The Calculus of Looping Sequences
for Modeling Biological Membranes

Roberto Barbuti[1], Andrea Maggiolo–Schettini[1],
Paolo Milazzo[1], and Angelo Troina[2,3]

[1] Dip. di Informatica, Università di Pisa, Largo B. Pontecorvo 3, 56127 - Pisa, Italy
[2] LIX - École Polytechnique Rue de Saclay, 91128 - Palaiseau, France
[3] LSV - ENS Cachan 61 Avenue du Président Wilson, 94235 - Cachan, France
{barbuti,maggiolo,milazzo,troina}@di.unipi.it

Abstract. We survey the formalism Calculus of Looping Sequences
(CLS) and a number of its variants from the point of view of their use for
describing biological membranes. The CLS formalism is based on term
rewriting and allows describing biomolecular systems. A first variant of
CLS, called Stochastic CLS, extends the formalism with stochastic time,
another variant, called LCLS (CLS with links), allows describing proteins
interaction at the domain level. A third variant is introduced for easier
description of biological membranes. This extension can be encoded into
CLS as well as other formalisms capable of membrane description such as
Brane Calculi and P Systems. Such encodings allow verifying and simu-
lating descriptions in Brane Calculi and P Systems by means of verifiers
and simulators developed for CLS.

1 Introduction

Cell biology, the study of the morphological and functional organization of cells,
is now an established field in biochemical research. Computer Science can help
the research in cell biology in several ways. For instance, it can provide biolo-
gists with models and formalisms capable of describing and analyzing complex
systems such as cells. In the last few years many formalisms originally developed
by computer scientists to model systems of interacting components have been
applied to Biology. Among these, there are Petri Nets [16], Hybrid Systems [1],
and the π-calculus [9,25]. Moreover, new formalisms have been defined for de-
scribing biomolecular and membrane interactions [2,7,8,11,21,23]. Others, such
as P Systems [17,18], have been proposed as biologically inspired computational
models and have been later applied to the description of biological systems.

The π–calculus and new calculi based on it [21,23] have been particularly suc-
cessful in the description of biological systems, as they allow describing systems
in a compositional manner. Interactions of biological components are modeled
as communications on channels whose names can be passed; sharing names of
private channels allows describing biological compartments. However, these cal-
culi offer very low–level interaction primitives, and this causes models to become

very large and difficult to read. Calculi such as those proposed in [7,8,11] give a more abstract description of systems and offer special biologically motivated operators. However, they are often specialized to the description of some particular kinds of phenomena such as membrane interactions or protein interactions. Finally, P Systems have a simple notation and are not specialized to the description of a particular class of systems, but they are still not completely general. For instance, it is possible to describe biological membranes and the movement of molecules across membranes, and there are some variants able to describe also more complex membrane activities. However, the formalism is not so flexible to allow describing easily new activities observed on membranes without extending the formalism to model such activities.

Therefore, we conclude that there is a need for a formalism having a simple notation, having the ability of describing biological systems at different levels of abstraction, having some notions of compositionality and being flexible enough to allow describing new kinds of phenomena as they are discovered, without being specialized to the description of a particular class of systems. For this reason in [3] we have introduced the Calculus of Looping Sequences (CLS).

CLS is a formalism based on term rewriting with some features, such as a commutative parallel composition operator, and some semantic means, such as bisimulations, which are common in process calculi. This permits to combine the simplicity of notation of rewriting systems with the advantage of a form of compositionality. Actually, in [4] we have defined bisimilarity relations on CLS terms which are congruences with respect to the operators. The bisimilarity relation may be used to verify a property of a system by assessing its bisimilarity with a system one knows to enjoy that property. The fact that bisimilarity is a congruence is very important for a compositional account of behavioral equivalence. In [5,6], we have defined two extensions of CLS. The first, Stochastic CLS, allows describing quantitative aspects of the modeled systems such as the time spent by occurrences of chemical reactions. The second, CLS with links, allows describing protein interaction more precisely at a lower level of abstraction, namely at the domain level.

In this paper, after recalling CLS and the two mentioned extensions, we focus on the modeling of biological membranes by means of CLS. Now, CLS does not offer an easy representation for membranes whose nature is fluid and for
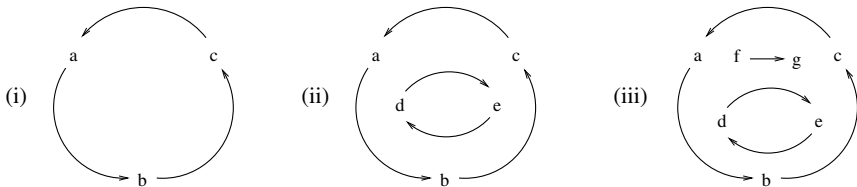

Fig. 1. (i) represents $\left(a \cdot b \cdot c\right)^{L}$; (ii) represents $\left(a \cdot b \cdot c\right)^{L} \rfloor \left(d \cdot e\right)^{L}$; (iii) represents $\left(a \cdot b \cdot c\right)^{L} \rfloor \left(\left(d \cdot e\right)^{L} \mid f \cdot g\right)$.

proteins which consequently move freely on membrane surfaces. For this reason, in [15] we have defined a CLS variant, called CLS+, which introduces a new operator allowing commutativity on membrane surfaces. We show how CLS+ can be encoded into CLS. In [3,15] we have shown how Brane Calculi [7] and P Systems [18] can be translated into CLS. Here we recall the ideas on which the translations are based.

CLS appears to allow description and manipulation of biological membranes and, moreover, offers, via translations, verification and simulation tools to other formalisms for membrane description.

2   The Calculus of Looping Sequences (CLS)

In this section we recall the Calculus of Looping Sequences (CLS) and we give some guidelines for the modeling of biological systems. CLS is essentially based on term rewriting, hence a CLS model consists of a term and a set of rewrite rules. The term is intended to represent the structure of the modeled system, and the rewrite rules to represent the events that may cause the system to evolve.

2.1   Formal Definition

We start with defining the syntax of terms. We assume a possibly infinite alphabet $\mathcal{E}$ of symbols ranged over by $a, b, c, \ldots$.

Definition 1 (Terms). *Terms T and sequences S of CLS are given by the following grammar:*

$$T \ ::= \ S \ \Big| \ \left(S\right)^L \rfloor T \ \Big| \ T \, | \, T$$
$$S \ ::= \ \epsilon \ \Big| \ a \ \Big| \ S \cdot S$$

where a is a generic element of $\mathcal{E}$, and ϵ represents the empty sequence. We denote with $\mathcal{T}$ the infinite set of terms, and with $\mathcal{S}$ the infinite set of sequences.

In CLS we have a sequencing operator $\_ \cdot \_$, a looping operator $\left(\_\right)^L$, a parallel composition operator $\_ \, | \, \_$ and a containment operator $\_ \rfloor \_$. Sequencing can be used to concatenate elements of the alphabet $\mathcal{E}$. The empty sequence ϵ denotes the concatenation of zero symbols. A term can be either a sequence or a looping sequence (that is the application of the looping operator to a sequence) containing another term, or the parallel composition of two terms. By definition, looping and containment are always applied together, hence we can consider them as a single binary operator $\left(\_\right)^L \rfloor \_$ which applies to one sequence and one term.

Brackets can be used to indicate the order of application of the operators, and we assume $\left(\_\right)^L \rfloor \_$ to have precedence over $\_ \, | \, \_$. In Figure 1 we show some examples of CLS terms and their visual representation.

In CLS we may have syntactically different terms representing the same structure. We introduce a structural congruence relation to identify such terms.

Definition 2 (Structural Congruence). *The structural congruence relations $\equiv_S$ and $\equiv_T$ are the least congruence relations on sequences and on terms, respectively, satisfying the following rules:*

$$S_1 \cdot (S_2 \cdot S_3) \equiv_S (S_1 \cdot S_2) \cdot S_3 \qquad S \cdot \epsilon \equiv_S \epsilon \cdot S \equiv_S S$$

$$S_1 \equiv_S S_2 \text{ implies } S_1 \equiv_T S_2 \text{ and } \left(S_1\right)^L \rfloor T \equiv_T \left(S_2\right)^L \rfloor T$$

$$T_1 \mid T_2 \equiv_T T_2 \mid T_1 \qquad T_1 \mid (T_2 \mid T_3) \equiv_T (T_1 \mid T_2) \mid T_3 \qquad T \mid \epsilon \equiv_T T$$

$$\left(\epsilon\right)^L \rfloor \epsilon \equiv_T \epsilon \qquad \left(S_1 \cdot S_2\right)^L \rfloor T \equiv_T \left(S_2 \cdot S_1\right)^L \rfloor T$$

Rules of the structural congruence state the associativity of $\cdot$ and $\mid$, the commutativity of the latter and the neutral role of ϵ. Moreover, axiom $\left(S_1 \cdot S_2\right)^L \rfloor T \equiv_T$ $\left(S_2 \cdot S_1\right)^L \rfloor T$ says that looping sequences can rotate. In the following, for simplicity, we will use $\equiv$ in place of $\equiv_T$.

Rewrite rules will be defined essentially as pairs of terms, with the first term describing the portion of the system in which the event modeled by the rule may occur, and the second term describing how that portion of the system changes when the event occurs. In the terms of a rewrite rule we allow the use of variables. As a consequence, a rule will be applicable to all terms which can be obtained by properly instantiating its variables. Variables can be of three kinds: two of these are associated with the two different syntactic categories of terms and sequences, and one is associated with single alphabet elements. We assume a set of term variables TV ranged over by $X, Y, Z, \ldots$, a set of sequence variables SV ranged over by $\widetilde{x}, \widetilde{y}, \widetilde{z}, \ldots$, and a set of element variables $\mathcal{X}$ ranged over by $x, y, z, \ldots$. All these sets are possibly infinite and pairwise disjoint. We denote by $\mathcal{V}$ the set of all variables, $\mathcal{V} = TV \cup SV \cup \mathcal{X}$, and with ρ a generic variable of $\mathcal{V}$. Hence, a pattern is a term that may include variables.

