RNAP}.site_{mid} \rangle\ ]_l \xrightarrow{c_{el}} [\ \langle site_{mid}.\overline{site}_{ini}.w.\overline{site}_{mid}\mathrm{RNAP} \rangle\ ]_l \\ prop(r_{el}) = c_{el}|\langle \overline{site}_{ini}.w.\mathrm{RNAP}.site_{mid} \rangle| \end{array} \tag{11}$$

– The last stage in RNA synthesis is transcription termination. When the RNAP reaches specific termination sites in the DNA a completed RNA molecule is released and the RNAP dissociates from the gene. Rule r_{ter} in (12) describes this process. The situation when the RNAP with a growing mRNA reaches a termination site is represented by the substring $\langle \overline{site}_{ini}.w.\text{RNAP}.site_{ter}\rangle$. The dissociation of the RNA polymerase from the DNA is described by rewriting the substring $\langle \overline{site}_{ini}.w.\text{RNAP}.site_{ter}\rangle$ with $\langle site_{ter}\rangle$. The release of the RNA polymerase is specified by the production of an object RNAP. Finally, the release of a completed mRNA is represented by the production of a new string $\langle \overline{site}_{ini}.w.\overline{site}_{ter}\rangle$.

$$\begin{aligned} r_{ter} : [\ \langle \overline{site}_{ini}.w.\text{RNAP}.site_{ter}\rangle\ ]_l &\xrightarrow{c_{ter}} [\ \text{RNAP} + \langle site_{ter}\rangle;\ \langle \overline{site}_{ini}.w.\overline{site}_{ter}\rangle]_l \\ prop(r_{ter}) &= c_{ter}|\langle \overline{site}_{ini}.w.\text{RNAP}.site_{ter}\rangle| \end{aligned} \tag{12}$$

• **Translation:** Translation is the whole process by which the nucleotide sequence of an mRNA is used to order and join the amino acids in a polypeptide chain to synthesize a protein. Ribosomes direct the formation of proteins. Similarly to transcription, the complex process of translation can be divided into three stages, initiation, elongation and termination.

– In prokaryotes, shortly after RNAP starts transcription and before it is over, ribosomes bind to specific sites in the growing mRNA called *ribosome binding sites* (RBS) to start translation. Rule r_{tli} describes translation initiation. In this rule the RBS is specified using the substring $\langle \overline{site}_{ini}\rangle$ and ribosomes are represented using the object Rib. An application of this rule in a compartment of type l consumes an object Rib and rewrites the substring $\langle \overline{site}_{ini}\rangle$ with $\langle \overline{site}_{ini}.\text{Rib}\rangle$.

$$\begin{aligned} r_{tli} : [\ \text{Rib} + \langle \overline{site}_{ini}\rangle\ ]_l &\xrightarrow{c_{tli}} [\ \langle \overline{site}_{ini}.\text{Rib}\rangle\ ]_l \\ prop(r_{tli}) &= c_{tli}|\text{Rib}||\langle \overline{site}_{ini}\rangle| \end{aligned} \tag{13}$$

Note that in our approach transcription and translation are specified as concurrent and parallel processes since rules representing translation initiation can be applied before rules describing transcription termination.

– Ribosomes direct elongation of the polypeptide sequence forming a protein by moving along a mRNA chain. In our approach we overlook the growing sequence of amino acids and only specify the movement of ribosomes along the mRNA as we focus on the release of the protein once translation is finished.

The rule in (14) describes a step of elongation. The translocation of a ribosome along the mRNA is achieved by rewriting the substring $\langle \text{Rib}.\overline{site}_{mid}\rangle$ with the substring $\langle \overline{site}_{mid}.\text{Rib}\rangle$.

$$\begin{aligned} r_{tle} : [\ \langle \text{Rib}.\overline{site}_{mid}\rangle\ ]_l &\xrightarrow{c_{tle}} [\ \langle \overline{site}_{mid}.\text{Rib}\rangle\ ]_l \\ prop(r_{tle}) &= c_{tle}|\langle \text{Rib}.\overline{site}_{mid}\rangle| \end{aligned} \tag{14}$$

– In translation termination ribosomes dissociate from a mRNA and release a completed polypeptide chain forming a protein when they reach specific sites marking termination points. This last process is described by the rule r_{tlt} in (15).

The situation when a ribosome reaches a termination site is represented by the substring $\langle \text{Rib}.\overline{site}_{ter} \rangle$. The dissociation of the ribosome from the mRNA and the release of the protein are described by rewriting the substring $\langle \text{Rib}.\overline{site}_{ter} \rangle$ with the substring $\langle \overline{site}_{ter} \rangle$ in the string representing the mRNA and the production of an object Rib and Prot specifying a free ribosome and a newly produced protein, respectively.

$$\begin{aligned} r_{tlt} : [\ \langle \text{Rib}.\overline{site}_{ter} \rangle\ ]_l &\xrightarrow{c_{tlt}} [\ \text{Rib} + \text{Prot} + \langle \overline{site}_{ter} \rangle\ ]_l \\ prop(r_{tlt}) &= c_{tlt} |\langle \text{Rib}.\overline{site}_{ter} \rangle| \end{aligned} \tag{15}$$

8 Analysis of P System Models Using PRISM

Most research in systems biology focuses on the development of models of biological systems accurately enough such as to be able to reveal new properties that can be difficult or impossible to discover through direct lab experiments. One key question is what one can do with a model, other than simple simulation. Is it enough just to realize many simulations of a model to obtain novel knowledge on the system under study? This question has been considered in detail for deterministic models where a rich theory has been produced to analyze systems of differential equations. However, this is not the case for stochastic models, as such systems defy conventional intuition and consequently are harder to conceive. The field is widely open for theoretical advances that help us to reason about systems in greater detail and with finer precision.

There are several attempts in this direction which consists of applying model checking tools to computational models of cellular systems [13]. There are previous studies investigating the use of model checking on P system [1,15]. In this section we will propose the use of a probabilistic symbolic model checking approach based on PRISM (Probabilistic and Symbolic Model Checker) [14].

Model checking is a well established and widely used formal method for verifying the correctness of real life systems. *Probabilistic model checking* is a probabilistic variant of the classical model checking augmented with quantitative information regarding the likelihood that transitions occur and the times at which they do so. One of the major advantages of probabilistic model checking is that it is an exhaustive approach, that is, all possible behaviors of the system are analyzed. Analytical methods based on probabilistic model checking consists of three different steps:

1. First, one must design a precise mathematical model of the system which is to be analyzed. In this work, P system models will be used as the formal description required in this step.
2. Once the formal model is built, one has to translate it into the specific language of the probabilistic model checker, PRISM in this case.
3. Finally, some properties of the model must be identified and expressed formally using temporal logic. This allows the probabilistic model checker to analyze these properties in an automatic way against the constructed model.

The fundamental components of the PRISM language are *modules*, *variables* and *commands*. A model is composed of a number of modules which can interact with each other. A module contains a number of local variables and commands. The values of these variables at any given time constitute the state of the module. The space of reachable states is computed using specified ranges for each variable and their initial values. The global state of the whole model is determined by the local state of all modules.

The behavior of each module is described by a set of commands. A predicate is associated with each command to determine when the command is applicable. The application of a command updates the values of the variables in the module describing a transition of the module. The application of commands is driven by some probabilistic information assign to them using specific expressions.

Once a probabilistic model has been specified and constructed in PRISM, one needs to identify one or more *properties* of the model to be analyzed by the model checker. This is done using temporal logic. One key feature of PRISM is the use of rewards associated with states and transitions. This allows to express reward-based properties which are quantitative in nature. Rewards associated with states, *cumulated rewards*, are incremented in proportion to the time spent in the state, while rewards associated with transitions *impulse rewards* are incremented each time the transition is taken.

Translation of P system Models into PRISM

As mentioned before in order to perform model checking analysis on a P system model it is necessary to translate it into the PRISM language. The three essential components of a P system are a membrane structure consisting of a number of membranes that can interact with each other, multisets of objects[4] and rewriting rules associated with membranes. These components can easily be mapped into the components of the PRISM language using modules to represent membranes, variables to describe objects and commands to specify rules. A detailed description of how to specify P systems models in the PRISM language is presented in what follows.

• Membrane structure: Recall that each membrane is uniquely identified with an identifier i. Therefore, for each membrane i a module with name `compartment_i` will be introduced in the model.

• Alphabet and initial multisets: For each object *obj* that can be present inside the compartment defined by membrane i a local variable `obj_i` will be declared in module `compartment_i`. The initial value of this variable is determined by the initial multiset associated with membrane i. The value range of the variables representing objects will be determined experimentally or it will be derived from the literature. In order to specify these ranges two constants will be declared `upb_obj_i` and `lob_obj_i`.

• Rewriting rules: Commands are used in PRISM to describe the rewriting rules of a P system. Given a rule of the form:

$$r_j^{l_i} : obj_1 \, [\, obj_2 \,]_l \xrightarrow{c_j^{l_i}} obj_1' \, [\, obj_2' \,]_l$$

[4] Strings are not easily represented in PRISM and will not be considered in this work.

with $obj_1 = o_1^1 + \cdots + o_{n_1}^1$,$obj_2 = o_1^2 + \cdots + o_{n_2}^2$,$obj_1' = oo_1^1 + \cdots + oo_{m_1}^1$,$obj_2' = oo_1^2 + \cdots + oo_{m_2}^2$ some finite multisets and $c_j^{l_i}$ the stochastic constant associated with the rule. Assuming that the label of membrane i is l and that it is embedded inside membrane k the objects in the rule are specified as follows. The variables `o_1_1_k`, ..., `o_n_1_1_k`, `oo_1_1_k`, ..., `oo_m_1_1_k` specify the objects from obj_1 and obj_1' in module `compartment_k`. The objects `o_1_2_i`, ..., `o_n_2_2_i` and `oo_1_2_i`, `oo_m_2_2_i` represent the objects from obj_2 and obj_2' in module `compartment_i`. The stochastic constants associated with the rules are specified using PRISM constants.

