Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

MODELING AND ANALYSIS OF BIOLOGICAL PROCESSES BY MEM(BRANE) CALCULI AND SYSTEMS

Nadia Busi

Department of Computer Science
University of Bologna
40127 Bologna, ITALY

Claudio Zandron

Department of Informatics,
Systems and Communication
University of Milano-Bicocca
20126 Milano, ITALY

ABSTRACT

In recent years, the modeling and analysis techniques developed in the area of formal languages and of concurrent process calculi have been successfully applied to the field of Systems Biology. In this setting, Brane Calculi and Membrane Systems are two of the most prominent approaches for the modeling of the behaviour of biological membranes. Membrane Systems have been introduced by Gh. Păun as a class of distributed parallel computing devices of a biochemical type, while Brane Calculi are a family of process calculi, based on a set of biologically inspired primitives of membrane interaction. In this paper we model the behaviour of a biological process – namely, the LDL Cholesterol Degradation Pathway – in both Brane Calculi and Membrane Systems. We also provide a brief discussion on the application of analysis techniques to this case study.

1 INTRODUCTION

Biological membranes play a fundamental role in the complex reactions which take place in cells of living organisms. The importance of this role has been considered in two different types of formalisms: Membrane Systems and Brane Calculi (Cardelli 2004).

Membrane Systems - also called P systems - (Păun 2002, Păun 2000) were introduced by Gh. Păun as a class of distributed parallel computing devices of a biochemical type. The definition of such models starts from the observation that any biological system is a complex hierarchical structure, with a flow of materials and information that underlies their functioning. Since the original definition of Membrane Systems, several variants have been proposed and investigated. The emphasis is mainly on the computational properties of the model, and it makes use of automata, languages, and complexity theoretic tools.

Regarding process calculi, the notions of membranes and compartments have been explicitly represented, e.g., in Brane Calculi and BioAmbients (Regev, A., et al. 2004). In particular, Brane Calculi are a family of process calculi based on a set of biologically inspired primitives of membrane interaction.

The main novelty of Brane Calculi w.r.t. previously defined calculi and membrane systems consists in the fact that the active entities reside on membranes, and not inside membranes. In Brane Calculi, as well as in other biologically inspired process calculi, the emphasis is mainly on the fidelity to the biological reality, and the main interest is in the systems biology area.

A cross fertilization of the two research areas has recently started, as witnessed by some works relating Membrane Systems and Brane Calculi.

The aim of this paper is to compare the expressive comfort of the two formalisms for the modeling and the analysis of biological processes. In particular, we provide a representation of the LDL Cholesterol Degradation Pathway in both Brane Calculi and P systems.

The paper is organized as follows: in Section 2 we recall the syntax and the semantics of the Full Brane Calculus, as well as some recently developed analysis techniques. The basic definitions of the class of Membrane Systems used in the present work are given in Section 3. A description of the LDL Degradation Pathway is provided in Section 4. The representations of such a pathway in Brane Calculus and in Membrane Systems are provided in Sections 5 and 6, respectively. Finally, Section 7 reports some comparison fo the two representations, and some conclusive remarks are provided in Section 8.

2 FULL BRANE CALCULUS: SYNTAX AND SEMANTICS

In this Section we recall the syntax and the semantics of the Full Brane Calculus (Cardelli 2004). A system consists of nested membranes, and a process is associated to each membrane. Besides containing other membranes, a membrane can also contain some (small) molecules. As done in (Cardelli 2004), we assume that small molecules do not change, do not have internal structure, and do not interact among themselves.

Definition 1 *Let Mol be an infinite set of names for molecules, ranged over by m, m',.... The set of systems is defined by the following grammar:*

$$P,Q \quad ::= \quad \diamond \mid P \circ Q \mid !P \mid \sigma(\!|P|\!) \mid m$$

The set of (finite) multisets of molecules is defined by the following grammar:

$$p,q \quad ::= \quad \diamond \mid p \circ q \mid m$$

The set of brane processes is defined by the following grammar:

$$\sigma,\tau \quad ::= \quad 0 \mid \sigma|\tau \mid !\sigma \mid a.\sigma$$

Variables a,b range over actions.