Definition 3 (Patterns). *Patterns P and sequence patterns SP of CLS are given by the following grammar:*

$$\begin{array}{rcl} P & ::= & SP \quad \mid \quad \left(SP\right)^L \rfloor P \quad \mid \quad P \mid P \quad \mid \quad X \\ SP & ::= & \epsilon \quad \mid \quad a \quad \mid \quad SP \cdot SP \quad \mid \quad \widetilde{x} \quad \mid \quad x \end{array}$$

where a is a generic element of $\mathcal{E}$, and $X, \widetilde{x}$ and x are generic elements of TV, SV and $\mathcal{X}$, respectively. We denote with $\mathcal{P}$ the infinite set of patterns.

We assume the structural congruence relation to be trivially extended to patterns. An *instantiation* is a partial function $\sigma : \mathcal{V} \to \mathcal{T}$. An instantiation must preserve the type of variables, thus for $X \in TV, \widetilde{x} \in SV$ and $x \in \mathcal{X}$ we have $\sigma(X) \in \mathcal{T}, \sigma(\widetilde{x}) \in \mathcal{S}$ and $\sigma(x) \in \mathcal{E}$, respectively. Given $P \in \mathcal{P}$, with $P\sigma$ we denote the term obtained by replacing each occurrence of each variable $\rho \in \mathcal{V}$ appearing in P with the corresponding term $\sigma(\rho)$. With Σ we denote the set of all the possible instantiations and, given $P \in \mathcal{P}$, with $Var(P)$ we denote the set of variables appearing in P. Now we define rewrite rules.

Definition 4 (Rewrite Rules). *A rewrite rule is a pair of patterns (P_1, P_2), denoted with $P_1 \mapsto P_2$, where $P_1, P_2 \in \mathcal{P}$, $P_1 \not\equiv \epsilon$ and such that $Var(P_2) \subseteq Var(P_1)$. We denote with $\Re$ the infinite set of all the possible rewrite rules.*

Table 1. Guidelines for the abstraction of biomolecular entities into CLS

Biomolecular Entity	CLS Term
Elementary object (genes, domains, other molecules, etc...)	Alphabet symbol
DNA strand	Sequence of elements repr. genes
RNA strand	Sequence of elements repr. transcribed genes
Protein	Sequence of elements repr. domains or single alphabet symbol
Molecular population	Parallel composition of molecules
Membrane	Looping sequence

A rewrite rule $P_1 \mapsto P_2$ states that a term $P_1\sigma$, obtained by instantiating variables in P_1 by some instantiation function σ, can be transformed into the term $P_2\sigma$. We define the semantics of CLS as a transition system, in which states correspond to terms, and transitions correspond to rule applications.

Definition 5 (Semantics). *Given a set of rewrite rules $\mathcal{R} \subseteq \Re$, the semantics of CLS is the least transition relation $\rightarrow$ on terms closed under $\equiv$, and satisfying the following inference rules:*

$$\frac{P_1 \mapsto P_2 \in \mathcal{R} \quad P_1\sigma \not\equiv \epsilon \quad \sigma \in \Sigma}{P_1\sigma \rightarrow P_2\sigma} \qquad \frac{T_1 \rightarrow T_2}{T \mid T_1 \rightarrow T \mid T_2} \qquad \frac{T_1 \rightarrow T_2}{(S)^L \rfloor T_1 \rightarrow (S)^L \rfloor T_2}$$

where the symmetric rule for the parallel composition is omitted.

A *model* in CLS is given by a term describing the initial state of the system and by a set of rewrite rules describing all the events that may occur.

2.2 Modeling Guidelines

We describe how CLS can be used to model biomolecular systems analogously to what done by Regev and Shapiro in [24] for the π–calculus. An abstraction is a mapping from a real–world domain to a mathematical domain, which may allow highlighting some essential properties of a system while ignoring other, complicating, ones. In [24], Regev and Shapiro show how to abstract biomolecular systems as concurrent computations by identifying the biomolecular entities and events of interest and by associating them with concepts of concurrent computations such as concurrent processes and communications. In particular, they give some guidelines for the abstraction of biomolecular systems to the π–calculus, and give some simple examples.

The use of rewrite systems, such as CLS, to describe biological systems is founded on a different abstraction. Usually, entities (and their structures) are abstracted by terms of the rewrite system, and events by rewriting rules. We have already introduced the biological interpretation of CLS operators in the previous section. Here we want to give more general guidelines.

Table 2. Guidelines for the abstraction of biomolecular events into CLS

Biomolecular Event	Examples of CLS Rewrite Rule
State change	$a \;\mapsto\; b$ $\widetilde{x} \cdot a \cdot \widetilde{y} \;\mapsto\; \widetilde{x} \cdot b \cdot \widetilde{y}$
Complexation	$a \mid b \;\mapsto\; c$ $\widetilde{x} \cdot a \cdot \widetilde{y} \mid b \;\mapsto\; \widetilde{x} \cdot c \cdot \widetilde{y}$
Decomplexation	$c \;\mapsto\; a \mid b$ $\widetilde{x} \cdot c \cdot \widetilde{y} \;\mapsto\; \widetilde{x} \cdot a \cdot \widetilde{y} \mid b$
Catalysis	$c \mid P_1 \;\mapsto\; c \mid P_2$ where $P_1 \;\mapsto\; P_2$ is the catalyzed event
State change on membrane	$\left(a \cdot \widetilde{x}\right)^L \rfloor X \;\mapsto\; \left(b \cdot \widetilde{x}\right)^L \rfloor X$
Complexation on membrane	$\left(a \cdot \widetilde{x} \cdot b \cdot \widetilde{y}\right)^L \rfloor X \;\mapsto\; \left(c \cdot \widetilde{x} \cdot \widetilde{y}\right)^L \rfloor X$ $a \mid \left(b \cdot \widetilde{x}\right)^L \rfloor X \;\mapsto\; \left(c \cdot \widetilde{x}\right)^L \rfloor X$ $\left(b \cdot \widetilde{x}\right)^L \rfloor (a \mid X) \;\mapsto\; \left(c \cdot \widetilde{x}\right)^L \rfloor X$
Decomplexation on membrane	$\left(c \cdot \widetilde{x}\right)^L \rfloor X \;\mapsto\; \left(a \cdot b \cdot \widetilde{x}\right)^L \rfloor X$ $\left(c \cdot \widetilde{x}\right)^L \rfloor X \;\mapsto\; a \mid \left(b \cdot \widetilde{x}\right)^L \rfloor X$ $\left(c \cdot \widetilde{x}\right)^L \rfloor X \;\mapsto\; \left(b \cdot \widetilde{x}\right)^L \rfloor (a \mid X)$
Catalysis on membrane	$\left(c \cdot \widetilde{x} \cdot SP_1 \cdot \widetilde{y}\right)^L \rfloor X \;\mapsto\; \left(c \cdot \widetilde{x} \cdot SP_2 \cdot \widetilde{y}\right)^L \rfloor X$ where $SP_1 \;\mapsto\; SP_2$ is the catalyzed event
Membrane crossing	$a \mid \left(\widetilde{x}\right)^L \rfloor X \;\mapsto\; \left(\widetilde{x}\right)^L \rfloor (a \mid X)$ $\left(\widetilde{x}\right)^L \rfloor (a \mid X) \;\mapsto\; a \mid \left(\widetilde{x}\right)^L \rfloor X$ $\widetilde{x} \cdot a \cdot \widetilde{y} \mid \left(\widetilde{z}\right)^L \rfloor X \;\mapsto\; \left(\widetilde{z}\right)^L \rfloor (\widetilde{x} \cdot a \cdot \widetilde{y} \mid X)$ $\left(\widetilde{z}\right)^L \rfloor (\widetilde{x} \cdot a \cdot \widetilde{y} \mid X) \;\mapsto\; \widetilde{x} \cdot a \cdot \widetilde{y} \mid \left(\widetilde{z}\right)^L \rfloor X$
Catalyzed membrane crossing	$a \mid \left(b \cdot \widetilde{x}\right)^L \rfloor X \;\mapsto\; \left(b \cdot \widetilde{x}\right)^L \rfloor (a \mid X)$ $\left(b \cdot \widetilde{x}\right)^L \rfloor (a \mid X) \;\mapsto\; a \mid \left(b \cdot \widetilde{x}\right)^L \rfloor X$ $\widetilde{x} \cdot a \cdot \widetilde{y} \mid \left(b \cdot \widetilde{z}\right)^L \rfloor X \;\mapsto\; \left(b \cdot \widetilde{z}\right)^L \rfloor (\widetilde{x} \cdot a \cdot \widetilde{y} \mid X)$ $\left(b \cdot \widetilde{z}\right)^L \rfloor (\widetilde{x} \cdot a \cdot \widetilde{y} \mid X) \;\mapsto\; \widetilde{x} \cdot a \cdot \widetilde{y} \mid \left(b \cdot \widetilde{z}\right)^L \rfloor X$
Membrane joining	$\left(\widetilde{x}\right)^L \rfloor (a \mid X) \;\mapsto\; \left(a \cdot \widetilde{x}\right)^L \rfloor X$ $\left(\widetilde{x}\right)^L \rfloor (\widetilde{y} \cdot a \cdot \widetilde{z} \mid X) \;\mapsto\; \left(\widetilde{y} \cdot a \cdot \widetilde{z} \cdot \widetilde{x}\right)^L \rfloor X$
Catalyzed membrane joining	$\left(b \cdot \widetilde{x}\right)^L \rfloor (a \mid X) \;\mapsto\; \left(a \cdot b \cdot \widetilde{x}\right)^L \rfloor X$ $\left(\widetilde{x}\right)^L \rfloor (a \mid b \mid X) \;\mapsto\; \left(a \cdot \widetilde{x}\right)^L \rfloor (b \mid X)$ $\left(b \cdot \widetilde{x}\right)^L \rfloor (\widetilde{y} \cdot a \cdot \widetilde{z} \mid X) \;\mapsto\; \left(\widetilde{y} \cdot a \cdot \widetilde{z} \cdot \widetilde{x}\right)^L \rfloor X$ $\left(\widetilde{x}\right)^L \rfloor (\widetilde{y} \cdot a \cdot \widetilde{z} \mid b \mid X) \;\mapsto\; \left(\widetilde{y} \cdot a \cdot \widetilde{z} \cdot \widetilde{x}\right)^L \rfloor (b \mid X)$
Membrane fusion	$\left(\widetilde{x}\right)^L \rfloor (X) \mid \left(\widetilde{y}\right)^L \rfloor (Y) \;\mapsto\; \left(\widetilde{x} \cdot \widetilde{y}\right)^L \rfloor (X \mid Y)$
Catalyzed membrane fusion	$\left(a \cdot \widetilde{x}\right)^L \rfloor (X) \mid \left(b \cdot \widetilde{y}\right)^L \rfloor (Y) \;\mapsto$ $\quad \left(a \cdot \widetilde{x} \cdot b \cdot \widetilde{y}\right)^L \rfloor (X \mid Y)$
Membrane division	$\left(\widetilde{x} \cdot \widetilde{y}\right)^L \rfloor (X \mid Y) \;\mapsto\; \left(\widetilde{x}\right)^L \rfloor (X) \mid \left(\widetilde{y}\right)^L \rfloor (Y)$
Catalyzed membrane division	$\left(a \cdot \widetilde{x} \cdot b \cdot \widetilde{y}\right)^L \rfloor (X \mid Y) \;\mapsto$ $\quad \left(a \cdot \widetilde{x}\right)^L \rfloor (X) \mid \left(b \cdot \widetilde{y}\right)^L \rfloor (Y)$