In general, rules need two membranes to interact in a synchronized way to exchange objects. Therefore when a rule affects two different compartments, the two modules representing them will synchronize the application of two different commands by using a label which identifies the rule `r_j_l_i`.

The command in module `compartment_i` describing the effect of an application of rule $r_j^{l_i}$ in compartment i will be:

```
[ r_j_l_i ] o_1_2_i > 0 & ... & o_n_2_2_i > 0 &
            oo_1_2_i < upb_oo_1_2_i & ... &
            oo_m_2_2_i < upb_oo_m_2_2_i - >
                 c_j_l_i * o_1_2_i * ... * o_n_2_2_i :
                   (o_1_2_i' = o_1_2_i - 1) & ... &
                   (o_n_2_2_i' = o_n_2_2_i - 1) &
                   (oo_1_2_i' = oo_1_2_i + 1) & ... &
                   (oo_m_2_2_i' = oo_m_2_2_i + 1);
```

The command in module `compartment_k` describing the effect of an application of rule $r_j^{l_i}$ in compartment k will be:

```
[ r_j_l_i ] o_1_1_k > 0 & ... & o_n_1_1_k > 0 &
            oo_1_1_k < upb_oo_1_1_k & ... &
            oo_m_1_1_k < upb_oo_m_1_1_k - >
                 o_1_1_k * ... * o_n_1_1_k :
                   (o_1_1_k' = o_1_1_k - 1) & ... &
                   (o_n_1_1_k' = o_n_1_1_k - 1) &
                   (oo_1_1_k' = oo_1_1_k + 1) & ... &
                   (oo_m_1_1_k' = oo_m_1_1_k + 1);
```

Observe that these two commands are applied when the guards hold, that is, if and only if there are some reactants in the corresponding membranes and the products have not reached the upper bounds determined experimentally. Also note that the rate of this transition is the product of the individual rates:

(`c_j_l_i * o_1_2_i * ... * o_n_2_2_i`) (`o_1_1_k * ... * o_n_1_1_k`)

When this transition is performed the local variables representing the reactants are decreased by one and the variables representing the products are increased by one.

Some Specifications of P System Properties in PRISM

The first step when analyzing a model in PRISM is to associate the appropriate rewards with the corresponding states and transitions. A typical analysis consists

of the study of the evolution over time of the number of objects or molecules and the number of applications of rules. Therefore, two different lists of rewards will be used. The first list will associate with each state a reward representing the number of a particular object. A constant `obj` is used to identify which object is being tracked at the moment. In a similar manner a list of rewards will be used to associated with each transition a reward of 1 representing that the rule has been applied once. A constant `rule` is used to identify which rule is being analyzed.

```
rewards "molecules"
obj = 1 : = o1_i;
⋮
obj = n : on_i ;
endrewards
```

```
rewards "rules"
[ r_1_env ] rule = 1 : 1;
⋮
[ r_14_cyto ] rule = 19 : 1;
endrewards
```

Once the corresponding rewards have been associated with particular states and transitions one can use PRISM to model check some properties of the system. The type of properties analyzed in this section are only intended to illustrate how to use PRISM to study the behavior of P system models. We do not intend to cover all possible properties, not even the most common ones, that can be checked in PRISM as the properties to study depend very much on the model being analyzed.

A typical analysis, when dealing with stochastic models, is to compute the expected number of molecules over time. This can be studied in PRISM using instantaneous reward properties where a constant `time` indicates the time instant for which the expected number of molecules is computed, see below left. PRISM also allows to reason about the evolution of P system models as a consequence of the applications of different rules. One can compute the expected number of applications of the different rules within T units of time using cumulative reward properties, see below right.

```
R = ? [ I = time ]
```

```
R = ? [ C <= T ]
```

Another important type of quantitative property which can be computed using PRISM is the expected time for an event to take place. This can be done with reachability reward properties. For instance the property specified below can be used to compute the expected time for the number of objects `o1_i` to get over a threshold `Th`.

```
R = ? [ F o1_i > Th ]
```

PRISM allows us to reason about the probability that a certain type of behavior is observed at specific times during the evolution of our stochastic models. This is done by using the operator `P` and a path property which can use the temporal operators next `X`, until `U` and bounded until `U time`. For instance the property below computes the probability of `o1_i` getting over a threshold `Th` within the first `T` units of time of the evolution of the model.

```
P = ? [ true U <= T o1_i > Th ]
```

Finally, PRISM allows us to reason about the *long run*, *equilibrium* or *steady state* behavior of our models. In this case the operator `S` is used. For example the probability of the number of objects `o1_i` to be between the values `o1_up` and `o1_down` in the long run can be computed using the following expression:

```
S = ? [ o1_i < o1_up & o1_i > o1_down ]
```

9 The Lac Operon System, a Case Study

In this section the lac operon regulation system in *Escherichia coli* (*E. coli*) is used as a case study to illustrate the general principles presented in this paper. Here we present a summary of the model developed in [24]. Gene expression is highly regulated in order to produce the necessary proteinic machinery to respond to environmental changes. At a given time a particular cell only synthesizes those proteins necessary for its survival under the specific conditions of that time. Gene expression is primarily regulated by mechanisms that control transcription initiation.

The lac operon is a group of three genes, lacZ, lacY and lacA physically linked together in an operon. These genes codify β-galactosidase, LacY and LacA, proteins involved in the metabolism and transport of lactose. The lac operon has a dual, positive and negative, regulation system that allows *E. coli* to uptake and consume lactose only in the absence of glucose [19].

9.1 A P System Specification of the Lac Operon

Our P system specification of the lac operon regulation system consists of the following construct:

$$\Pi_{lac} = ((\Sigma_{lac}^{obj}, \Sigma_{lac}^{str}), \{e, s, c\}, [\,[\,[\,\,]_3\,]_2\,]_1, M_1, M_2, M_3, ((R_e^{obj}, \emptyset), (R_s^{obj}, \emptyset), (R_c^{obj}, R_c^{str}))$$

– Specification of the molecular entities: In our P system specification, Π_{lac}, each protein and proteinic complex is represented by an individual object in the alphabet Σ_{lac}^{obj}. As discussed in section 7.3 the specification of operons as strings is more accurate than as individual objects. Following this idea the lac operon is represented by the following string whose components define the relevant sites for the regulation of the operon. These sites are represented by the symbols in Σ_{lac}^{str}.

$$\langle \text{cap.op.} \overbrace{\text{lacZ}_s.\text{lacZ}_m \cdots \text{lacZ}_m.\text{lacZ}_e}^{30} . \overbrace{\text{lacY}_s.\text{lacY}_m \cdots \text{lacY}_m.\text{lacY}_e}^{12} . \overbrace{\text{lacA}_s.\text{lacA}_m \cdots \text{lacA}_m.\text{lacA}_e}^{6} \rangle$$

– Specification of the relevant regions: In the lac operon regulation system the cell surface plays a crucial role since the proteins involved in the selective uptake of glucose and lactose are located in this region of the system. According to section 7.1 two membranes are used to specify an *E. coli* bacterium in our P

systems specification Π_{lac}. Specifically, membrane 2 with label s is introduced to describe the cell surface and membrane 3 embedded in the previous one with label c specifies the cytoplasm. Moreover, in the lac operon system the growing media is also a relevant region as bacteria response differently according to its conditions (presence or absence of glucose/lactose). Therefore, membrane 1 labeled by e is used to describe the growing media or environment. Figure 1 depicts a graphical representation of the membrane structure in Π_{lac}.

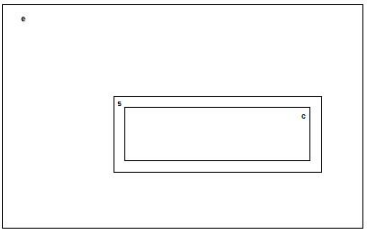

Fig. 1. Membrane structure in the lac operon regulation system

– Specification of the molecular interactions:

The molecular interactions in the regulation system of the lac operon are specified using the rewriting rules in $((R_e^{obj}, \emptyset), (R_s^{obj}, \emptyset), (R_c^{obj}, R_c^{str}))$. Here we only present a few rules to illustrate our approach. For a complete specification of the molecular interactions in the lac operon see [24].