The term $\diamond$ represents the empty system; the parallel composition operator on systems is $\circ$. The replication operator ! denotes the parallel composition of an unbounded number of instances of a system. The term $\sigma(\!|P|\!)$ denotes the brane that performs process σ and contains system P. The term m represents a single molecule.

Multisets of molecules will be used used below to define the operation of interaction between membranes and molecules.

The term 0 denotes the empty process, whereas | is the parallel composition of processes; with $!\sigma$ we denote the parallel composition of an unbounded number of instances of process σ. Term $a.\sigma$ is a guarded process: after performing the action a, the process behaves as σ.

We adopt the following abbreviations: with a we denote $a.0$, with $(\!|P|\!)$ we denote $0(\!|P|\!)$, and with $\sigma(\!|\,|\!)$ we denote $\sigma(\!|\diamond|\!)$.

The structural congruence relation on systems and processes is defined as follows (with abuse of notation we use $\equiv$ to denote both structural congruence on systems and structural congruence on processes):

Definition 2 *The structural congruence $\equiv$ is the least congruence relation satisfying the following axioms:*

$$\begin{array}{ll}
P \circ Q \equiv Q \circ P & \sigma \mid \tau \equiv \tau \mid \sigma \\
P \circ (Q \circ R) \equiv (P \circ Q) \circ R & \sigma \mid (\tau \mid \rho) \equiv (\sigma \mid \tau) \mid \rho \\
P \circ \diamond \equiv P & \sigma \mid 0 \equiv \sigma \\
 & \\
!\diamond \equiv \diamond & !0 \equiv 0 \\
!(P \circ Q) \equiv !P \circ !Q & !(\sigma \mid \tau) \equiv !\sigma \mid !\tau \\
!!P \equiv !P & !!\sigma \equiv !\sigma \\
P \circ !P \equiv !P & \sigma \mid !\sigma \equiv !\sigma \\
 & \\
0(\!|\diamond|\!) \equiv \diamond &
\end{array}$$

Note that the set of multisets of a molecules is a subset of the set of systems; hence, the first three structural congruence axioms for systems (i.e., the axioms for commutative monoids) also hold for multisets of molecules.

Definition 3 *The basic reaction rules are the following:*

$$(\texttt{par}) \quad \frac{P \rightarrow Q}{P \circ R \rightarrow Q \circ R}$$

$$(\texttt{brane}) \quad \frac{P \rightarrow Q}{\sigma(\!|P|\!) \rightarrow \sigma(\!|Q|\!)}$$

$$(\texttt{strucong}) \quad \frac{P' \equiv P \qquad P \rightarrow Q \qquad Q \equiv Q'}{P' \rightarrow Q'}$$

Rules `(par)` and `(brane)` are the contextual rules that respectively permit to a system to execute also if it is in parallel with another process or if it is inside a membrane, respectively. Rule `(strucong)` ensures that two structurally congruent systems have the same reactions.

With $\rightarrow^*$ we denote the reflexive and transitive closure of a relation $\rightarrow$. Given a reduction relation $\rightarrow$, we say that a system P has a *divergent computation* (or infinite computation) if there exists an infinite sequence of systems $P_0, P_1, \ldots, P_i, \ldots$ such that $P = P_0$ and $\forall i \geq 0 : P_i \rightarrow P_{i+1}$.

2.1 The Actions of Brane Calculi

The set of actions introduced in (Cardelli 2004) comprises both operations representing membranes interactions and operations for interactions between molecules and membranes.