First of all, we select the biomolecular entities of interest. Since we want to describe cells, we consider molecular populations and membranes. Molecular populations are groups of molecules that are in the same compartment of the cell. Molecules can be of many types: we classify them as DNA and RNA strands, proteins, and other molecules. DNA and RNA strands and proteins can be seen as non–elementary objects. DNA strands are composed by genes, RNA strands are composed by parts corresponding to the transcription of individual genes, and proteins are composed by parts having the role of interaction sites (or domains). Other molecules are considered as elementary objects, even if they are complexes. Membranes are considered as elementary objects, in the sense that we do not describe them at the level of the lipids they are made of. The only interesting properties of a membrane are that it may contain something (hence, create a compartment) and that it may have molecules on its surface.

Now, we select the biomolecular events of interest. The simplest kind of event is the change of state of an elementary object. Then, we may have interactions between molecules: in particular complexation, decomplexation and catalysis. These interactions may involve single elements of non–elementary molecules (DNA and RNA strands, and proteins). Moreover, we may have interactions between membranes and molecules: in particular a molecule may cross or join a membrane. Finally, we may have interactions between membranes: in this case there may be many kinds of interactions (fusion, division, etc. . .).

The guidelines for the abstraction of biomolecular entities and events into CLS are given in Table 1 and Table 2, respectively. Entities are associated with CLS terms: elementary objects are modeled as alphabet symbols, non–elementary objects as CLS sequences and membranes as looping sequences. Biomolecular events are associated with CLS rewrite rules. In the figure we give some examples of rewrite rules for each type of event. The list of examples is not complete: one could imagine also rewrite rules for the description of complexation/decomplexation events involving more than two molecules, or catalysis events in which the catalyzing molecule is on a membrane and the catalyzed event occurs in its content, or more complex interactions between membranes. We remark that in the second example of rewrite rule associated with the complexation event we have that one of the two molecules which are involved should be either an elementary object or a protein modeled as a single alphabet symbol. As before, this is caused by the problem of modeling protein interaction at the domain level. This problem is solved by the extension of CLS with links, called LCLS, we shall describe in the following.

2.3   Examples

A well–known example of biomolecular system is the epidermal growth factor (EGF) signal transduction pathway[26,19]. If EGF proteins are present in the environment of a cell, they should be interpreted as a proliferation signal from the environment, and hence the cell should react by synthesizing proteins which stimulate its proliferation. A cell recognizes the EGF signal because it has on its membrane some EGF receptor proteins (EGFR), which are transmembrane

proteins (they have some intra–cellular and some extra–cellular domains). One of the extra–cellular domains binds to one EGF protein in the environment, forming a signal–receptor complex on the membrane. This causes a conformational change on the receptor protein that enables it to bind to another one signal–receptor complex. The formation of the binding of the two signal–receptor complexes (called dimerization) causes the phosphorylation of some intra–cellular domains of the dimer. This, in turn, causes the internal domains of the dimer to be recognized by a protein that is inside the cell (in the cytoplasm), called SHC. The protein SHC binds to the dimer, enabling a long chain of protein–protein interactions, which finally activate some proteins, such as one called ERK, which bind to the DNA and stimulate synthesis of proteins for cell proliferation.

Now, we use CLS to build a model of the first steps of the EGF signaling pathway up to the binding of the signal-receptor dimer to the SHC protein. In the following we shall refine the model by using the LCLS extension to describe interactions at the domain level.

We model the EGFR, EGF, and SHC proteins as the alphabet symbols $EGFR$, EGF and SHC, respectively. The cell is modeled as a looping sequence (representing its external membrane), initially composed only by $EGFR$ symbols, containing SHC symbols and surrounded by EGF symbols. The rewrite rules modeling the first steps of the pathway are the following:

$$EGF \mid \left(EGFR \cdot \widetilde{x}\right)^{L} \rfloor X \;\mapsto\; \left(CMPLX \cdot \widetilde{x}\right)^{L} \rfloor X \tag{R1}$$

$$\left(CMPLX \cdot \widetilde{x} \cdot CMPLX \cdot \widetilde{y}\right)^{L} \rfloor X \;\mapsto\; \left(DIM \cdot \widetilde{x} \cdot \widetilde{y}\right)^{L} \rfloor X \tag{R2}$$

$$\left(DIM \cdot \widetilde{x}\right)^{L} \rfloor X \;\mapsto\; \left(DIMp \cdot \widetilde{x}\right)^{L} \rfloor X \tag{R3}$$

$$\left(DIMp \cdot \widetilde{x}\right)^{L} \rfloor \left(SHC \mid X\right) \;\mapsto\; \left(DIMpSHC \cdot \widetilde{x}\right)^{L} \rfloor X \tag{R4}$$

Rule R1 describes the binding of a EGF protein to a EGFR receptor protein on the membrane surface. The result of the binding is a signal-receptor complex denoted $CMPLX$. Rule R2 describes the dimerization of two signal-receptor complex, the result is denoted DIM. Rule R3 describes the phosphorylation (and activation) of a signal-receptor dimer, that is the replacement of a DIM symbol with a $DIMp$ symbol. Finally, rule R4 describes the binding of an active dimer $DIMp$ with a SHC protein contained in the cytoplasm. The result is a $DIMpSHC$ symbol placed on the membrane surface.

A possible initial term for the model in this example is given by a looping sequence composed by some $EGFR$ symbols, containing some SHC symbols and with some EGF symbols outside. A possible evolution of such a term by means of application of the given rewrite rules is the following (we write on each transition the name of the rewrite rule applied):

$$EGF \mid EGF \mid \left(EGFR \cdot EGFR \cdot EGFR \cdot EGFR\right)^{L} \rfloor \left(SHC \mid SHC\right)$$

$$\xrightarrow{(R1)} EGF \mid \left(EGFR \cdot CMPLX \cdot EGFR \cdot EGFR\right)^{L} \rfloor \left(SHC \mid SHC\right)$$

$$\xrightarrow{(R1)} \left(EGFR \cdot CMPLX \cdot EGFR \cdot CMPLX\right)^{L} \rfloor \left(SHC \mid SHC\right)$$

$$\xrightarrow{(R2)} \ \left(EGFR \cdot DIM \cdot EGFR\right)^{L} \rfloor (SHC \mid SHC)$$

$$\xrightarrow{(R3)} \ \left(EGFR \cdot DIMp \cdot EGFR\right)^{L} \rfloor (SHC \mid SHC)$$

$$\xrightarrow{(R4)} \ \left(EGFR \cdot DIMpSHC \cdot EGFR\right)^{L} \rfloor SHC$$

We show another example of modeling of a biomolecular system with CLS, that is the modeling of a simple gene regulation process. This kind of processes are essential for cell life as they allow a cell to regulate the production of proteins that may have important roles for instance in metabolism, growth, proliferation and differentiation.

The example we consider is as follows: we have a simple DNA fragment consisting of a sequence of three genes. The first, denoted p, is called *promoter* and is the place where a *RNA polymerase* enzyme (responsible for translation of DNA into RNA) binds to the DNA. The second, denoted o, is called *operator* and it is the place where a *repressor* protein (responsible for regulating the activity of the RNA polymerase) binds to the DNA. The third, denoted as g, is the gene that encodes for the protein whose production is regulated by this process.