The uptake of glucose, Gluc, and lactose, Lact, from the environment by the proteins LacY and EIICB∼P located on the cell surface, membrane labeled by s, is specified by the binding rules $r_4, r_8 \in R_e^{obj}$. The delivering to the cytoplasm of the sugars is described by the releasing rules $r_6, r_9 \in R_s^{obj}$.

r_4: Gluc [EIICB∼P]$_s \xrightarrow{c_4}$ [EIICB∼P–Gluc]$_s$
r_6: EIICB∼P–Gluc [ ]$_c \xrightarrow{c_6}$ EIICB [Gluc∼P]$_c$
r_8: Lact [LacY]$_s \xrightarrow{c_8}$ [Lact-LacY]$_s$
r_9: Lact-LacY [ ]$_c \xrightarrow{c_9}$ LacY [Lact]$_c$

Glucose uptake needs a phosphate group from EIICB∼P. This protein is phosphorylated by EIIA∼P which is in turn is involved in the production of the activator cAMP according to the recruitment and releasing rules $r_{19} \in R_c^{obj}$ and $r_{20} \in R_s^{obj}$. As a consequence of these rules in the presence of glucose EIIA∼P will be utilized in the glucose transport system and it will not be available to produce the activator.

r_{19}: AC-EIIA∼P [ATP]$_c \xrightarrow{c_{19}}$ AC∼P-EIIA∼P-ATP [ ]$_c$
r_{20}: AC–EIIA∼P–ATP [ ]$_c \xrightarrow{c_{20}}$ AC∼P–EIIA∼P [cAMP]$_c$

When lactose is transported inside the cytoplasm it interacts with β-galactosidase producing as a byproduct allolactose, Allolact. Allolactose binds

to the repressor, LacI, forming a complex, rule $r_{15} \in R_c^{obj}$. This changes the repressor making it incapable of binding to the operator of the operon.

r_{15}: [LacI + Allolact]$_c \overset{c_{15}}{\to}$ [LacI-Allolact]$_c$

The mechanism by which LacI represses the transcription of the lac operon is by reversibly binding to a specific site called operator. This site is represented by the substring $\langle$ op $\rangle$. This process is represented by the rewriting rules on multiset of objects and strings $r_{25}, r_{26} \in R_c^{str}$.

r_{25}: [LacI + $\langle$ op $\rangle$]$_c \overset{c_{25}}{\to}$ [$\langle$ op$^{LacI}\rangle$]$_c$
r_{26}: [$\langle$ op$^{LacI}\rangle$]$_c \overset{c_{26}}{\to}$ [LacI + $\langle$ op $\rangle$]$_c$

The activator CRP-cAMP$_2$ binds to another specific site represented by the substring $\langle$ cap $\rangle$ according to similar transcription factor binding and debinding rules. The RNAP recognizes with different affinities the unoccupied and occupied sites showing a higher transcription initiation rate in the latter case. This is represented in the rules $r_{29}, r_{30} \in R_c^{str}$. Note that there is a 40 fold increase between c_{29} and c_{30}.

r_{29}: [RNAP + $\langle$ cap $\rangle$]$_c \overset{c_{29}}{\to}$ [$\langle$ cap.RNAP $\rangle$]$_c$, $c_{29} = 5 \times 10^{-4} molec^{-1} sec^{-1}$
r_{30}: [RNAP + $\langle$ cap$^{CRP-cAMP_2}$ $\rangle$]$_c \overset{c_{30}}{\to}$ [$\langle$ cap$^{CRP-cAMP_2}$. RNAP $\rangle$]$_c$,
$c_{30} = 0.02 molec^{-1} sec^{-1}$

An example of transcription elongation rule in the lac operon is $r_{36} \in R_c^{str}$. Here the RNAP transcribes a specific site of the lacY gene and attaches the corresponding ribonucleotides $\overline{\text{lacY}}_m$ to the growing mRNA , $\langle$ $\overline{\text{op}}.w$ $\rangle$.

r_{36}: [$\langle$ $\overline{\text{op}}.w$. RNAP. lacY$_m$ $\rangle$]$_c \overset{c_{36}}{\to}$ [$\langle$ lacY$_m$.$\overline{\text{op}}.w.\overline{\text{lacY}}_m$.RNAP $\rangle$]$_c$

The transcription of the lac operon terminates when the RNAP reaches the transcription termination site represented by the string $\langle$ lacA$_e$ $\rangle$. Rule $r_{40} \in R_c^{str}$ specifies the dissociation of the RNAP from the operon and the releasing of a complete mRNA strand, $\langle$ $\overline{\text{op}}.w.\overline{\text{lacA}}_e$ $\rangle$.

r_{40}: [$\langle$ $\overline{\text{op}}.w$.RNAP.lacA$_e$ $\rangle$]$_c \overset{c_{40}}{\to}$ [RNAP + $\langle$ lacA$_e$ $\rangle$; $\langle$ $\overline{\text{op}}.w.\overline{\text{lacA}}_e$ $\rangle$]$_c$

Examples of translation initiation and elongation are rules $r_{41} \in R_c^{str}$ and $r_{45} \in R_c^{str}$ respectively. Note that the rewriting rule on multiset of objects and strings r_{41} describes the recognition by a ribosome of the RBS for lacZ. The rules r_{45} specifies the movement along the mRNA of ribosomes.

r_{41}: [Rib + $\langle$ $\overline{\text{lacZ}}_s$ $\rangle$]$_c \overset{c_{41}}{\to}$ [$\langle$ Rib.$\overline{\text{lacZ}}_s$ $\rangle$]$_c$
r_{45}: [$\langle$ Rib.$\overline{\text{lacZ}}_m$ $\rangle$]$_c \overset{c_{45}}{\to}$ [$\langle$ $\overline{\text{lacZ}}_m$.Rib $\rangle$]$_c$

Finally translation finishes when ribosomes reach termination sites in the mRNA. For instance, rule $r_{46} \in R_c^{str}$, represents translation termination for lacZ when a ribosome reaches the termination site $\overline{\text{lacZ}}_e$ and releases a molecule β-galactosidase, the protein codified by the lacZ gene.

r_{46}: [$\langle$ Rib.$\overline{\text{lacZ}}_e$ $\rangle$]$_c \overset{c_{46}}{\to}$ [$\beta-$Galac + Rib + $\langle$ $\overline{\text{lacZ}}_e$ $\rangle$]$_c$

9.2 P System Models of the Lac Operon

A family of P system models based on our specification Π_{lac} is introduced to study the behavior of the lac operon regulation system under different initial conditions. The parameters of Π_{lac}, $\mathcal{P}(\Pi_{lac}) = (\mathcal{M}_0(\Pi_{lac}), \mathcal{C}(\Pi_{lac}))$ consists of the initial multisets associated with the environment M_1, cell surface M_2 and cytoplasm M_3 and the stochastic constants associated with the rewriting rules $\mathcal{C}(\Pi_{lac})$. Since we are interested in the behavior of the lac operon system under different environmental conditions specific values $\mathcal{C}^0$, M_2^0 and M_3^0 will be given to $\mathcal{C}(\Pi_{lac})$, M_2 and M_3 [24]. In contrast we will vary the values given to M_1 to represent different conditions. More specifically, we will study our system in the presence of lactose and absence of glucose, $M_1^1 = (e, Lact^{3000}, \emptyset)$ and presence of lactose and glucose $M_1^2 = (e, Lact^{3000} + Glucose^{3000}, \emptyset)$. Our study will be performed by running simulation using the algorithm introduced in section 6.2.

– Presence of lactose: This case is represented by the P system model $(\Pi_{lac}; (M_1^1, M_2^0, M_3^0), \mathcal{C}^0)$. Under this condition our simulation showed that the state of the promoter of the lac operon is $\langle\ \mathrm{cap}^{CRP-cAMP_2}.op\ \rangle$. Since lactose is in the media allolactose will appear in the cytoplasm and inutilize the repressor LacI. The absence of glucose in the media will allow AC–EIIA∼P to synthesize the activator cAMP which will interact with the protein CRP and bind to the promoter of the lac operon to increasing transcription initiation by RNAP. This configuration of the promoter yields a full transcription of the operon by many RNAPs, Figure 2 top left, which in turn produces a massive number of the proteins encoded in the operon, for instance LacY, Figure 2 top right.
– Presence of lactose and glucose: This situation is represented by the P system model $(\Pi_{lac}; (M_1^2, M_2^0, M_3^0), \mathcal{C}^0)$. The state of the promoter in this case is $\langle$cap.op$\rangle$ which corresponds with a low transcription of the lac operon in spite of the presence of lactose, this phenomenon is referred to as *catabolite repression*. The presence of lactose in the media excludes the repressors but the presence of glucose in the media represses the synthesis of the activator which produces a non-repressed non-activated operon. Under these conditions only a few RNAP will be active transcribing the operon, Figure 2 bottom left, and only after a delay which corresponds to the time necessary to consume glucose, the proteins codified in the operon are produced, Figure 2 bottom right.

9.3 An Analysis of Gene Expression Using P Systems and PRISM

As it can be seen in the results obtained when modeling the lac operon system gene expression shows a considerable level of noise. In this section in order to illustrate the use of P system models and PRISM to study the stochasticity in cellular systems we will use the abstract gene regulation system in Figure 3. This simple model consists only of rewriting rules on multisets of objects modeling transcription, translation and the interactions between a transcription factor and a gene.

We start by checking the average time the gene is occupied by a transcription factor for different affinities, $\frac{c_6}{c_5}$, and number of transcription factors. This can be

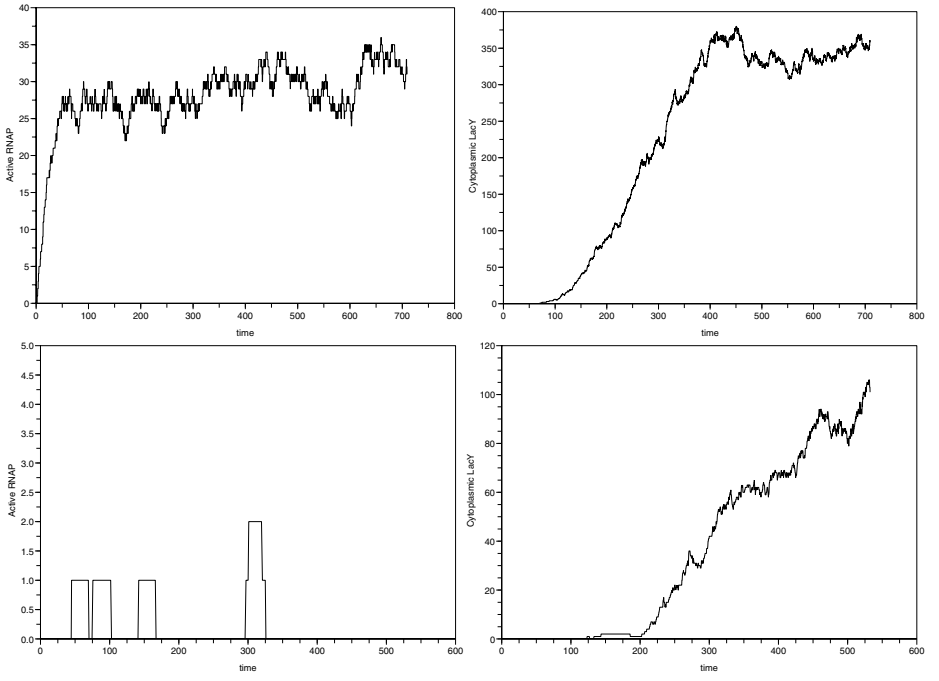

Fig. 2. Number of RNAP transcribing the operon (left) and LacY molecules (right) in the presence of lactose and absence of glucose (top) and in the presence of lactose and glucose (bottom)