The actions for membrane interactions can be classified into two groups. The first group contains actions inspired by endocytosis/exocytosis. Endocytosis is the process of incorporating external material into a cell by "engulfing" it with the cell membrane, while exocytosis is the reverse process. As endocytosis can engulf an arbitrary amount of material, giving rise to an uncontrollable process, in (Cardelli 2004) two more basic operations are used: *phagocytosis*, engulfing just one external membrane, and *pinocytosis*, engulfing zero external membranes.

The second group of actions is inspired by membrane fusion and splitting. To make membrane splitting more controllable, in (Cardelli 2004) two more basic operations are used: *budding*, consisting in splitting off one internal membrane, and *dripping*, consisting in splitting off zero internal membranes. Membrane fusion, or merging, is called *mating*.

Regarding the interaction between molecules and membranes, (Cardelli 2004) observes that membranes contain catalysts that can cause molecules, floating respectively inside and outside the membrane, to interact with each other without crossing the membrane. Membranes can bind molecules on either sides of their surface, and can release molecules on either sides of their surface. Usually, coordinated bindings and releases happen completely or not at all. Hence, the ability of a membrane to bind and release multiple molecules simultaneously is represented by a single *bind&release* operation.

Definition 4 *Let Name be a denumerable set of names, ranged over by n,m,…. The set of actions of the Full Brane Calculus is defined by the following grammar:*

$$\begin{array}{rcl} a & ::= & \circlearrowleft_n \mid \circlearrowleft_n^{\perp}(\sigma) \mid \circlearrowright_n \mid \circlearrowright_n^{\perp} \mid \circledcirc(\sigma) \\ & & mate_n \mid mate_n^{\perp} \mid bud_n \mid bud_n^{\perp}(\sigma) \mid drip(\sigma) \\ & & p(q) \rightrightarrows p'(q') \end{array}$$

Action $\circlearrowleft_n$ denotes phagocytosis; the co-action $\circlearrowleft_n^{\perp}$ is meant to synchronize with $\circlearrowleft_n$; names n are used to pair-up related actions and co-actions. The co-phago action is equipped with a process σ, this process will be associated to the new membrane that engulfs the external membrane. Action $\circlearrowright_n$ denotes exocytosis, and synchronizes with the co-action $\circlearrowright_n^{\perp}$. Exocytosis causes an irreversible mixing of membranes. Action $\circledcirc$ denotes pinocytosis. The pino action is equipped with a process σ: this process will be associated to the new membrane, that is created inside the brane performing the pino action.

Actions $mate_n$ and $mate_n^{\perp}$ will synchronize to obtain membrane fusion. Action bud_n permits to split off one internal membrane, and synchronizes with the co-action $bud_n^{\perp}$. Action *drip* permits to split off zero internal membranes. Actions $bud^{\perp}$ and *drip* are equipped with a process σ, that will be associated to the new membrane created by the brane performing the action.

The action $p(q) \rightrightarrows p'(q')$ binds, in general, the multiset p of molecules outside the membrane and the multiset q of molecules inside the membrane; if that is possible, it instantly releases the multiset p' of molecules outside the membrane and the multiset q' of molecules inside the membrane.

Definition 5 *The reaction relation for the Full Brane Calculus is the least relation containing the axioms in Table 1, and satisfying the rules in Definition 3.*

2.2 Decidability of Properties in Full Brane Calculus

In (Busi and Gorrieri 2006) we showed that the existence of a divergent computation is a decidable property for the MBD basic Brane Calculus (without molecules). This result exploits the theory of well-structured transition systems developed in (Finkel and Schnoebelen 2001).

In (Busi 2006) we extended this result to a broader class of properties and systems. Regarding the set of decidability properties, besides providing a constructive method for deciding divergence, the theory of well-structured transition systems (Finkel and Schnoebelen 2001) also provides methods for deciding other behavioural properties, such as, e.g., control state maintainability, inevitability and boundedness. The theory can be applied to finitely branching transition systems, provided that the set of states can be equipped with a well-quasi-ordering $\preceq$, i.e., a quasi-ordering relation which is compatible with the transition relation and such that each infinite sequence of states admits an increasing subsequence.