When the repressor is not bound to the DNA, the RNA polymerase can scan the sequence of genes and transcribe gene g into a piece of RNA that will be later translated into the protein encoded by g. When the repressor is bound to the DNA, it becomes an obstacle for the RNA polymerase that cannot scan any more the sequence of genes.

The CLS model of this simple regulation process is a follows. The sequence of genes is represented as the CLS sequence $p \cdot o \cdot g$, the RNA polymerase enzyme as $polym$, the repressor protein as $repr$, and the piece of RNA obtained by the translation of gene g as rna. The rewrite rules describing the process are the following:

$$polym \mid p \cdot \widetilde{x} \quad \mapsto \quad pp \cdot \widetilde{x} \tag{R1}$$
$$repr \mid \widetilde{x} \cdot o \cdot \widetilde{y} \quad \mapsto \quad \widetilde{x} \cdot ro \cdot \widetilde{y} \tag{R2}$$
$$pp \cdot o \cdot \widetilde{x} \quad \mapsto \quad p \cdot po \cdot \widetilde{x} \tag{R3}$$
$$\widetilde{x} \cdot po \cdot g \quad \mapsto \quad \widetilde{x} \cdot o \cdot pg \tag{R4}$$
$$\widetilde{x} \cdot pg \quad \mapsto \quad polym \mid rna \mid \widetilde{x} \cdot g \tag{R5}$$

Rules R1 and R2 describe the binding of the RNA polymerase and of the repressor to the corresponding genes in the DNA sequences. The results of these bindings are that the symbols representing the two genes are replaced by pp and ro, respectively. Rules R3, R4 and R5 describe the activity of the RNA polymerase enzyme in the absence of the repressor: it moves from gene p to gene o in rule R3, then it moves from gene o to gene g in rule R4, and finally it produces the RNA fragment and leaves the DNA in rule R5. Note that, in order to apply rule R3, the repressor must be not bound to the DNA.

The only possible evolution of a term representing an initial situation in which no repressors are present is

$$polym \mid p \cdot o \cdot g \xrightarrow{(R1)} pp \cdot o \cdot g \xrightarrow{(R3)} p \cdot po \cdot g$$
$$\xrightarrow{(R4)} p \cdot o \cdot pg \xrightarrow{(R5)} polym \mid rna \mid p \cdot o \cdot g$$
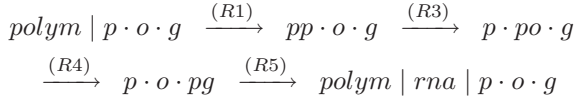
that represent the case in which the RNA polymerase enzyme can scan the DNA sequence and transcribe gene g into a piece of RNA. When the repressor is present, instead, a possible evolution is

$$repr \mid polym \mid p \cdot o \cdot g \xrightarrow{(R1)} repr \mid pp \cdot o \cdot g \xrightarrow{(R2)} pp \cdot ro \cdot g$$

and it corresponds to a situation in which the repressor stops the transcription of the gene by hampering the activity of the RNA polymerase.

3   Two Extensions of CLS

In this section we describe two extensions of CLS. The first, Stochastic CLS, allows describing quantitative aspects of the modeled systems, such as the time spent by occurrences of chemical reactions. The second, CLS with links, allows describing protein interaction more precisely at a lower level of abstraction, namely at the domain level.

3.1   Stochastic CLS

In CLS only qualitative aspects of biological systems are considered, such as their structure and the presence (or the absence) of certain molecules. As a consequence, on CLS models it is only possible to verify properties such as the reachability of particular states or causality relationships between events. It would be interesting to verify also properties such as the time spent to reach a particular state, or the probability of reaching it. To face this problem, in [6] we have developed a stochastic extension of CLS, called *Stochastic CLS*, in which quantitative aspects, such as time and probability are taken into account.
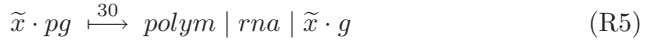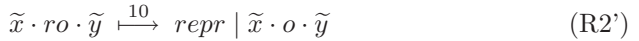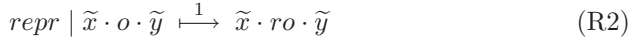
The standard way of extending a formalism to model quantitative aspects of biological systems is by incorporating the stochastic framework developed by Gillespie with its simulation algorithm for chemical reactions [12] in the semantics of the formalism. This has been done, for instance, for the π–calculus [20,22]. The idea of Gillespie's algorithm is that a rate constant is associated with each chemical reaction that may occur in the system. Such a constant is obtained by multiplying the kinetic constant of the reaction by the number of possible combinations of reactants that may occur in the system. The resulting rate constant is then used as the parameter of an exponential distribution modeling the time spent between two occurrences of the considered chemical reaction.

The use of exponential distributions to represent the (stochastic) time spent between two occurrences of chemical reactions allows describing the system as a

Continuous Time Markov Chain (CTMC), and consequently it allows verifying properties of the described system by means of analytic means and by means of stochastic model checkers.

In Stochastic CLS, incorporating Gillespie's stochastic framework is not a simple exercise. The main difficulty is counting the number of possible reactant combinations of the chemical reaction described by a rewrite rule. This means counting the number of different positions where the rewrite rule can be applied, by taking into account that rules may contain variables. We have defined the Stochastic CLS in [6], and showed how to derive a CTMC from the semantics of a system modeled in Stochastic CLS. This allows performing simulation and verification of properties of the described systems, for instance by using stochastic model checkers, such as PRISM [13].

Let us consider the simple regulation process we modeled with CLS in Section 2.3. We now extend the CLS model by including a kinetic constant in each rewrite rule. The result is a Stochastic CLS model. In order to make the model a little more realistic we add two rewrite rules describing the unbinding of the RNA polymerase and of the repressor from the DNA. Hence, the rewrite rules of the Stochastic CLS model are the following:

$$polym \mid p \cdot \widetilde{x} \overset{0.1}{\longmapsto} pp \cdot \widetilde{x} \tag{R1}$$

$$pp \cdot \widetilde{x} \overset{2}{\longmapsto} polym \mid p \cdot \widetilde{x} \tag{R1'}$$

$$repr \mid \widetilde{x} \cdot o \cdot \widetilde{y} \overset{1}{\longmapsto} \widetilde{x} \cdot ro \cdot \widetilde{y} \tag{R2}$$

$$\widetilde{x} \cdot ro \cdot \widetilde{y} \overset{10}{\longmapsto} repr \mid \widetilde{x} \cdot o \cdot \widetilde{y} \tag{R2'}$$

$$pp \cdot o \cdot \widetilde{x} \overset{100}{\longmapsto} p \cdot po \cdot \widetilde{x} \tag{R3}$$

$$\widetilde{x} \cdot po \cdot g \overset{100}{\longmapsto} \widetilde{x} \cdot o \cdot pg \tag{R4}$$

$$\widetilde{x} \cdot pg \overset{30}{\longmapsto} polym \mid rna \mid \widetilde{x} \cdot g \tag{R5}$$

We developed a simulator based on Stochastic CLS, and we used it to study the behaviour of the regulation process. In particular, we performed simulations by varying the quantity of repressors and we observed the production of RNA fragments in each case. The initial configuration of the system is given by the following term

$$\underbrace{repr \mid \dots \mid repr}_{n} \mid \underbrace{polym \mid \dots \mid polym}_{100} \mid p \cdot o \cdot g$$
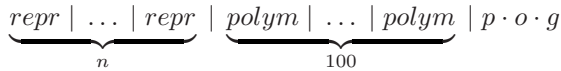
and we performed simulations with $n = 0, 10, 25$ and 50. The results of the simulations are shown in Figure 2. By varying the number of repressors from 0 to 50 the rate of transcription of the DNA into RNA molecules decreases.

3.2   CLS with Links (LCLS)

A formalism for modeling proteins interactions at the domain level was developed in the seminal paper by Danos and Laneve [11], and extended in [14]. This
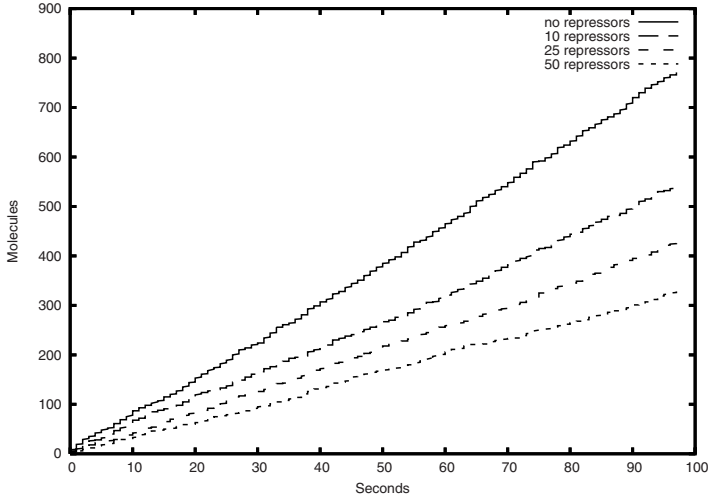
Fig. 2. Simulation result of the regulation process: number of RNA molecules over time

formalism allows expressing proteins by a node with a fixed number of domains; binding between domains allow complexating proteins. In this section we extend CLS to represent proteins interaction at the domain level. Such an extension, called Calculus of Linked Looping Sequences (LCLS), is obtained by labelling elementary components of sequences. Two elements with the same label are considered to be linked.

To model a protein at the domain level in CLS it would be natural to use a sequence with one symbol for each domain. However, the binding between two domains of two different proteins, that is the linking between two elements of two different sequences, cannot be expressed in CLS. To represent this, we extend CLS by labels on basic symbols. If in a term two symbols appear having the same label, we intend that they represent domains which are bound to each other. If in a term there is a symbol with a label and no other symbol with the same label, we intend that the term represents only a part of a system we model, and that the symbol will be linked to some other symbol in another part of the term representing the full model.