P System Rules	PRISM Specification
	`module compartment`
	`gene : [ 0 .. 1 ] init 0;`
	`rna : [ 0 .. uprna ] init 0;`
	`prot : [ 0 .. upprot] init 0;`
$r_1 : [\, gene \,]_c \xrightarrow{c_1} [\, gene + rna \,]_c$	`[ r1 ] gene = 0 & rna < uprna ->`
$r_2 : [\, rna \,]_c \xrightarrow{c_2} [\, rna + prot \,]_c$	`c_1 : (rna' = rna + 1);`
$r_3 : [\, rna \,]_c \xrightarrow{c_3} [\ ]_c$	`[ r2 ] rna > 0 & prot < upprot ->`
$r_4 : [\, prot \,]_c \xrightarrow{c_4} [\ ]_c$	`c_2*rna : (prot' = prot + 1);`
$r_5 : [\, Tf + gene \,]_c \xrightarrow{c_5} [\, Tf\text{–}gene \,]_c$	`[ r3 ] rna > 0 -> c_3*rna : (rna' = rna - 1;)`
$r_6 : [\, Tf\text{–}gene \,]_c \xrightarrow{c_6} [\, Tf + gene \,]_c$	`[ r4 ] prot > 0 -> c_4*prot : (prot' = prot - 1);`
	`[ r5 ] Tf > 0 & gene = 0 -> c_5*Tf : (gene' = 1);`
	`[ r6 ] gene = 1 -> c_6 : (gene' = 0);`
	`endmodule`

Fig. 3. P System rules and PRISM specification for an abstract gene regulation system

done in PRISM by associating a reward of one to every state in which the gene is occupied and using a cumulative reward query. Our results, Figure 4 right, show that for affinities between 10^{12} to 10^9 M fewer than five transcription factors are enough to occupy the gene almost all the time. For an affinity of 10^8 M the

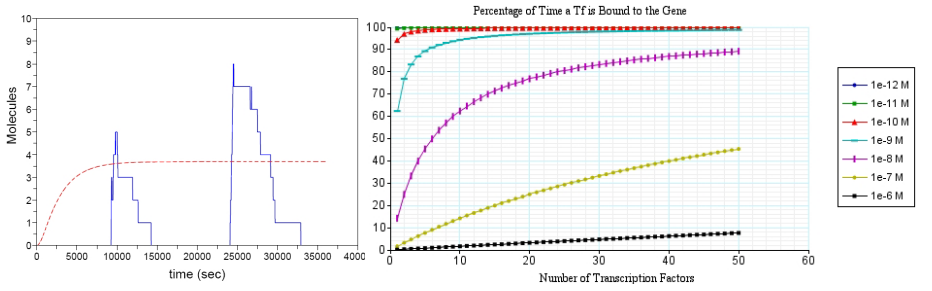

Fig. 4. Number of proteins in the expected evolution and in a single simulation (left) and expected percentage of time a transcription factor is bound to the gene (right)

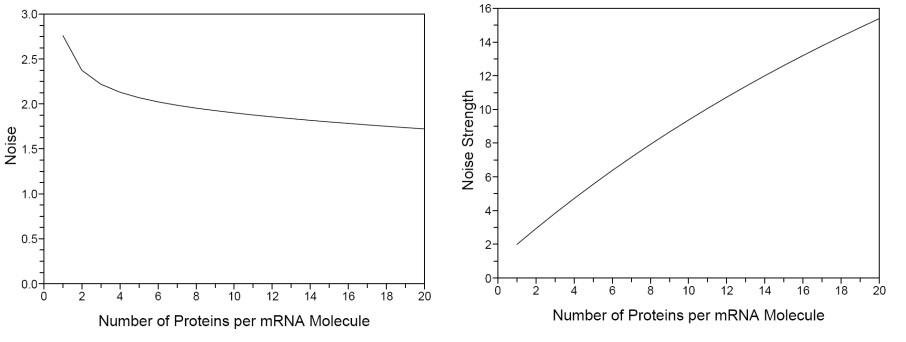

Fig. 5. Noise (left) and noise strength (right) in translational bursting

system is able to discriminate more precisely between only a few transcription factors, fewer than ten, and many, more than thirty. In the first case the gene is occupied less than half of the time whereas for the second case the gene is occupy most of the time. For affinities of the order of $10^7 - 10^6$ M the gene is seldom occupied even when more than fifty transcription factors are present in the system.

We also study the expected number of proteins produced when a very low transcription rate is considered, $c_1 = 10^{-4} sec^{-1}$, Figure 4 left. Note that the expected number of proteins in the long run is three. Nevertheless, when running a single simulation we observe sporadic bursts in the number of protein reaching high peaks. This phenomenon is known as *translation bursting* and is due to a high translation efficiency or expected number of proteins produce by a single molecule of mRNA. Using PRISM we checked that this value is equal to $\frac{c_2}{c_3}$. For different expected number of proteins produced by a molecule of mRNA we study the noise in the system, $\eta = \frac{\sigma}{\mu}$ [5] and the noise strength $\phi = \frac{\sigma^2}{\mu}$. These values were computed using PRISM. Our results show that although high translation efficiencies produce a slight decrease in the level of noise, Figure 5 left,

[5] σ represents the standard deviation and μ the expected value.

they produce a considerable increase in the noise strength, Figure 5 right. This analysis highlights the important of posttranscriptional regulation, a mechanism that is widely overlooked in gene regulation modeling.

10 Conclusions

Modeling in systems biology is subject to very intensive research. Nevertheless, most modeling approaches proposed so far do not present a unifying framework for the specification of the structural components of the cell together with the description of their dynamical interactions. In this work, we have presented P systems as a high level computational modeling framework which integrates the structural and dynamical aspects of cellular systems in a comprehensive and relevant way while providing the required formalization to perform mathematical and computational analysis. The non deterministic and maximally parallel original strategy has been replaced by an adaption of Gillespie algorithm to the multicompartmental structure of P systems in order to develop a stochastic and quantitative framework.

We have discussed a methodology to specify compartments using membranes; molecular entities as objects or strings depending on the relevance of the internal structure and the most important molecular interactions as rewriting rules on multisets of objects and strings. Our modeling framework is not restricted to the simple generation of simulations of our models. We have taken the first steps towards the development of techniques to analyze P system models based on probabilistic and symbolic model checking.

Finally, our work has uncovered several open problems and future lines of research in the use of P systems as a modeling framework in systems biology.

1. The adaption of Gillespie algorithm to a multicompartmental structure has produced a local algorithm easily implemented in an event-driven object-oriented programming style. Such an implementation could be multithreaded on a hyper-threading machine and would also lend itself to full message-passing implementation on a parallel computing cluster. In spite of this no such implementation has been addressed yet.
2. One of the key advantages of P systems with respect to other modeling approaches is the explicit specification of the structural components of cellular systems, more precisely of compartments. In this respect, it is worth noting that up to now P systems have overlooked a key component of the structure of living cells, the cytoskeleton, a dense network of protein filaments that permeate the cytosol and mechanically support membranes and proteins. It is also involved in a great variety of processes like molecular transport, cell division, cell mobility, etc.
3. Regarding membrane interactions in cellular systems we have not investigated important processes where membranes are crucial like cell division, cell adhesion, biofilm formation, etc. The specification and simulation of this type of processes remain an open problem and a future direction to explore.

4. Finally, although very easy to understand P systems present a current limitation to the transparency and utility of the specifications and models designed within their framework. The P system abstractions are purely textual and so far lack of a graphical formal representation for the visualization of the modeled systems.

Summing up, P systems constitute a reliable formal framework for the specification, simulation and analysis of cell systems biology models presenting many challenging future directions.

Acknowledgement

The work of Gh.P. was supported by project BioMAT 2-CEx06-11-97/19.09.06 and project CellSim PC-1284.

General Membrane Computing References

1. G. Ciobanu, Gh. Păun, M.J. Pérez-Jiménez, eds.: *Applications of Membrane Computing*. Springer, 2006 (**AMC 2006** in references below).
2. M. Ionescu, Gh. Păun, T. Yokomori: Spiking neural P systems. *Fundamenta Informaticae*, 71, 2-3 (2006), 279–308.
3. Gh. Păun: Computing with membranes. *Journal of Computer and System Sciences*, 61, 1 (2000), 108–143 (first circulated as Turku Center for Computer Science-TUCS Report 208, November 1998, `www.tucs.fi`).
4. Gh. Păun: *Computing with Membranes: An Introduction*. Springer, 2002.
5. Gh. Păun: Membrane computing. Main ideas, basic results, applications. In *Molecular Computational Models: Unconventional Approaches* (M. Gheorghe, ed.), Idea Group Publ., London, 2004, 1–31.
6. Gh. Păun: Introduction to membrane computing. In *Proc. Brainstorming Workshop on Uncertainty in Membrane Computing*, Palma de Mallorca, Nov. 2004, 17–65.
7. Gh. Păun, M.J. Pérez-Jiménez: Membrane computing. Brief introduction, recent results and applications. *BioSystems*, 85, 1 (2006), 11–22.
8. M.J. Pérez-Jiménez: An approach to computational complexity in membrane computing. In *Membrane Computing, 5th International Workshop, WMC 2004, Milan, Italy, June 2004, Revised, Selected and Invited Papers*, LNCS 3365, Springer, 2005, 85–109.
9. P Systems Web Page: `http://psystems.disco.unimib.it`.