Table 1: The Set of Axioms of the Reduction Rule for the Full Brane Calculus.

`(phago)`	$\circlearrowleft_n.\sigma\vert\sigma_0\llparenthesis P\rrparenthesis \circ \circlearrowleft_n^{\perp}(\rho).\tau\vert\tau_0\llparenthesis Q\rrparenthesis \rightarrow \tau\vert\tau_0\llparenthesis \rho\llparenthesis\sigma\vert\sigma_0\llparenthesis P\rrparenthesis\rrparenthesis \circ Q\rrparenthesis$
`(exo)`	$\circlearrowright_n^{\perp}.\tau\vert\tau_0\llparenthesis \circlearrowright_n.\sigma\vert\sigma_0\llparenthesis P\rrparenthesis \circ Q\rrparenthesis \rightarrow P \circ \sigma\vert\sigma_0\vert\tau\vert\tau_0\llparenthesis Q\rrparenthesis$
`(pino)`	$\circledcirc(\rho).\sigma\vert\sigma_0\llparenthesis P\rrparenthesis \rightarrow \sigma\vert\sigma_0\llparenthesis \rho\llparenthesis\,\rrparenthesis \circ P\rrparenthesis$
`(mate)`	$mate_n.\sigma\vert\sigma_0\llparenthesis P\rrparenthesis \circ mate_n^{\perp}.\tau\vert\tau_0\llparenthesis Q\rrparenthesis \rightarrow \sigma\vert\sigma_0\vert\tau\vert\tau_0\llparenthesis P \circ Q\rrparenthesis$
`(bud)`	$bud_n^{\perp}(\rho).\tau\vert\tau_0\llparenthesis bud_n.\sigma\vert\sigma_0\llparenthesis P\rrparenthesis \circ Q\rrparenthesis \rightarrow \rho\llparenthesis\sigma\vert\sigma_0\llparenthesis P\rrparenthesis\rrparenthesis \circ \tau\vert\tau_0\llparenthesis Q\rrparenthesis$
`(drip)`	$drip(\rho).\sigma\vert\sigma_0\llparenthesis P\rrparenthesis \rightarrow \rho\llparenthesis\,\rrparenthesis \circ \sigma\vert\sigma_0\llparenthesis P\rrparenthesis$
`(B&R)`	$p \circ p(q) \rightrightarrows p'(q').\sigma\vert\sigma_0\llparenthesis q \circ P\rrparenthesis \rightarrow p' \circ \sigma\vert\sigma_0\llparenthesis q' \circ P\rrparenthesis$

The *control state maintainability problem* consists in checking the following property: given an initial state s and a finite set $X = \{s_1, \ldots, s_n\}$ of states, there exists a computation, starting from s, where all states cover one of the s_i (i.e., for all states s' reachable during the computation, there exists $i \in \{1, \ldots, n\}$ such that $s_i \preceq s'$).

The *inevitability problem* is the dual problem of the control state maintainability problem, and consists in checking if all computations starting from an initial state s eventually visit a state not covering one of the s_i.

The *boundedness problem* consists in checking if the set of states reachable from an initial state s is finite.

In (Busi 2006) we showed that the aforementioned properties are decidable on the full brane calculus without the *phago* operation.

As the reaction relation of Brane Calculus is not finitely branching, an alternative, behaviourally equivalent semantics, based on a notion of normal forms is provided, and the well-quasi ordering is defined on systems in normal forms. See (Busi 2006) for the technical details.

2.3 Recursive Definitions in Brane Calculus

In Brane Calculus systems (and processes) with an infinite behaviour are defined by using the replication operator. An alternative, sometimes more expressive way of for defining systems with an infinite behaviour in process calculi consists in using (mutually) recursive definitions for systems (see, e.g., Busi, Gabbrielli and Zavattaro 2003 for a comparison of replication and recursive definitions for CCS Milner 1989 and π-calculus Milner, Parrow and Walker 1992).