As membranes create compartments, elements inside a looping sequence cannot be linked to elements outside. Elements inside a membrane can be linked either to other elements inside the membrane or to elements of the membrane itself. An element can be linked at most to another element.

As an example, we model in LCLS the first steps of the EGF pathway described before. We model the EGFR protein as the sequence $R_{E1} \cdot R_{E2} \cdot R_{I1} \cdot R_{I2}$ where R_{E1} and R_{E2} are two extra–cellular domains and R_{I1} and R_{I2} are two intra–cellular domains. The membrane of the cell is modeled as a looping

sequence which could contain EGFR proteins. Outside the looping sequence (i.e. in the environment) there could be EGF proteins, and inside (i.e. in the cytoplasm) there could be SHC proteins. Rewrite rules modeling the pathway are the following:

$$EGF \mid \left(R_{E1} \cdot \widetilde{x} \right)^L \rfloor X \;\; \mapsto \;\; \left(sR_{E1} \cdot \widetilde{x} \right)^L \rfloor X \tag{R1}$$

$$\left(sR_{E1} \cdot R_{E2} \cdot x \cdot y \cdot \widetilde{x} \cdot sR_{E1} \cdot R_{E2} \cdot z \cdot w \cdot \widetilde{y} \right)^L \rfloor X \;\; \mapsto$$
$$\left(sR_{E1} \cdot R_{E2}^1 \cdot x \cdot y \cdot sR_{E1} \cdot R_{E2}^1 \cdot z \cdot w \cdot \widetilde{x} \cdot \widetilde{y} \right)^L \rfloor X \tag{R2}$$

$$\left(R_{E2}^1 \cdot R_{I1} \cdot \widetilde{x} \cdot R_{E2}^1 \cdot R_{I1} \cdot \widetilde{y} \right)^L \rfloor X \;\; \mapsto$$
$$\left(R_{E2}^1 \cdot PR_{I1} \cdot \widetilde{x} \cdot R_{E2}^1 \cdot R_{I1} \cdot \widetilde{y} \right)^L \rfloor X \tag{R3}$$

$$\left(R_{E2}^1 \cdot PR_{I1} \cdot \widetilde{x} \cdot R_{E2}^1 \cdot R_{I1} \cdot \widetilde{y} \right)^L \rfloor X \;\; \mapsto$$
$$\left(R_{E2}^1 \cdot PR_{I1} \cdot \widetilde{x} \cdot R_{E2}^1 \cdot PR_{I1} \cdot \widetilde{y} \right)^L \rfloor X \tag{R4}$$

$$\left(R_{E2}^1 \cdot PR_{I1} \cdot R_{I2} \cdot \widetilde{x} \cdot R_{E2}^1 \cdot PR_{I1} \cdot R_{I2} \cdot \widetilde{y} \right)^L \rfloor (SHC \mid X) \;\; \mapsto$$
$$\left(R_{E2}^1 \cdot PR_{I1} \cdot R_{I2}^2 \cdot \widetilde{x} \cdot R_{E2}^1 \cdot PR_{I1} \cdot R_{I2} \cdot \widetilde{y} \right)^L \rfloor (SHC^2 \mid X) \tag{R5}$$

Rule R1 represents the binding of the EGF protein to the receptor domain R_{E1} with sR_{E1} as a result. Rule R2 represents that when two EGFR proteins activated by proteins EGF occur on the membrane, they may bind to each other to form a dimer (shown by the link 1). Rule R3 represents the phosphorylation of one of the internal domains R_{I1} of the dimer, and rule R4 represents the phosphorylation of the other internal domain R_{I1} of the dimer. The result of each phosphorylation is pR_{I1}. Rule R5 represents the binding of the protein SHC in the cytoplasm to an internal domain R_{I2} of the dimer. Remark that the binding of SHC to the dimer is represented by the link 2, allowing the protein SHC to continue the interactions to stimulate cell proliferation.

Let us denote the sequence $R_{E1} \cdot R_{E2} \cdot R_{I1} \cdot R_{I2}$ by EGFR. By starting from a cell with some EGFR proteins on its membrane, some SHC proteins in the cytoplasm and some EGF proteins in the environment, a possible evolution is the following:

$$EGF \mid EGF \mid \left(EGFR{\cdot}EGFR{\cdot}EGFR \right)^L \rfloor (SHC \mid SHC)$$
$$\xrightarrow{(R1)} EGF \mid \left(sR_{E1}{\cdot}R_{E2}{\cdot}R_{I1}{\cdot}R_{I2}{\cdot}EGFR{\cdot}EGFR \right)^L \rfloor (SHC \mid SHC)$$
$$\xrightarrow{(R1)} \left(sR_{E1}{\cdot}R_{E2}{\cdot}R_{I1}{\cdot}R_{I2}{\cdot}EGFR{\cdot}sR_{E1}{\cdot}R_{E2}{\cdot}R_{I1}{\cdot}R_{I2} \right)^L \rfloor (SHC \mid SHC)$$
$$\xrightarrow{(R2)} \left(sR_{E1}{\cdot}R_{E2}^1{\cdot}R_{I1}{\cdot}R_{I2}{\cdot}sR_{E1}{\cdot}R_{E2}^1{\cdot}R_{I1}{\cdot}R_{I2}{\cdot}EGFR \right)^L \rfloor (SHC \mid SHC)$$

$$\xrightarrow{(R3)} \left(sR_{E1}{\cdot}R_{E2}^1{\cdot}pR_{I1}{\cdot}R_{I2}{\cdot}sR_{E1}{\cdot}R_{E2}^1{\cdot}R_{I1}{\cdot}R_{I2}{\cdot}EGFR\right)^L \rfloor (SHC \mid SHC)$$

$$\xrightarrow{(R4)} \left(sR_{E1}{\cdot}R_{E2}^1{\cdot}pR_{I1}{\cdot}R_{I2}{\cdot}sR_{E1}{\cdot}R_{E2}^1{\cdot}pR_{I1}{\cdot}R_{I2}{\cdot}EGFR\right)^L \rfloor (SHC \mid SHC)$$

$$\xrightarrow{(R5)} \left(sR_{E1}{\cdot}R_{E2}^1{\cdot}pR_{I1}{\cdot}R_{I2}^2{\cdot}sR_{E1}{\cdot}R_{E2}^1{\cdot}pR_{I1}{\cdot}R_{I2}{\cdot}EGFR\right)^L \rfloor (SHC^2 \mid SHC)$$

4   CLS and Membranes

What could seem strange in CLS is the use of looping sequences for the description of membranes, as sequencing is not a commutative operation and this do not correspond to the usual fluid representation of membranes in which objects can move freely. What one would expect is to have a multiset or a parallel composition of objects on a membrane. In the case of CLS, what could be used is a parallel composition of sequences. To address this problem, we define an extension of CLS, called CLS+, in which the looping operator can be applied to a parallel composition of sequences, and we show that we can translate quite easily CLS+ models into CLS ones.

4.1   Definition of CLS+

Terms in CLS+ are defined as follows.

Definition 6 (Terms). *Terms T, branes B, and sequences S of CLS+ are given by the following grammar:*

$$
\begin{aligned}
T &::= S \;\mid\; \left(B\right)^L \rfloor T \;\mid\; T \mid T \\
B &::= S \;\mid\; B \mid B \\
S &::= \epsilon \;\mid\; a \;\mid\; S \cdot S
\end{aligned}
$$

where a is a generic element of $\mathcal{E}$. We denote with $\mathcal{T}$ the infinite set of terms, with $\mathcal{B}$ the infinite set of branes, and with $\mathcal{S}$ the infinite set of sequences.

The structural congruence relation of CLS+ is a trivial extension of the one of CLS. The only difference is that commutativity of branes replaces rotation of looping sequences.

Definition 7 (Structural Congruence). *The structural congruence relations $\equiv_S$, $\equiv_B$ and $\equiv_T$ are the least congruence relations on sequences, on branes and on terms, respectively, satisfying the following rules:*

$$S_1 \cdot (S_2 \cdot S_3) \equiv_S (S_1 \cdot S_2) \cdot S_3 \qquad S \cdot \epsilon \equiv_S \epsilon \cdot S \equiv_S S$$

$$S_1 \equiv_S S_2 \; implies \; S_1 \equiv_B S_2$$

$$B_1 \mid B_2 \equiv_B B_2 \mid B_1 \qquad B_1 \mid (B_2 \mid B_3) \equiv_B (B_1 \mid B_2) \mid B_3 \qquad B \mid \epsilon \equiv_B B$$

$$S_1 \equiv_S S_2 \; implies \; S_1 \equiv_T S_2$$

$$B_1 \equiv_B B_2 \; implies \; \left(B_1\right)^L \, \rfloor \, T \equiv_T \left(B_2\right)^L \, \rfloor \, T$$

$$T_1 \mid T_2 \equiv_T T_2 \mid T_1 \quad T_1 \mid (T_2 \mid T_3) \equiv_T (T_1 \mid T_2) \mid T_3 \quad T \mid \epsilon \equiv_T T \quad \left(\epsilon\right)^L \, \rfloor \, \epsilon \equiv \epsilon$$

Now, to define patterns in CLS+ we consider an additional type of variables with respect of CLS, namely brane variables. We assume a set of brane variables BV ranged over by $\overline{x}, \overline{y}, \overline{z}, \ldots$.