Applications to Biology and Bio-medicine

1. S. Aguzzolli, I.I. Ardelean, D. Besozzi, B. Gerla, C. Manara: P systems under uncertainty: The case of transmembrane proteins. In *Proc. Brainstorming Workshop on Uncertainty in Membrane Computing*, Palma de Mallorca, 2004, 107–118.

2. I.I. Ardelean: Biological roots and applications of P systems. Further suggestions. In *Membrane Computing, WMC2006, Leiden, Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 1–17.
3. I.I. Ardelean, D. Besozzi: On modeling ion fluxes across biological membranes with P systems. In *Proc. Third Brainstorming Week on Membrane Computing*, Sevilla, 2005, RGNC Report 01/2005, 35–42.
4. I.A. Ardelean, D. Besozzi: Some notes on the interplay between P systems and chemotaxis in bacteria. In *Proc. Fourth Brainstorming Week on Membrane Computing*, Sevilla, 2006, RGNC Report 02/2006, 41–48.
5. I.I. Ardelean, D. Besozzi, M.H. Garzon, G. Mauri, S. Roy: P system models for mechanosensitive channels. In **AMC 2006**, 43–82.
6. I. Ardelean, M. Cavaliere: Playing with a probabilistic P system simulator: Mathematical and biological problems. *New Generation Computing*, 22, 4 (2004), 311–329.
7. I.I. Ardelean, M. Cavaliere: Modelling biological processes by using a probabilistic P system software. *Natural Computing*, 2, 2 (2003), 173–197.
8. J. Auld, L. Bianco, G. Ciobanu, M. Gheorghe, D. Pescini, F.J. Romero-Campero: Population P systems – A model for the behaviour of systems of bio-entities. In *Pre-proc. WMC7*, Leiden, The Netherlands, 2006, 15–20.
9. R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, A. Troina: The calculus of looping sequences for modeling biological membranes. In *Membrane Computing. Eight Workshop on Membrane Computing, WMC2007, Thessaloniki, Greece, June 2007. Revised, Selected and Invited Papers*, LNCS 4860, Springer, 2007.
10. F. Bernardini: *Membrane Systems for Molecular Computing and Biological Modelling*. PhD Thesis, Univ. Sheffield, Anglia, 2006.
11. F. Bernardini, M. Gheorghe, N. Krasnogor: Quorum sensing P systems. *Theoretical Computer Sci.*, 371, 1-2 (2007), 20–33.
12. F. Bernardini, M. Gheorghe, N. Krasnogor, R.C. Muniyandi, M.J. Pérez-Jiménez, F.J. Romero-Campero: On P systems as a modelling tool for biological systems. In *Membrane Computing, International Workshop, WMC6, Vienna, Austria, 2005, Selected and Invited Papers*, LNCS 3850, Springer, 2006, 114–133.
13. F. Bernardini, M. Gheorghe, N. Krasnogor, M.J. Pérez-Jiménez, F.J. Romero-Campero: Modelling vibrio fischeri's behaviour using P systems. Submitted 2005.
14. F. Bernardini, V. Manca: Dynamical aspects of P systems. *BioSystems*, 70, 2 (2003), 85–93.
15. F. Bernardini, F.J. Romero-Campero, M. Gheorghe, M.J. Pérez-Jiménez: A modeling approach based on P systems with bounded parallelism. In *Membrane Computing, 7th International Workshop, WMC 2006, Leiden, The Netherlands, July 2006, Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 49–65.
16. D. Besozzi: *Computational and Modelling Power of P Systems*. PhD Thesis, Univ. degli Studi di Milano, Italy, 2004.

17. D. Besozzi, I.I. Ardelean, G. Mauri: The potential of P systems for modeling the activity of mechanosensitive channels in E. Coli. In *Pre-proc. WMC 2003*, Tarragona, GRLMC Report 28/03, 84–102.
18. D. Besozzi, P. Cazzaniga, D. Pescini, G. Mauri: Seasonal variance in P systems models for metapopulations. *Progress in Natural Science*, 17, 4 (2007), 392–400.
19. D. Besozzi, G. Ciobanu: A P system description of the sodium-potassium pump. In *Membrane Computing, 5th International Workshop, WMC 2004, Milan, Italy, June 2004, Revised, Selected and Invited Papers*, LNCS 3365, Springer, 2005, 210–223.
20. L. Bianco: Psim – A computational platform for metabolic P systems. In *Membrane Computing. Eight Workshop on Membrane Computing, WMC2007, Thessaloniki, Greece, June 2007. Revised, Selected and Invited Papers*, LNCS 4860, Springer, 2007.
21. L. Bianco, F. Fontana, G. Franco, V. Manca: P systems for biological dynamics. In **AMC 2006**, 83–128.
22. L. Bianco, F. Fontana, V. Manca: P systems with reaction maps. *Intern. J. Found. Computer Sci.*, 17, 1 (2006), 27–48.
23. L. Bianco, F. Fontana, V. Manca: Computation of biochemical dynamics using MP systems. *Computational Methods in Systems Biology*, Intern. Conference, Trento, 2006, poster.
24. L. Bianco, V. Manca: Metabolic algorithms and signal transduction dynamical networks. In *Proc. Brainstorming Workshop on Uncertainty in Membrane Computing*, Palma de Mallorca, 2004, 119–120.
25. L. Bianco, V. Manca: Symbolic generation and representation of complex oscillations. *Intern. J. Computer Math.*, 83, 7 (2006), 549–568.
26. L. Bianco, D. Pescini, P. Siepmann, N, Krasnogor, F.J. Romero-Campero, M. Gheorghe: Towards a P systems Pseudomonas quorum sensing model. In *Membrane Computing, 7th International Workshop, WMC 2006, Leiden, The Netherlands, July 2006, Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 197–214.
27. M. Cavaliere: Modelling biological processes in P systems. Handling imprecision and constructing new models. In *Proc. Brainstorming Workshop on Uncertainty in Membrane Computing*, Palma de Mallorca, 2004, 143–144.
28. M. Cavaliere, I.I. Ardelean: Modelling respiration in bacteria and respiration/photosynthesis interaction in Cyanobacteria by using a P system simulator. In **AMC 2006**, 129–158.
29. M. Cavaliere, S. Sedwards: Membrane systems with peripheral proteins: transport and evolution. *Technical Report* 04/2006, Centre for Computational and Systems Biology, Trento, 2006.
30. M. Cavaliere, S. Sedwards: Modelling cellular processes using membrane systems with peripheral and integral proteins. *Computational Methods in Systems Biology*, Intern. Conference, Trento, 2006, LNBI 4210, Springer, 2006, 108–126.
31. M. Cavaliere, S. Sedwards: Decision problems in membrane systems with peripheral proteins, transport and evolution. *Technical Report* 12/2006, Centre for Computational and Systems Biology, Trento, 2006.

32. P. Cazzaniga, D. Pescini, D. Besozzi, E. Martegani, G. Mauri: Stochastic modelling and simulations of the Ras/cAMP/PKA pathway in the yeast Saccharomyces cerevisiae. *Proc. CMBS 2007.*
33. P. Cazzaniga, D. Pescini, D. Besozzi, G. Mauri: Tau leaping stochastic simulation method in P systems. In *Membrane Computing, 7th International Workshop, WMC 2006, Leiden, The Netherlands, July 2006, Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 298–313.
34. P. Cazzaniga, D. Pescini, F.J. Romero-Campero, D. Besozzi, G. Mauri: Stochastic approaches in P systems for simulating biological systems. In *Proc. Brainstorming Week on Membrane Computing 2006*, Fenix Editora, Sevilla, 2006, 145–164.
35. S. Cheruku, A. Păun, F.J. Romero-Campero, M.J. Pérez-Jiménez, O.H. Ibarra: Simulating FAS-induced apoptosis by using P systems. *Progress in Natural Science*, 17, 4 (2007), 424–431.
36. G. Ciobanu: Modeling cell-mediated immunity by means of P systems. In **AMC 2006**, 159–180.
37. D.V. Corne, P. Frisco: Dynamics of HIV infection studied with cellular automata and conformon-P systems. *BioSystems*, 2007.
38. S. Dunn, P. Stivers: P system models of bistable, enzyme driven chemical reaction networks. In *Bio-Inspired Modeling of Cognitive Tasks, Proc. IWINAC 2007*, Mar Menor, 2007, LNCS 4527, 203–213.
39. F. Fontana, L. Bianco, V. Manca: P systems and the modeling of biochemical oscillations. In *Membrane Computing, International Workshop, WMC6, Vienna, Austria, 2005, Selected and Invited Papers*, LNCS 3850, Springer, 2006, 199–208.
40. G. Franco, N. Jonoska, B. Osborn, A. Plaas: Knee joint injury and repair modeled by membrane systems. *BioSystems*, 2007.
41. G. Franco, V. Manca: A membrane system for the leukocyte selective recruitment. In *Membrane Computing. Intern. Workshop, WMC2003, Tarragona*, LNCS 2933, Springer, 2004, 181–190.
42. R. Freund, T. Gschwandtner: P systems for modelling biological processes in living cells. Submitted 2006.
43. P. Frisco, D.W. Corne: Advances in modeling the dynamics of HIV infection with conformon-P systems. In *Membrane Computing. Eight Workshop on Membrane Computing, WMC2007, Thessaloniki, Greece, June 2007. Revised, Selected and Invited Papers*, LNCS 4860, Springer, 2007.
44. M. Gheorghe: P systems – a new computational approach in systems biology. In *Pre-proceedings of BIC-TA 2006. Volume of Membrane Computing Section*, 7–14.
45. M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, A. Riscos-Núñez, F.J. Romero-Campero: A membrane computing view on tumours. *Progress in Natural Science*, 17, 4 (2007), 449–457.
46. M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, F.J. Romero-Campero: Simulating avascular tumors with membrane systems. In *Proc. Third Brainstorming Week on Membrane Computing*, Sevilla, 2005, RGNC Report 01/2005, 185–196.