The syntax and the semantics of recursive definitions is presented below. We assume two sets of constants, ranged over by D (resp. d), to denote recursively defined systems (resp. processes).

The class of systems is defined by adding the productions $P ::= D$ and $\sigma ::= d$ to the grammar in Definition 1. We assume that each constant D (resp. d) has a unique defining equation of the form $D = P$ (resp. $d = \sigma$) where P may contain D as a subsystem (resp. σ may contain d as a subprocess).

The semantics is defined by adding the unfolding axioms $P \equiv D$ and $\sigma \equiv d$ to the structural congruence axioms in Definition 2.

For example, the system $dpino(\!|\ |\!)$, where the process $dpino$ is recursively defined as $dpino = \circledcirc(dpino)$, evolves to a sequence of systems with a growing number of nested membranes.

Replication can be defined by means of recursive definitions: for example, the system $!P$ can be defined as $BangP$, with the defining equation $BangP = P \circ BangP$. On the other hand, the characterization of a class of recursively defined systems, for which there exists a behaviourally equivalent system in Brane Calculus with replication, deserves further investigation.

In some cases, recursive definitions have been revealed useful to provide a more intuitive modeling of real systems.

3 MEMBRANE SYSTEMS

As previously said, *Membrane Systems* (also called *P Systems*) are a class of *distributed* and *parallel* computing devices of a biochemical inspiration.

Such systems are based upon the notion of *membrane structure,* which is a structure composed by several cell-membranes, hierarchically embedded in a main membrane called the *skin membrane*. Usually, this structure is represented by means of a Venn diagram, where all the sets are subsets of a unique set and they are not allowed to be intersected (that is, two sets are either one the subset of the other, or they are disjoint). The membranes define *regions;* in particular, a region is defined by a membrane and by the membranes immediately inside it.

We associate with each region a set of *objects*, described by some symbols over an alphabet, and a set of *evolution rules*. The objects evolve according to the evolution rules, which can modify the objects to obtain new objects, and can move objects through the membranes to reach different regions. The evolution rules are applied in a non-deterministic maximally parallel manner: at each step, all the objects which can evolve should evolve. In other words, all objects which can be rewritten by means of an evolution rule, have to be used by such rule. If different sets of rules can be applied, then one of them is non-deterministically chosen to be applied. At each step of computation, all applicable rules are applied at the same time, in parallel, on all objects which can evolve.

The evolution rules can modify the membrane structure too, by *dissolving* the membranes. When a membrane is dissolved, all the objects in this membrane remain free in the membrane placed immediately outside, while the evolution rules of the dissolved membrane are lost. The skin membrane is never dissolved.

A computation device is obtained: starting from an initial configuration, with a certain number of objects in certain membranes, we let the system evolve. If a computation *halts,* that is no further evolution rule can be applied, the result of the computation is defined to be the set of objects in a specified membrane (or expelled through the skin membrane). If a computation never halts (i.e., one or more object can be rewritten forever), then it provides no output.

We stress the fact that membrane systems were initially introduced to explore the computational nature of the cell and of various features of membranes. For this reason, the model was defined abstracting a number of principles underlying the functioning of biological membranes. It was only later that, exploiting the bio-inspired aspects of the model, it has been applied to the investigation and modeling of different biological processes, especially involving membranes.

An up-to-date bibliography of the Membrane Systems area and other useful resources can be found at the following web address:

`<http://psystems.disco.unimib.it>`

Various kind of evolution rules can be considered within membrane systems, which formalize different biological processes and features. In this paper, we will restrict our attention to systems where objects are modified by means of rewriting rules, bind rules and release rules, and where objects are moved through the membrane structure by means of symport and antiport rules.