Definition 8 (Patterns). Patterns P, brane patterns BP and sequence patterns SP of CLS+ are given by the following grammar:

$$
\begin{aligned}
P \;\; &::= \;\; SP \;\; \mid \;\; \left(BP\right)^L \, \rfloor \, P \;\; \mid \;\; P \mid P \;\; \mid \;\; X \\
BP \;\; &::= \;\; SP \;\; \mid \;\; BP \mid BP \;\; \mid \;\; \overline{x} \\
SP \;\; &::= \;\; \epsilon \;\; \mid \;\; a \;\; \mid \;\; SP \cdot SP \;\; \mid \;\; \widetilde{x} \;\; \mid \;\; x
\end{aligned}
$$

where a is a generic element of $\mathcal{E}$, and $X, \overline{x}, \widetilde{x}$ and x are generic elements of TV, BV, SV and $\mathcal{X}$, respectively. We denote with $\mathcal{P}$ the infinite set of patterns.

Definition 9 (Rewrite Rules). A rewrite rule is a pair of patterns (P_1, P_2), denoted with $P_1 \mapsto P_2$, where $P_1, P_2 \in PP$, $P_1 \not\equiv \epsilon$ and such that $Var(P_2) \subseteq Var(P_1)$. We denote with $\Re$ the infinite set of all the possible rewrite rules.

Now, differently from CLS, we have that a rule such as $a \mid b \mapsto c$ could be applied to elements of a looping sequence. For instance, $a \mid b \mapsto c$ can be applied to the term $\left(a \mid b\right)^L \, \rfloor \, d$ so to obtain the term $\left(c\right)^L \, \rfloor \, d$. However, a rule such as $\left(a\right)^L \, \rfloor \, b \mapsto c$ still cannot be applied to elements of a looping sequences, as $\left(\left(a\right)^L \, \rfloor \, b\right)^L \, \rfloor \, c$ is not a CLS+ term.

The rules that can be applied to elements of a looping sequence are those having the form (B_1, B_2) with $B_1, B_2 \in \mathcal{B}$. We call these rules *brane rules* and we denote as $\Re_\mathcal{B} \subset \Re$ their infinite set. Now, in the semantics of CLS+ we have to take into account brane rules and allow them to be applied also to elements of looping sequences. Hence, we define the semantics as follows.

Definition 10 (Semantics). Given a set of rewrite rules $\mathcal{R} \subseteq \Re$, and a set of brane rules $\mathcal{R}_\mathcal{B} \subseteq \mathcal{R}$, such that $(\mathcal{R} \setminus \mathcal{R}_\mathcal{B}) \cap \Re_\mathcal{B} = \varnothing$, the semantics of CLS is the least transition relation $\to$ on terms closed under $\equiv$, and satisfying the following inference rules:

$$\frac{(P_1, P_2) \in \mathcal{R} \quad P_1\sigma \not\equiv \epsilon \quad \sigma \in \Sigma}{P_1\sigma \to P_2\sigma} \qquad \frac{T_1 \to T_2}{T \mid T_1 \to T \mid T_2} \qquad \frac{T_1 \to T_2}{\left(B\right)^L \, \rfloor \, T_1 \to \left(B\right)^L \, \rfloor \, T_2}$$

$$\frac{(BP_1, BP_2) \in \mathcal{R}_\mathcal{B} \quad BP_1\sigma \not\equiv \epsilon \quad \sigma \in \Sigma}{BP_1\sigma \to_\mathcal{B} BP_2\sigma}$$

$$\frac{B_1 \to_\mathcal{B} B_2}{B \mid B_1 \to_\mathcal{B} B \mid B_2} \qquad \frac{B_1 \to_\mathcal{B} B_2}{\left(B_1\right)^L \, \rfloor \, T \to \left(B_2\right)^L \, \rfloor \, T}$$

where $\rightarrow_{\mathcal{B}}$ is a transition relation on branes, and where the symmetric rules for the parallel composition of terms and of branes are omitted.

In the definition of the semantics of CLS+ we use an additional transition relation $\rightarrow_{\mathcal{B}}$ on branes. This relation is used to describe the application of a brane rule to elements of a looping sequence. As usual, a CLS+ model is composed by a term, representing the initial state of the modeled system, and a set of rewrite rules.

In the following section we show that CLS+ models can be translated into CLS models. The translation into CLS is compositional and preserves the semantics of the model.

4.2 Translating CLS+ into CLS

The first step of the translation of a CLS+ model into CLS is a preprocessing procedure. For each brane rule (BP_1, BP_2) in the CLS+ model, we add to the set of rules of the model a new rule, namely $\left(\left(BP_1 \mid \overline{x} \right)^L \rfloor X, \left(BP_2 \mid \overline{x} \right)^L \rfloor X \right)$. This new rule is redundant in the model, as every time it can be applied to a CLS+ term, also the original one can be applied with the same result. However, the translation we are going to define will translate the original rule into a CLS rule that will be applicable only inside looping sequences, or at the top level of the term, and will translate the new rule into a CLS rule applicable only to elements that compose a looping sequence.

Now, the translation of CLS+ into CLS consists mainly of an encoding function, denoted $\{\!\!|\cdot|\!\!\}$, which maps CLS+ patterns into CLS patterns. This encoding function will be used to translate each rewrite rule of the CLS+ model into a rewrite rule for the corresponding CLS model, and to translate the term representing the initial state of the system in the CLS+ model into a CLS term for the corresponding CLS model.

The encoding function for CLS+ patterns is defined as follows. We assume a total and injective function from brane variables into a subset of term variables that are never used in CLS models. More easily, we assume brane variables to be a subset of the term variables of CLS. Moreover, we assume *in* and *out* to be symbols of the alphabet $\mathcal{E}$ never used in CLS models.

The encoding follows the "ball–bearing" technique described by Cardelli in [7]. Intuitively, every CLS+ looping sequence is translated into a couple of CLS looping sequences, one contained in the other, with the brane patterns of the CLS+ looping sequence between the two corresponding CLS looping sequences.

Definition 11 (Encoding Function). *The encoding function $\{\!\!|\cdot|\!\!\}$ maps CLS+ patterns into CLS patterns, and is given by the following recursive definition:*

$$\{\!\![SP]\!\!\} = SP$$

$$\{\!\![X]\!\!\} = X$$

$$\{\!\![(BP)^L \rfloor P]\!\!\} = \left(out \right)^L \rfloor \left(BP \mid \left(in \right)^L \rfloor \{\!\![P]\!\!\} \right)$$

$$\{\!\![P_1 \mid P_2]\!\!\} = \{\!\![P_1]\!\!\} \mid \{\!\![P_2]\!\!\}$$

A CLS rewrite rule is obtained from each CLS+ rewrite rule of the translated model by applying the encoding function to the two patterns of the rule. More precisely, given a CLS+ rule $P_1 \mapsto P_2$, the corresponding CLS rule is $(in)^L \rfloor (\{\![P_1]\!\} \mid X) \mapsto (in)^L \rfloor (\{\![P_2]\!\} \mid X)$ where X is a term variable that does not occur in P_1 and P_2. For example, by applying the encoding to the two patterns of the CLS+ rewrite rule

$$R = b \cdot x \mid c \mapsto b \cdot x$$

we obtain

$$R_{\{\![\cdot]\!\}} = (in)^L \rfloor (b \cdot x \mid c \mid X) \mapsto (in)^L \rfloor (b \cdot x \mid X).$$

The encoding of a CLS+ term into a CLS term is as follows: given a CLS+ term T the corresponding CLS term is $(in)^L \rfloor \{\![T]\!\}$. In this case we have that the encoding function never encounters variables. Consider, as an example, the following CLS+ term:

$$T = a \mid \left(c \mid d \mid b \cdot b \mid d \right)^L \rfloor d$$

the corresponding CLS term is as follows:

$$T_{\{\![\cdot]\!\}} = (in)^L \rfloor (a \mid (out)^L \rfloor (c \mid d \mid b \cdot b \mid d \mid (in)^L \rfloor d))$$

Now, it is easy to see that R can be applied to T, because parallel components in the looping sequence can be commuted, and the result of the application is

$$T' = a \mid \left(b \cdot b \mid d \mid d \right)^L \rfloor d$$

but the corresponding CLS rewrite rule $R_{\{\![\cdot]\!\}}$ cannot be applied to $T_{\{\![\cdot]\!\}}$. However, we have that $R \in \mathcal{R}_\mathcal{B}$, and hence, by the preprocessing phase described above, we have that also

$$R' = \left(b \cdot x \mid c \mid \overline{x} \right)^L \rfloor X \mapsto \left(b \cdot x \mid \overline{x} \right)^L \rfloor X$$

is a rule of the CLS+ model. By translating rule R' we obtain

$$R'_{\{\![\cdot]\!\}} = (in)^L \rfloor ((out)^L \rfloor (b \cdot x \mid c \mid \overline{x} \mid (in)^L \rfloor X) \mid Y) \mapsto$$
$$(in)^L \rfloor ((out)^L \rfloor (b \cdot x \mid \overline{x} \mid (in)^L \rfloor X) \mid Y)$$

that can be applied to $T_{\{\![\cdot]\!\}}$. The result of the application is

$$(in)^L \rfloor (a \mid (out)^L \rfloor (b \cdot b \mid d \mid d \mid (in)^L \rfloor d))$$

that corresponds exactly to the encoding of T'.

4.3 CLS, Brane Calculi and P Systems

Brane Calculi are a family of process calculi specialized in the description of membrane activity, and they allow associating processes with membranes of a membrane structure. Each process is composed by actions whose execution has an effect on the membrane structure. Some examples of actions are phagocytosis (a membrane engulfs another one), exocytosis (a membrane expels another one), and pinocytosis (a new membrane is created inside another one). These three actions are enough to define the simplest of Brane Calculi, namely the PEP calculus. Other actions, such as fusion of membranes and mitosis can be used to define different calculi of the family. Moreover, extensions of Brane Calculi allow describing interactions with molecules and complexes, such as letting them enter and exit membranes.