47. T. Hinze, S. Hayat, T. Lenser, N. Matsumaru, P. Dittrich: Hill kinetics meets P systems. A case study on gene regulatory networks as computing agents in silico and in vivo. In *Membrane Computing. Eight Workshop on Membrane Computing, WMC2007, Thessaloniki, Greece, June 2007. Revised, Selected and Invited Papers*, LNCS 4860, Springer, 2007.
48. T. Hinze, T. Lenser, P. Dittrich: A protein substructure based P system for description and analysis of cell signalling network. In *Membrane Computing, WMC2006, Leiden, Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 409–423.
49. D. Jackson, M. Gheorghe, M. Holcombe, F. Bernardini: An agent-based behavioural model of Monomorium pharaonis colonies. In *Membrane Computing. Intern. Workshop, WMC2003, Tarragona*, LNCS 2933, Springer, 2004, 232–239.
50. N. Krasnogor, M. Gheorghe, G. Terrazas, S. Diggle, P. Williams, M. Camara: An appealing computational mechanism drawn from bacterial quorum sensing. *Bulletin of the EATCS*, 85 (2005), 135–148.
51. V. Manca: Topics and problems in metabolic P systems. In *Proc. Brainstorming Week on Membrane Computing 2006*, Fenix Editora, Sevilla, 2006, Vol. 2, 173–184.
52. V. Manca: MP systems approaches to biochemical dynamics: biological rhythms and oscillations. In *Membrane Computing, WMC2006, Leiden, Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 86–99.
53. V. Manca: Metabolic P systems for biochemical dynamics. *Progress in Natural Science*, 17, 4 (2007), 384–391.
54. V. Manca, L. Bianco, F. Fontana: Evolution and oscillation in P systems. Applications to biological phenomena. In *Membrane Computing. International Workshop WMC5, Milano, Italy, 2004*, LNCS 3365, Springer, 2005, 63–84.
55. S. Marcus: Bridging P systems and genomics: A preliminary approach. In *Membrane Computing, International Workshop, WMC-CdeA 2002. Curtea de Argeş, Romania, August 2002, Revised Papers*, LNCS 2597, Springer, 2003, 371–376.
56. E. Martegani, R. Tisi, F. Belotti, S. Colombo, C. Paiardi, J. Winderickx, P. Cazzaniga, D. Besozzi, G. Mauri: Identification of an intracellular signalling complex for ras/camp pathway in yeast: experimental evidences and modelling. *ISSY 25 Conf.*, Hanassari, Espo, Finland, 2006.
57. I. Nepomuceno, J.A. Nepomuceno, F.J. Romero-Campero: A tool for working with the SBML format in P systems modelling biological processes. In *Proc. Third Brainstorming Week on Membrane Computing*, Sevilla, 2005, RGNC Report 01/2005, 219–228.
58. T.Y. Nishida: Simulation of photosynthesis by a K-subset transforming system with membranes. *Fundamenta Informaticae*, 49, 1-3 (2002), 249–259.
59. T.Y. Nishida: A membrane computing model of photosynthesis. In **AMC 2006**, 181–202.
60. A. Păun, M.J. Pérez-Jiménez, F.J. Romero-Campero: Modeling signal transduction using P systems. In *Membrane Computing, WMC2006, Leiden, Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 100–122.

61. M.J. Pérez-Jiménez: P systems-based modelling of cellular signalling pathways. In *Pre-proc. WMC7, 2006*, Leiden, The Netherlands, 54–73.
62. M.J. Pérez-Jiménez, F.J. Romero-Campero: Modelling EGFR signalling network using continuous membrane systems. In *Third Workshop on Computational Methods in Systems Biology*, Edinburgh, 2005.
63. M.J. Pérez-Jiménez, F.J. Romero-Campero: A study of the robustness of the EGFR signalling cascade using continuous membrane systems. In *Proc. IWINAC 2005*, La Palma de Gran Canaria, LNCS 3561, Springer, 2005, vol. I, 268–278.
64. D. Pescini, D. Besozzi, G. Mauri: Investigating local evolutions in dynamical probabilistic P systems. In *Proc. SYNASC 05*, IEEE Press, 2005, 440–447.
65. D. Pescini, D. Besozzi, C. Zandron, G. Mauri: Analysis and simulation of dynamics in probabilistic P systems. In *Proc. DNA11*, UWO, London, Ontario, 2005, LNCS 3892, Springer, 2006, 236–247.
66. A. Profir, N. Barbacari, C. Zelinschi: Gene regulatory network modelling by means of membrane systems. In *Pre-proc. WMC6*, Vienna, July 2005, 162–178.
67. A. Profir, E. Boian, N. Barbacari, E. Guţuleac, C. Zelinschi: Modelling molecular genetic triggers by means of P transducers. In *Pre-proc. WMC2004*, Milano, Italy, 360–362.
68. Y. Pu, Y. Yu, X. Dong: Simulation of biomolecular processes by using stochastic P systems. In *Proc. Workshop on High Performance Computing in the Life Sciences*, Ouro Preto, Brasil, October 2006.
69. F.J. Romero-Campero: *P Systems, a Computational Modelling Framework for Systems Biology*. PhD Thesis, Univ. Sevilla, 2008.
70. A. Romero-Jiménez, M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez: Graphical modelling of higher plants using P systems. In *Membrane Computing, WMC2006, Leiden, The Netherlands, Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 496–507.
71. F.J. Romero-Campero, M. Gheorghe, G. Ciobanu, J. Auld, M.J. Pérez-Jiménez: Cellular modelling using P systems and process algebra. *Progress in Natural Science*, 17, 4 (2007), 375–383.
72. I. Stamatopoulou, M. Gheorghe, P. Kefalas: Modelling of dynamic configuration of biology-inspired multi-agent systems with communicating X-machines and population P systems. In *Membrane Computing. Eight Workshop on Membrane Computing, WMC2007, Thessaloniki, Greece, June 2007. Revised, Selected and Invited Papers*, LNCS 4860, Springer, 2007.
73. Y. Suzuki, S. Ogishima, H. Tanaka: Modeling the p53 signaling network by using P systems. In *Pre-proceedings of Workshop on Membrane Computing, WMC2003*, Tarragona, GRLMC Report 28/03, 449–454.
74. Y. Suzuki, H. Tanaka: Abstract rewriting systems on multisets, ARMS, and their application for modelling complex behaviours. In *Brainstorming Week on Membrane Computing*, Tarragona, February 2003, TR 26/03, URV, 2003, 313–331.
75. Y. Suzuki, H. Tanaka: Modeling p53 signalins pathways by using multiset processing. In **AMC 2006**, 203–214.

76. G. Terrazas, N. Krasnogor, M. Gheorghe, F. Bernardini, S. Diggle, M. Camara: An environment aware P system model of quorum sensing. In *Proc. New Computational Paradigms. First Conf. on Computability in Europe, CiE2005, Amsterdam*, LNCS 3536, Springer, 2005, 479–485.
77. M. Umeki, Y. Suzuki: Direct simulation of the Oregonator model by using a class of P systems. In *Membrane Computing. Eight Workshop on Membrane Computing, WMC2007, Thessaloniki, Greece, June 2007. Revised, Selected and Invited Papers*, LNCS 4860, Springer, 2007.

Applications to Linguistics

1. G. Bel Enguix: Preliminaries about some possible applications of P systems to linguistics. In *Membrane Computing, WMC-CdeA2002, Curtea de Argeş, Romania, August 2002, Revised Papers*, LNCS 2597, Springer, 2003, 74–89.
2. G. Bel Enguix: Analyzing P systems structure. Working, predictions and some linguistic suggestions. In *Proc. Workshop on Membrane Computing, Milan, Italy*, 2004, 138–150.
3. G. Bel Enguix: Unstable P systems. Applications to linguistics. In *Membrane Computing, 5th International Workshop, WMC 2004, Milan, Italy, June 2004, Revised, Selected and Invited Papers*, LNCS 3365, Springer, 2005, 190–209.
4. G. Bel-Enguix, R. Gramatovici: Active P automata and natural language processing. In *Membrane Computing. Intern. Workshop, WMC2003, Tarragona*, LNCS 2933, Springer, 2004, 31–42.
5. G. Bel Enguix, M.D. Jiménez-López: Biosyntax. An overview. *Fundamenta Informaticae*, 64, 1-4 (2005), 17–28.
6. G. Bel Enguix, M.D. Jiménez-López: Modelling parallel conversations with P systems. *Proc. SYNASC 05*, IEEE Press, 2005, 395–398.
7. G. Bel Enguix, M.D. Jiménez-López: Linguistic membrane systems and applications. In **AMC 2006**, 347–388.
8. G. Bel Enguix, M.D. Jiménez-López: Dynamic meaning membrane systems: An application to the description of semantic change. *Fundamenta Informaticae*, 76, 3 (2007), 219-237.
9. R. Gramatovici, G. Bel Enguix: Parsing with P automata. In **AMC 2006**, 389–410.