To slightly ease the notation, when a symbol $z \in V$ is repeated k times consecutively in a string, we will also use the shorter form z^k. For example, the string $aaabbbbbcc$ will be also written as $a^3b^5c^2$.

Formally, a membrane system (of degree $d, d \geq 1$) is a construct $\Pi = (V, \mu, M_1, \ldots, M_n, R_1, \ldots, R_n, i_0)$ where:

- V is an alphabet.
- μ is a membrane structure consisting of n membranes, with the membranes and the regions labelled in a one-to one manner with elements of a given set; we will use the labels $1, 2, \ldots, n$.
- $M_i, 1 \leq i \leq n$ are finite languages representing structured objects over V associated with the regions $1, 2, \ldots, n$ of μ.
- $R_i, 1 \leq i \leq n$ are finite sets of evolution rules associated with the regions $1, 2, \ldots, n$ of μ;
- $i_0 \in \{1, 2, \ldots n\} \cup \{\infty\}$. If i_0 is a number between 1 and n then it specifies the *output membrane* of Π; if $i_0 = \infty$ or if it is omitted, then the output is read in the outer region.

Evolution rules can be of the following forms:

- Rewriting rules: $u \to v$, where $u \in V^+$ and $v = v^{'}$ or $v = v^{'}\delta$, where $v^{'}$ is a string over V and δ is a special symbols not in V. One or more symbols in a string are replaced with a string over V. When a rule containing the symbol δ is applied, then the membrane which delimits externally the region where the rule is applied is dissolved. All objects remain free in the region immediately outside, while all evolution and communication rules are lost.
- Bind rules: $u, v \to u\&v$ where $u, v \in V^+ \cup \{\&\}$, and $\&$ is a special symbol not in V. Two strings are bound together.
- Release rules: $u\&v \to u, v$ where $u, v \in V^+ \cup \{\&\}$, and $\&$ is a special symbol not in V. Two bound strings are separated.
- Antiport rules: $u[_i v]_i \to v[_i u]_i$, where $u, v \in V^*$.
  A string v is moved from the region i to the region immediately outside. At the same time, a string u is moved in the opposite direction. A particular case is obtained when one of the strings u or v (but, of course, not both) is empty, as this indicates the movement of an object alone. In this case, such a rule is called uniport rule.
- Symport rules: the form of these rules can be either $[_i u, v]_i \to u, v[_i \ \ ]_i$ or $u, v[_i \ \ ]_i \to [_i \ u, v \ ]_i$, for $u, v \in V^+$.
  In the first case, two strings are moved together from region i to the region immediately outside. In the second case, the strings, taken from the region immediately outside region i, are brought to region i.

The $(n+1)$-tuple $(\mu, M_1, M_2, ..., M_n)$ constitutes the *initial configuration* of the system. In general, any sequence $(\mu^{'}, M^{'}_{i_1}, ..., M^{'}_{i_k})$ with $\mu^{'}$ a membrane structure obtained by removing from μ all membranes different from $i_1, \ldots, i_k$ (of course the skin membrane is not removed), with $M^{'}_j$ set of strings over V, $1 \leq j \leq k$, and $\{i_1, \ldots, i_k\} \subseteq \{1, 2, \ldots, n\}$, is called a *configuration* of Π.

It should be noted that the membranes preserve the initial labelling in all subsequent configurations, thus the correspondence between membranes, multisets of objects and sets of evolution rules is well specified by the subscript of these elements.

For two configurations

$$C_1 = (\mu^{'}, M^{'}_{i_1}, ..., M^{'}_{i_k}), C_2 = (\mu^{''}, M^{''}_{j_1}, ..., M^{''}_{j_k})$$

of Π, we say that we have a *transition* $C_1 \Rightarrow C_2$ if we can pass from C_1 to C_2 by applying the evolution rules appearing in $R_{i_1}, \ldots, R_{i_k}$ in a non-deterministic maximal parallel manner. That is: each object which can evolve must evolve, and all objects evolve at the same time, in parallel. Of course, starting from the same configuration, it could be possible to have different choices for the maximal set of rules to be applied. In such a case, one choice among all possible is non-deterministically applied.