We have given a sound and complete encoding of the PEP calculus into CLS in [3,15]. Here, to recall shortly the encoding technique, we give a very simple example of PEP system and we show its translation into CLS. The PEP system we consider is the following

$$\phi(\!|\diamond|\!) \circ \phi^{\perp}(\mathbf{0})(\!|\diamond|\!)$$

representing two adjacent membranes $\phi_n(\!|\diamond|\!)$ and $\phi_n^{\perp}(\mathbf{0})(\!|\diamond|\!)$ ($\circ$ denotes juxtaposition) both containing nothing of relevant (what is between brackets $(\!|\ |\!)$ is the content of the membrane and $\diamond$ is the null system). The processes associated with the two membranes are ϕ and $\phi^{\perp}(\mathbf{0})$, respectively, representing two complementary phagocytosis actions: the first says that the membrane it is associated with can be engulfed by another membrane, and the second that the membrane it is associated with can engulf another membrane, that will be sourrounded by another new membrane whose associated process is the parameter of the action (in this case it is the idle process $\mathbf{0}$). Hence, in accordance with the semantics of the PEP calculus, we have that the only transition that can be performed by the system is the following, leading to a system that is equivalent to the null system $\diamond$:

$$\phi(\!|\diamond|\!) \circ \phi^{\perp}(\mathbf{0})(\!|\diamond|\!) \quad \rightarrow \quad \mathbf{0}(\!|\mathbf{0}(\!|\mathbf{0}(\!|\diamond|\!)|\!)|\!) \quad \equiv \quad \diamond$$

By applying the encoding to the system we obtain the following CLS term T:

$$act \cdot circ \cdot e \cdot brane \cdot b \cdot \phi \cdot a \cdot \mathbf{0} \cdot a \cdot b \cdot \mathbf{0} \cdot e \cdot brane \cdot d \cdot \phi^{\perp} \cdot c \cdot \mathbf{0} \cdot c \cdot \mathbf{0} \cdot c \cdot d \cdot \mathbf{0}$$

where act is a sort of program counter that precedes the symbol representing the next action to be executed, symbol $circ$ represents $\circ$, symbol $brane$ represents a membrane $(\!|\ |\!)$, symbols ϕ and $\phi^{\perp}$ represent the corresponding actions, symbol $\mathbf{0}$ represents the idle process and symbols a, b, c, d and e are used as separators of actions and parameters. The translation consists also of a set of CLS rewrite rules to be applied to terms obtained by the encoding of PEP systems. Such a set of rewrite rules does not depend on the encoded PEP system, hence it is always the same. By applying rewrite rules, the long sequence obtained from the encoding is transformed into a hierarchy of looping sequences corresponding to the membrane hierarchy in the original PEP system, then rewrite rules are

applied that correspond to the semantics of the actions occurring in the processes associated with membranes.

Hence, by means of application of rewrite rules, the result of the encoding of the PEP system may evolve as follows (where $\rightarrow^*$ represents a sequence of rewrite rule applications):

$$
\begin{aligned}
T \quad \rightarrow \quad & act \cdot brane \cdot b \cdot \phi \cdot a \cdot \mathbf{0} \cdot a \cdot b \cdot \mathbf{0} \mid act \cdot brane \cdot d \cdot \phi^\perp \cdot c \cdot \mathbf{0} \cdot c \cdot \mathbf{0} \cdot c \cdot d \\
\rightarrow^* \quad & \left(act \cdot \phi \cdot a \cdot \mathbf{0} \cdot a \right)^L \rfloor act \cdot \mathbf{0} \mid \left(act \cdot \phi^\perp \cdot c \cdot \mathbf{0} \cdot c \cdot \mathbf{0} \cdot c \right)^L \rfloor act \cdot \mathbf{0} \\
\rightarrow \quad & \left(act \cdot \mathbf{0} \right)^L \rfloor \left(act \cdot \mathbf{0} \mid \left(act \cdot \mathbf{0} \right)^L \rfloor \left(act \cdot \mathbf{0} \right)^L \rfloor act \cdot \mathbf{0} \right) \\
\rightarrow^* \quad & act \cdot \mathbf{0}
\end{aligned}
$$

Differently from Brane Calculi, P Systems (in their most common formulation) do not allow describing complex membrane activities such as phagocytosis and exocytosis. However, they are specialized in the description of reactions between molecules which are placed in a compartment of a complex membrane structure.

A P System is a membrane structure (a nesting of membranes) in which there could be multisets of objects representing molecules. A set of multiset rewrite rules is associated with each membrane, and describe the reactions that may occur between the molecules contained in the membrane. The result of the application of a rewrite rule can either remain in the same membrane, or exit the membrane, or enter an inner membrane. Priorities can be imposed on rewrite rules, meaning that some rules can be applied only if some others cannot, and it is possible for a membrane to dissolve and release its content to in the environment.

A peculiarity of P Systems is that rewrite rules are applied in a fully–parallel manner, namely in one step of evolution of the system all rules are applied as many times as possible (to different molecules), and this is one of the main differences with respect to CLS in which at each step one only rewrite rule is applied. We show that P Systems can be translated into CLS, and that the execution of a (fully parallel) step of a P System is simulated by a sequence of steps in CLS. A variant of P Systems, called Sequential P Systems, in which rules are applied sequentially is described in [10]. We do not consider the translation of this variant into CLS as it would be quite trivial and of little interest.

To recall the encoding technique, we give a simple example of P System and we show its translation into CLS. We focus on the translation of multiple parallelism, hence we consider a P System (depicted in Figure 3) consisting of a single membrane with only two rules, without priorities and without membrane dissolutions. We give a simplified translation: more details can be found in [15].

The alphabet of objects in the considered P System is $\{a, b, c\}$. A multiset of objects from this alphabet is represented by a CLS term as follows: let n_a, n_b and n_c be the number of occurrences of a, b and c in the multiset, respectively, then

1

$$abb$$
$$r_1 : a \rightarrow (ab, here)$$
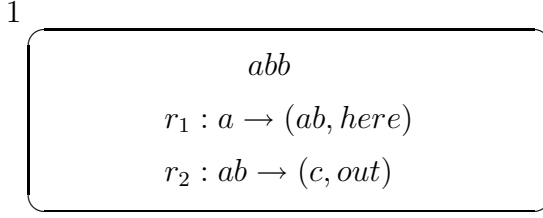$$r_2 : ab \rightarrow (c, out)$$

Fig. 3. A simple example of a P system

$$a \cdot \overbrace{1 \cdot \ldots \cdot 1}^{n_a} \mid b \cdot \overbrace{1 \cdot \ldots \cdot 1}^{n_b} \mid c \cdot \overbrace{1 \cdot \ldots \cdot 1}^{n_c}$$

is the term representing the multiset. We choose this representation as it allows us checking whether an object is absent, by checking whether the corresponding symbol if followed by zero 1s. An empty multiset is represented as $a \mid b \mid c$.

The CLS term obtained by the translation of the considered P System is the following:

$$(1)^L \rfloor (Check \mid a \cdot 1 \mid b \cdot 1 \cdot 1 \mid c \mid r_1 \mid r_2 \mid (next)^L \rfloor (a \mid b \mid c))$$

where the membrane of the P System is represented by a looping sequence composed by the membrane label (in this case 1). Inside the looping sequence there is a *Check* symbol representing the current state of the system, the translation of the multiset of objects of the membrane, two symbols r_1 and r_2 corresponding to the evolutionary rules of the membrane, and an empty multiset surrounded by a looping sequence *next*. This empty multiset is used to store temporary information on the result of the application of evolutionary rules.

The CLS rewrite rules obtained by the encoding of the considered P System are the following:

$$(1)^L \rfloor (X \mid Check \mid a \cdot 1 \cdot \widetilde{x} \mid r_1) \mapsto (1)^L \rfloor (X \mid Check \mid a \cdot 1 \cdot \widetilde{x} \mid r_1 \cdot 1) \quad \text{(C1)}$$

$$(1)^L \rfloor (X \mid Check \mid a \mid r_1) \mapsto (1)^L \rfloor (X \mid Check \mid a \mid r_1 \cdot 0) \quad \text{(C2)}$$

$$(1)^L \rfloor (X \mid Check \mid a \cdot 1 \cdot \widetilde{x} \mid b \cdot 1 \cdot \widetilde{y} \mid r_1 \cdot z \mid r_2) \mapsto$$
$$(1)^L \rfloor (X \mid Check \mid a \cdot 1 \cdot \widetilde{x} \mid b \cdot 1 \cdot \widetilde{y} \mid r_1 \cdot z \mid r_2 \cdot 1) \quad \text{(C3)}$$

$$(1)^L \rfloor (X \mid Check \mid a \mid r_1 \cdot z \mid r_2) \mapsto (1)^L \rfloor (X \mid Run \mid a \mid r_1 \cdot z \mid r_2 \cdot 0) \quad \text{(C4)}$$

$$\left(1\right)^{L} \rfloor \left(X \mid Check \mid b \mid r_1 \cdot z \mid r_2\right) \;\mapsto\; \left(1\right)^{L} \rfloor \left(X \mid Run \mid b \mid r_1 \cdot z \mid r_2 \cdot 0\right) \quad \text{(C5)}$$

$$\left(1\right)^{L} \rfloor \left(X \mid Run \mid a \cdot 1 \cdot \widetilde{x} \mid r_1 \cdot 1 \mid \left(next\right)^{L} \rfloor \left(Y \mid a \cdot \widetilde{y} \mid b \cdot \widetilde{z}\right)\right) \;\mapsto\;$$
$$\left(1\right)^{L} \rfloor \left(X \mid Run \mid a \cdot \widetilde{x} \mid r_1 \cdot 1 \mid \left(next\right)^{L} \rfloor \left(Y \mid a \cdot 1 \cdot \widetilde{y} \mid b \cdot 1 \cdot \widetilde{z}\right)\right) \quad \text{(R1)}$$