Applications to Economics

1. J. Bartosik: Paun's systems in modeling of human resource management. *Proc. Second Conf. Tools and Methods of Data Transformation*, WSU Kielce, 2004.
2. J. Bartosik: Heaps of pieces and Paun's systems. *Proc. Second Conf. Tools and Methods of Data Transformation*, WSU Kielce, 2004.

3. J. Bartosik: Membrany dynamicsne w modelowaniu systemow ekonomicznych. *Conf. Bad. Oper. i Syst.*, 2006.
4. J. Bartosik, W. Korczynski: Systemy membranowe jako modele hierarchicznych struktur zarzadzania. *Mat. Pokonferencyjne Ekonomia, Informatyka, Zarzadzanie. Teoria i Praktyka*, Wydzial Zarzadzania AGH, Tom II, AGH 2002.
5. F. Bernardini, M. Gheorghe, M. Margenstern, S. Verlan: Producer/consumer in membrane systems and Petri nets. *CiE 2007.*
6. R. Freund, M. Oswald, T. Schirk: How a membrane agent buys goods in a membrane store. *Progress in Natural Science*, 17, 4 (2007), 442–448.
7. W. Korczynski: Transformacje systemow Pauna jako model przeksztalcen systemowych. *Raport z Badan Grantu W2/2003*, WSU Kielce, 2004.
8. W. Korczynski: On a model of economic systems. *Second Conf. Tools and Methods of Data Transformation*, WSU Kielce, 2004.
9. W. Korczynski: Păun's systems and accounting. *Pre-proc. Sixth Workshop on Membrane Computing*, Vienna, Austria, July 2005, 461–464.
10. W. Korczynski, G. Wawrzola, S. Wawrzola: On a reconstruction problem for membrane systems. *Second Conf. Tools and Methods of Data Transformation*, WSU Kielce, 2004.
11. M. Oswald: Independent agents in a globalized world modelled by tissue P systems. In *Workshop on Artificial Life and Robotics*, 2006.
12. Gh. Păun, R. Păun: Membrane computing as a framework for modeling economic processes. In *Proc. SYNASC 05*, Timişoara, Romania, IEEE Press, 2005, 11–18.
13. Gh. Păun, R. Păun: A membrane computing approach to economic modeling: The producer-retailer interactions and investments. *Analiză şi Prospectivă Economică* (*Economic Analysis and Forecasting*), Part I: 3, 1 (2006), 30–37, Part II: 4, 2 (2006), 47–54.
14. Gh. Păun, R. Păun: Membrane computing models for economics. An invitation-survey. *Studii şi Cercetări de Calcul Economic şi Cibernetică Economică* (*Economic Studies and Research*), 2007.

Applications to Computer Science

Sorting and Ranking

1. A. Alhazov, D. Sburlan: Static sorting algorithms for P systems. In *Preproceedings of Workshop on Membrane Computing, WMC2003*, Tarragona, GRLMC Report 28/03, 17–40.
2. A. Alhazov, D. Sburlan: Static sorting P systems. In **AMC 2006**, 215–252.
3. J.J. Arulanandhan: Implementing bead-sort with P systems. In *Proc. Unconventional Models of Computation Conf. 2002*, LNCS 2509, Springer, 2002, 115–125.
4. M. Ionescu, D. Sburlan: Some applications of spiking neural P systems. In *Pre-proc. Eight Workshop on Membrane Computing, WMC2007, Thessaloniki, Greece, June 2007*, 383–394.

Simulating Circuits and Parallel Architectures

1. A. Binder, R. Freund, G. Lojka, M. Oswald: Applications of membrane systems in distributed systems. *Progress in Natural Science*, 17, 4 (2007), 401–410.
2. M. Cavaliere, V. Deufemia: Specifying dynamic software architectures by using membrane systems. In *Proc. Third Brainstorming Week on Membrane Computing*, Sevilla, 2005, RGNC Report 01/2005, 87–106.
3. R. Ceterchi, M. Pérez-Jiménez: Simulating shuffle-exchange networks with P systems. In *Proc. Second Brainstorming Week on Membrane Computing*, Sevilla, 2004, RGNC Report 01/2004, 117–129.
4. R. Ceterchi, M. Pérez-Jiménez: A perfect shuffle algorithm for reduction processes and its simulation with P systems. In *Proc ICCC, Oradea*, Ed. Univ. Oradea, 2004, 92–98.
5. R. Ceterchi, M.J. Pérez-Jiménez: Simulating parallel architectures with P systems. In *Proc. Fourth Workshop on Membrane Computing, Milan, Italy*, 2004, 184–185.
6. R. Ceterchi, M.J. Pérez-Jiménez, A.I. Tomescu: Simulating the bitonic sort on a 2D-mesh with P systems. In *Membrane Computing. Eight Workshop on Membrane Computing, WMC2007, Thessaloniki, Greece, June 2007. Revised, Selected and Invited Papers*, LNCS 4860, Springer, 2007.
7. R. Ceterchi, D. Sburlan: Simulating Boolean circuits with P systems. In *Membrane Computing. Intern. Workshop, WMC2003, Tarragona*, LNCS 2933, Springer, 2004, 104–122.
8. A. Leporati, C. Zandron, G. Mauri: Simulating the Fredkin gate with energy-based P systems. *J. Univ. Computer Science*, 10, 5 (2004), 600–619.
9. A. Leporati, C. Zandron, G. Mauri: Reversible P systems to simulate Fredkin circuits. *Fundamenta Informaticae*, 74, 4 (2006), 529–548.
10. D. Sburlan: From cells to software architecture. A P system outlook of computational design. *Third Workshop on Mathematical Modelling of Environmental and Life Sciences Problems*, Constanţa, 2004.
11. I. Stamatopoulou, P. Kefalas, M. Gheorghe: OPERAS$_{CC}$ – An instance of a formal framework for MAS modelling based on population P systems. In *Membrane Computing. Eight Workshop on Membrane Computing, WMC2007, Thessaloniki, Greece, June 2007. Revised, Selected and Invited Papers*, LNCS 4860, Springer, 2007.
12. K. Ueda, N. Kato: LNMtal – a language model with links and membranes. In *Membrane Computing. 5th Intern Workshop, WMC2004. Milan, Italy, June 2004. Revised Selected and Invited Papers*, LNCS 3365, Springer, 2005, 110–125.

Computer Graphics, Picture Languages

1. R. Ceterchi, R. Gramatovici, N. Jonoska: P systems for tiling rectangular pictures. In *Membrane Computing. Intern. Workshop, WMC2003, Tarragona*, LNCS 2933, Springer, 2004, 88–103.

2. R. Ceterchi, R. Gramatovici, N. Jonoska, K.G. Subramanian: Tissue-like P systems with active membranes for picture generation. *Fundamenta Informaticae*, 56, 4 (2003), 311–328.
3. K.S. Dersanambika, K. Krithivasan: Contextual array P systems. *Intern. J. Computer Math.*, 81, 8 (2004), 955–969.
4. K.S. Dersanambika, K. Krithivasan, K.G. Subramanian: P systems generating hexagonal picture languages. In *Pre-proceedings of Workshop on Membrane Computing, WMC2003*, Tarragona, GRLMC Report 28/03, 209–221.
5. R. Freund, M. Oswald, A. Păun: P systems generating trees. In *Membrane Computing. 5th Intern. Workshop, WMC2004, Milan, Italy, June 2004. Revised Selected and Invited Papers*, LNCS 3365, Springer, 2005, 309–219.
6. A. Georgiou, M. Gheorghe: P systems used in plant graphics. In *Preproceedings of Workshop on Membrane Computing, WMC2003*, Tarragona, GRLMC Report 28/03, 266–272.
7. A. Georgiou, M. Gheorghe, F. Bernardini: Membrane-based devices used in computer graphics. In **AMC 2006**, 253–282.
8. S. N. Krishna, K. Krithivasan, R. Rama: P systems with picture objects. *Acta Cybernetica*, 15, 1 (2001), 53–74.
9. R. Rama, H. Ramesh: On generating trees by P systems. *Proc. SYNASC 05, Timişoara, Romania*, IEEE Press, 2005, 462–466.
10. A. Romero-Jiménez, M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez: The growth of branching structures with P systems. In *Proc. Fourth Brainstorming Week on Membrane Computing*, Sevilla, 2006, RGNC Report 02/2006, Vol. II, 253–266.
11. K.G. Subramanian, S. Hemalatha, C. Sri Hari Nagore: On image generation by sequential/parallel rewriting P systems. In *Proc. Intern. Conf. on Signal Processing, Communications and Networking*, Anna University, Chennai, IEEE & IETE, 2007, 70–73.
12. K.G. Subramanian, R. Saravanan, K. Rangarajan: Array P systems and basic puzzle grammars. *National Conf. in Intelligent Optimization Modeling*, Gandhigram Rural Institute-Deemed University, India, March 2006.

Cryptography

1. A. Atanasiu: Authentication of messages using P systems. In *Membrane Computing, International Workshop, WMC-CdeA 2002. Curtea de Argeş, Romania, August 2002, Revised Papers*, LNCS 2597, Springer, 2003, 33–42.
2. S.N. Krishna, R. Rama: Breaking DES using P systems. *Theoretical Computer Sci.*, 299, 1-3 (2003), 495–508.
3. O. Michel, F. Jacquemard: An analysis of a public key protocol with membranes. In **AMC 2006**, 283–302.
4. A. Obtulowicz: On P systems with active membranes solving integer factorization problem in a polynomial time. In *Multiset Processing*, LNCS 2236, Springer, 2001, 257–286.
5. A. Obtulowicz: Membrane computing and one-way functions. *Intern. J. Found. Computer Sci.*, 12, 4 (2001), 551–558.