Hence, starting from the initial configuration C_0 of our system, we get a sequence of transitions. Such a sequence of transitions in a P system Π is called a *computation* with respect to Π.

We say a computation $C_0 \Rightarrow C_1 \Rightarrow \ldots \Rightarrow C_m, \ m \geq 0$ is *successful* if and only if there is no rule in C_m which can be applied to the objects present in C_m, i.e., the computation *halts*. On the contrary, the computation is *unsuccessful* if it can continue forever, that is, there exists a configuration C_{m+1} such that $C_m \Rightarrow C_{m+1}$. Note that it is not necessary to have $C_m \neq C_{m+1}$.

We stress the fact that, due to the non-deterministic nature of the system, different evolutions are possible starting from the same initial configuration. We obtain, in fact, a tree of computations. The outputs of the system are given by all possible halting evolutions which can be reached from the initial configuration.

A P system Π can thus be seen as a computing device which nature is distinguished on the basis of the output membrane. If $i_1 \in \{1, \ldots, n\}$ then the system Π works in *internal mode*. The result of a computation is given by the objects in the output membrane when the computation halts.

Otherwise the system works in *external mode:* we observe the system from outside and we collect the objects expelled through the skin membrane.

By $L(\Pi)$ we denote the language generated by Π in the way just described.

Finally, we wish to point out that the above definitions concern the basic variant of Membrane Systems, mainly considering the computational aspects of them. Nonetheless, it should be noted that when such systems are used to simulate biological processes, some definitions could be misleading or inappropriate. For instance, the maximal parallel application of the rules at each step could not be realistic when considering certain biological processes. Another important difference usually stressed when Membrane Systems are applied to the simulation of biological processes concerns the computation of a system. In fact, when a biological process is simulated, one is interested in the states of the system during the whole transition (halting or not), and in collecting various information on such states: the final result of the process is not, in general, the only information one is interested in. For this reason, when Membrane Systems are applied to the study of biological processes, non-halting sequence of transition are usually considered, and the term *evolution* of the system is used instead of *computation* of the systems, in order to underline this difference.

4 EXAMPLE: LDL CHOLESTEROL DEGRADATION

In the following we model and analyse the endocytic pathway for internalizing low-density lipoprotein (LDL) (Lodish et al. 1999) — see Figure 1.

LDL is one of several complexes that carry cholesterol through the bloodstream. A LDL particle has an outer phospholipid shell containing a single molecule of a large protein known as *apoB-100*; the core of a particle is packed with cholesterol in the form of cholesteryl esters. Most mammalian cells produce receptors that bind to apoB-100 and subsume LDL particles by receptor-mediated endocytosis. By this mechanism cells acquire from the bloodstream the cholesterol required for the membrane synthesis that occurs during cell growth.

After endocytosis, the LDL particles are transported to lysosomes via the endocytic pathway and then are degraded by lyosomal hydrolases. The LDL receptors, which dissociate from their ligands in the late endosome, recycle to the cell surface.

Individuals with the *familiar hypercholesterolemia* disorder usually have developed a mutant form of the LDL receptor, causing high serum level of cholesterol and increasing the risk of cardiovascular diseases. It could either happen that such mutant receptors do not bind LDL, or that the receptors bind LDL normally, but the LDL-receptor complex cannot be internalized by the cell.