$$\left(1\right)^{L} \rfloor \left(X \mid Run \mid a \cdot 1 \cdot \widetilde{x} \mid b \cdot 1 \cdot \widetilde{y} \mid r_2 \cdot 1 \mid \left(next\right)^{L} \rfloor \left(Y \mid c \cdot \widetilde{z}\right)\right) \;\mapsto\;$$
$$\left(1\right)^{L} \rfloor \left(X \mid Run \mid a \cdot \widetilde{x} \mid b \cdot \widetilde{y} \mid r_2 \cdot 1 \mid \left(next\right)^{L} \rfloor \left(Y \mid c \cdot 1 \cdot \widetilde{z}\right)\right) \quad \text{(R2)}$$

$$\left(1\right)^{L} \rfloor \left(X \mid Run \mid a \mid r_1 \cdot 1\right) \;\mapsto\; \left(1\right)^{L} \rfloor \left(X \mid Run \mid a \mid r_1 \cdot 0\right) \quad \text{(R3)}$$

$$\left(1\right)^{L} \rfloor \left(X \mid Run \mid a \mid r_2 \cdot 1\right) \;\mapsto\; \left(1\right)^{L} \rfloor \left(X \mid Run \mid a \mid r_2 \cdot 0\right) \quad \text{(R4)}$$

$$\left(1\right)^{L} \rfloor \left(X \mid Run \mid b \mid r_2 \cdot 1\right) \;\mapsto\; \left(1\right)^{L} \rfloor \left(X \mid Run \mid b \mid r_2 \cdot 0\right) \quad \text{(R5)}$$

$$\left(1\right)^{L} \rfloor \left(X \mid Run \mid r_1 \cdot 0 \mid r_2 \cdot 0\right) \;\mapsto\; \left(1\right)^{L} \rfloor \left(X \mid Update \mid r_1 \cdot 0 \mid r_2 \cdot 0\right) \quad \text{(R6)}$$

$$\left(1\right)^{L} \rfloor \left(X \mid Update \mid x \cdot \widetilde{x} \mid \left(next\right)^{L} \rfloor \left(Y \mid x \cdot 1 \cdot \widetilde{y}\right)\right) \;\mapsto\;$$
$$\left(1\right)^{L} \rfloor \left(X \mid Update \mid x \cdot 1 \cdot \widetilde{y} \cdot \widetilde{x} \mid \left(next\right)^{L} \rfloor \left(Y \mid x\right)\right) \quad \text{(U1)}$$

$$\left(1\right)^{L} \rfloor \left(X \mid Update \mid \left(next\right)^{L} \rfloor \left(a \mid b \mid c\right)\right) \;\mapsto\;$$
$$\left(1\right)^{L} \rfloor \left(X \mid Check \mid \left(next\right)^{L} \rfloor \left(a \mid b \mid c\right)\right) \quad \text{(U2)}$$

Rules (C1)–(C5) describe the steps performed by the system while it is in *Check* state: the objective of this phase is to test whether each evolutionary rule is applicable or not. When all rules have been tested, the systems moves into a state called *Run*, whose steps are given by the application of rules (R1)–(R6). In this second phase, evolutionary rules previously identified as applicable are actually applied, and the result of the application is stored inside the looping sequence *next*. Finally, when no evolutionary rule is further applicable, the system moves into a state called *Update*, in which the content of the looping sequence *next* is used to reset the multiset of objects of the membrane by applying rule (U1)–(U2). When this update operation has been performed, the system moves back to the *Check* state.

5   Conclusions

We have surveyed the formalism CLS and a number of its variants from the point of view of its use for describing biological membranes. Verification and simulation tools have been developed for CLS and its variants and can be used to study properties of membrane systems. Via translations, these tools can be used to study systems described by other formalisms such as Brane Calculi and P Systems, capable of describing biological membranes.

References

1. Alur, R., Belta, C., Ivancic, F., Kumar, V., Mintz, M., Pappas, G.J., Rubin, H., Schug, J.: Hybrid Modeling and Simulation of Biomolecular Networks. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 19–32. Springer, Heidelberg (2001)
2. Barbuti, R., Cataudella, S., Maggiolo-Schettini, A., Milazzo, P., Troina, A.: A Probabilistic Model for Molecular Systems. Fundamenta Informaticae 67, 13–27 (2005)
3. Barbuti, R., Maggiolo-Schettini, A., Milazzo, P., Troina, A.: A Calculus of Looping Sequences for Modelling Microbiological Systems. Fundamenta Informaticae 72, 21–35 (2006)
4. Barbuti, R., Maggiolo-Schettini, A., Milazzo, P., Troina, A.: Bisimulation Congruences in the Calculus of Looping Sequences. In: Barkaoui, K., Cavalcanti, A., Cerone, A. (eds.) ICTAC 2006. LNCS, vol. 4281, pp. 93–107. Springer, Heidelberg (2006)
5. Barbuti, R., Maggiolo-Schettini, A., Milazzo, P.: Extending the Calculus of Looping Sequences to Model Protein Interaction at the Domain Level. In: Mandoiu, I., Zelikovsky, A. (eds.) ISBRA 2007. LNCS (LNBI), vol. 4463, pp. 638–649. Springer, Heidelberg (2007)
6. Barbuti, R., Maggiolo-Schettini, A., Milazzo, P., Tiberi, P., Troina, A.: Stochastic CLS for the Modeling and Simulation of Biological Systems (submitted for publication), draft available at: http://www.di.unipi.it/~milazzo/
7. Cardelli, L.: Brane Calculi. Interactions of Biological Membranes. In: Danos, V., Schachter, V. (eds.) CMSB 2004. LNCS (LNBI), vol. 3082, pp. 257–280. Springer, Heidelberg (2005)
8. Chabrier-Rivier, N., Chiaverini, M., Danos, V., Fages, F., Schachter, V.: Modeling and Querying Biomolecular Interaction Networks. Theoretical Computer Science 325(1), 25–44 (2004)
9. Curti, M., Degano, P., Priami, C., Baldari, C.T.: Modelling Biochemical Pathways through Enhanced pi-calculus. Theoretical Computer Science 325, 111–140 (2004)
10. Dang, Z., Ibarra, O.H.: On P Systems Operating in Sequential and Limited Parallel Modes. In: Workshop on Descriptional Complexity of Formal Systems, pp. 164–177 (2004)
11. Danos, V., Laneve, C.: Formal Molecular Biology. Theoretical Computer Science 325, 69–110 (2004)
12. Gillespie, D.: Exact Stochastic Simulation of Coupled Chemical Reactions. Journal of Physical Chemistry 81, 2340–2361 (1977)
13. Kwiatkowska, M., Norman, G., Parker, D.: Probabilistic Symbolic Model Checking with PRISM: a Hybrid Approach. Int. Journal on Software Tools for Technology Transfer 6, 128–142 (2004)
14. Laneve, C., Tarissan, F.: A Simple Calculus for Proteins and Cells. In: MeCBIC 2006. Workshop on Membrane Computing and Biological Inspired Process Calculi (to appear in ENTCS)
15. Milazzo, P.: Qualitative and Quantitative Formal Modeling of Biological Systems. PhD Thesis, Università di Pisa (2007)
16. Matsuno, H., Doi, A., Nagasaki, M., Miyano, S.: Hybrid Petri Net Representation of Gene Regulatory Network. In: Pacific Symposium on Biocomputing, pp. 341–352. World Scientific Press, Singapore (2000)

17. Păun, G.: Computing with Membranes. Journal of Computer and System Sciences 61, 108–143 (2000)
18. Păun, G.: Membrane Computing. An Introduction. Springer, Heidelberg (2002)
19. Pérez–Jiménez, M.J., Romero–Campero, F.J.: A Study of the Robustness of the EGFR Signalling Cascade Using Continuous Membrane Systems. In: Mira, J.M., Álvarez, J.R. (eds.) IWINAC 2005. LNCS, vol. 3561, pp. 268–278. Springer, Heidelberg (2005)
20. Priami, C.: Stochastic π–Calculus. The Computer Journal 38, 578–589 (1995)
21. Priami, C., Quaglia, P.: Beta Binders for Biological Interactions. In: Danos, V., Schachter, V. (eds.) CMSB 2004. LNCS (LNBI), vol. 3082, pp. 20–33. Springer, Heidelberg (2005)
22. Priami, C., Regev, A., Silvermann, W., Shapiro, E.: Application of a Stochastic Name–Passing Calculus to Representation and Simulation of Molecular Processes. Information Processing Letters 80, 25–31 (2001)
23. Regev, A., Panina, E.M., Silverman, W., Cardelli, L., Shapiro, E.: BioAmbients: An Abstraction for Biological Compartments. Theoretical Computer Science 325, 141–167 (2004)
24. Regev, A., Shapiro, E.: The π–Calculus as an Abstraction for Biomolecular Systems. In: Modelling in Molecular Biology. Natural Computing Series, pp. 219–266. Springer, Heidelberg (2004)
25. Regev, A., Silverman, W., Shapiro, E.Y.: Representation and Simulation of Biochemical Processes Using the pi-calculus Process Algebra. In: Pacific Symposium on Biocomputing, pp. 459–470. World Scientific Press, Singapore (2001)
26. Wiley, H.S., Shvartsman, S.Y., Lauffenburger, D.A.: Computational Modeling of the EGF–Receptor System: a Paradigm for Systems Biology. Trends in Cell Biology 13, 43–50 (2003)