Optimization

1. L. Huang: *Research on Membrane Computing. Optimization Methods*. PhD Thesis, Institute of Advanced Process Control, Zhejiang University, China, 2007.
2. L. Huang, X.-X. He, N. Wang, Y. Xie: P systems based multi-objective optimization algorithm. *Progress in Natural Science*, 17, 4 (2007), 458–464.
3. L. Huang, I.H. Suh: Design of controllers for marine Diesel engine by membrane computing. *Intern. J. Innovative Computing, Information and Control*, 2007.
4. L. Huang, L. Sun, N. Wang, X.M. Jin: Multiobjective optimization of simulated moving bed by tissue P system. *Chinese J. Chemical Engineering*, 15, 5 (2007), 683–690.
5. L. Huang, N. Wang: An optimization algorithms inspired by membrane computing. In *Proc. ICNC 2006*, LNCS 4222, Springer, 2006, 49–55.
6. L. Huang, N. Wang: Multiobjective optimization for controllers. *Acta Automatica Sinica*, 2007.
7. L. Huang, N. Wang: A variant of P systems for optimization. *Neurocomputing*, 2007.
8. L. Huang, N. Wang: An extension of membrane computing – a type of evolutionary computing. *J. Central South Univ., China*, 2007.
9. T.Y. Nishida: An application of P systems – A new algorithm for NP-complete optimization problems. In *Proceedings of the 8th World Multi-Conference on Systems, Cybernetics and Informatics*, vol. V, 2004, 109–112.
10. T.Y. Nishida: Membrane algorithm – an approximate algorithm for NP-complete optimization problems exploiting P systems. In *Membrane Computing, International Workshop, WMC6, Vienna, Austria, 2005, Selected and Invited Papers*, LNCS 3850, Springer, 2006, 55–66.
11. T.Y. Nishida: Membrane algorithms. Approximate algorithms for NP-complete optimization problems. In **AMC 2006**, 303–314.
12. T.Y. Nishida: Membrane algorithm with Brownian subalgorithm and genetic subalgorithm. *Intern. J. Found. Computer Sci.*, 18, 6 (2007), 1353–1360.
13. D. Zaharie, G. Ciobanu: Distributed evolutionary algorithms inspired by membranes in solving continuous optimization problems. In *Membrane Computing, WMC2006, Leiden, The Netherlands. Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 536–554.

Others

1. B. Aman, G. Ciobanu: Translating mobile ambients into P systems. In *Proc. MeCBIC 2006*, Veneţia, 2006.
2. B. Aman, G. Ciobanu: On the relationship between P systems and mobile ambients. *FML Technical Report*, `http://iit.iit.tuiasi.ro/TR/`, 2006.
3. A. Atanasiu: Arithmetic with membranes. In *Pre-proc. Workshop on Multiset Processing*, Curtea de Argeş, Romania, TR 140, CDMTCS, Univ. Auckland, 2000, 1–17.

4. M. Gheorghe: P systems – A modelling language. In *Pre-Proc. Unconventional Programming Paradigms, UPP04*, Le Mont Saint-Michel, September 2004, 23–27.
5. M.A. Gutiérrez-Naranjo, V. Rogozhin: Deductive databases and P systems. *Computer Science J. of Moldova*, 12, 1 (2004), 80–88.
6. I. Petre, L. Petre: Mobile ambients and P systems. *J. Univ. Computer Science*, 5, 9 (1999), 588–598.
7. Z. Qi, C. Fu, S. Shi, J. You: The P system based model for mobile transactions. In *Fourth Intern. Conf. on Computer and Information Theory*, CIT2004, 534–539.
8. Z. Qi, R. Rao, G. Xue, J. You: A new formal model based on P systems for mobile transactions. In *Proc. IEEE Intern. Conf. on Services Computing*, SCC2004, 16–22.
9. V. Rogozhin, E. Boian: Simulation of mobile ambients by P systems. Part 1, *Pre-proceedings of Workshop on Membrane Computing, WMC2003*, Tarragona, GRLMC Report 28/03, 404–427. Part 2, *Proc. Second Brainstorming Week on Membrane Computing*, Sevilla, 2004, RNGC Report 01/2004, 431–442.
10. V. Rogojin, E. Boian: Simulation of mobile ambients by tissue P systems with a dynamic network of membranes. In *Proc. ICCC, Oradea*, Ed. Univ. Oradea, 2004, 377–382.
11. Y. Suzuki, Y. Fujiwara, J. Takabayashi, H. Tanaka: Artificial life applications of a class of P systems: Abstract rewriting systems on multisets. In *Multiset Processing*, LNCS 2236, Springer, 2001, 299–346.
12. Y. Suzuki, H. Tanaka: Artificial life and P systems. In *Pre-proc. Workshop on Multiset Processing*, Curtea de Argeş, Romania, TR 140, CDMTCS, Univ. Auckland, 2000, 265–285.

References

1. Andrei, O., Ciobanu, G., Lucanu, D.: Executable specification of P systems. In: Mauri, G., Păun, G., Jesús Pérez-Jímenez, M., Rozenberg, G., Salomaa, A. (eds.) WMC 2004. LNCS, vol. 3365, pp. 126–145. Springer, Heidelberg (2005)
2. Ardelean, I.I., Cavaliere, M.: Modelling biological processes by using a probabilistic P system software. Natural Computing 2(2), 173–197 (2003)
3. Besozzi, D., Ciobanu, G.: A P systems description of the sodium-potassium pump. In: Mauri, G., Păun, G., Jesús Pérez-Jímenez, M., Rozenberg, G., Salomaa, A. (eds.) WMC 2004. LNCS, vol. 3365, pp. 210–223. Springer, Heidelberg (2005)
4. Bianco, L., Fontana, F., Manca, V.: P systems with reaction maps. International Journal of Foundations of Computer Science 17(1), 27–48 (2006)
5. Cardelli, L.: Brane calculi: Interactions of biological membranes. In: Danos, V., Schachter, V. (eds.) CMSB 2004. LNCS (LNBI), vol. 3082, pp. 257–278. Springer, Heidelberg (2005)
6. Cheruku, S., Paun, A., Romero-Campero, F.J., Pérez-Jiménez, M.J., Ibarra, O.H.: Simulating fas-induced apoptosis by using P systems. Progress in Natural Science 17(4), 424–431 (2007)

7. Ciobanu, G., Pan, L., Păun, G.: P systems with minimal parallelism. Theoretical Computer Science 378(1), 117–130 (2007)
8. Fontana, F., Bianco, L., Manca, V.: P systems and the modelling of biochemical oscillations. In: Freund, R., Păun, G., Rozenberg, G., Salomaa, A. (eds.) WMC 2005. LNCS, vol. 3850, pp. 199–208. Springer, Heidelberg (2006)
9. Fontana, F., Manca, V.: Discrete solutions to differential equations by metabolic P systems. Theoretical Computer Science 372(2-3), 165–182 (2007)
10. Freund, R.: P systems working in the sequential mode on arrays and strings. International Journal of Foundations of Computer Science 16(4), 663–682 (2005)
11. Gillespie, D.T.: Stochastic simulation of chemical kinetics. Annu. Rev. Phys. Chem. 58, 35–55 (2007)
12. Goss, P.J., Peccoud, J.: Quantitative modelling of stochastic system in molecular biology by using stochastic petri nets. Proc. Natl. Acad. Sci. USA 95, 6750–6755 (1998)
13. Heath, J., Kwiatkowska, M.Z., Norman, G., Parker, D., Tymchyshyn, O.: Probabilistic model checking of complex biological pathways. Theoretical Computer Science 391(3), 239–257 (2008)
14. Kwiatkowska, M., Norman, G., Parker, D.: Stochastic model checking. In: Bernardo, M., Hillston, J. (eds.) SFM 2007. LNCS, vol. 4486, pp. 220–270. Springer, Heidelberg (2007)
15. Li, C., Dang, Z., Ibarra, O.H., Yen, H.-C.: Signaling p systems and verification problems. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 1462–1473. Springer, Heidelberg (2005)
16. Milner, R.: Communication and Mobile Systems: The π-calculus. Cambridge University Press, Cambridge (1999)
17. Pérez-Jiménez, M.J., Romero-Campero, F.J.: P systems, a new computational modelling tool for systems biology. In: Transactions on Computational Systems Biology VI, pp. 176–197 (2006)
18. Pescini, D., Besozzi, D., Mauri, G., Zandron, C.: Dynamical probabilistic p systems. International Journal of Foundations of Computer Science 17(1), 183–195 (2006)
19. Ptashne, M., Gann, A.: Genes and Signals. Cold Spring Harbor Laboratory Press (2002)
20. Reddy, V., Liebman, M., Maverovouniotis, M.: Qualitative analysis of biochemical reaction systems. Computers in Biology and Medicine 26(1), 9–24 (1996)
21. Regev, A., Panina, E., Silvermann, W., Cardelli, L., Shapiro, E.: Bioambients: an abstraction for biological compartments. Theoretical Computer Science 325, 141–167 (2004)
22. Regev, A., Shapiro, E.: The π-calculus as an abstraction for biomolecular systems. In: Modelling in Molecular Biology, pp. 1–50. Springer, Berlin (2004)
23. Romero-Campero, F.J., Pérez-Jiménez, M.J.: A model of the quorum sensing system in vibrio fischeri using P systems. Artificial Life 14(1), 95–109 (2008)
24. Romero-Campero, F.J., Pérez-Jiménez, M.J.: Modelling gene expression control using P systems: The lac operon, a case study. BioSystems 91(3), 438–457 (2008)