5 MODELING THE LDL DEGRADATION PATHWAY IN BRANE CALCULUS

We show how to model the LDL degradation pathway in Brane Calculus. We start with a modeling that makes use

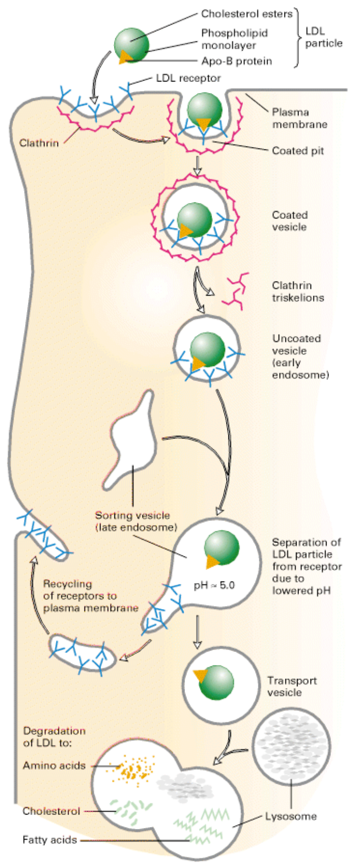

Figure 1: Endocytic Pathway for Internalizing Low-Density Lipoprotein (Lodish et al. 1999).

of recursive definitions, as it turns out to be more faithful to the biological reality. Then, to perform an analysis of its behaviour, we move to an equivalent model that does not make use of recursive definitions.

The modeling with recursive definitions is reported in Table 2. In such a modeling, we perform some abstraction: for example, the clathrin molecules are not explicitly represented. A system is composed by a molecule of LDL, denoted by *LDLpart*, and a cell, containing a *ldlReceptor* process on its surface and an unbounded number of copies of late endosomes and lysosomes (represented respectively by the *LateEndo* and *Lysosome* membranes). The *ldlReceptor* binds the *LDLpart* by performing a bind&release operation, then it creates an internal vesicle by pinocytosis. The process on this vesicle is *ldlRecPart*, representing the receptor-particle complex. Such a vesicle fuses with a late endosome *LateEndo* by performing the $mate_{le}$ action. The acidic pH in the late endosome, represented by the "virtual molecule" *acid*, triggers the bind&release action that releases the *LDLparticle*. At this point, a vesicle with process *recVesicle* – representing the receptor – is dripped. The late endosome fuses with a *Lysosome* by performing the $mate_{lyso}$ action. At this point, the enzymes *DestroyEnz* contained in the lysosome trigger the bind&release action that decompose the *LDLpart* in its constituent parts (*LDLconstituents*). Independently from these fusion and decomposition actions, the vesicle with process *recVesicle* performs an exocytosis, thus permitting to the *ldlReceptor* process to reach the cell surface.

Table 2: Modeling the LDL Cholesterol Degradation Pathway in Brane Calculus with Recursive Definitions.

$$\begin{array}{rl}
System_1 = & LDLpart \circ \\
& ldlReceptor(\!|\; RLateEndo \circ \\
& \qquad RLysosome \;|\!) \\
RLateEndo = & mate^{\perp}_{le}(\!|acid|\!) \circ RLateEndo \\
RLysosome = & mate^{\perp}_{lyso}(\!|DestroyEnz|\!) \circ \\
& RLysosome \\
ldlReceptor = & LDLpart(\diamond) \rightrightarrows \diamond(\diamond). \\
& \circledcirc(ldlRecPart).\circlearrowleft^{\perp}_{rec} \\
ldlRecPart = & mate_{le} \mid \\
& \diamond(acid) \rightrightarrows \diamond(LDLpart). \\
& drip(recVesicle).mate_{lyso} \mid \\
& \diamond(DestroyEnz \circ LDLpart) \rightrightarrows \\
& \diamond(LDLconstituents) \\
recVesicle = & \circlearrowleft_{rec}.ldlReceptor
\end{array}$$

The description of the system reported above contains recursive definitions (for the *ldlReceptor*), hence it cannot be analyzed by the techniques illustrated in Section 2.2. To perform some kind of analysis, in Table 3 we provide an alternative, behaviourally equivalent modelization that does not make use of recursive definitions. The basic idea is to model the receptor as a molecule (floating outside the cell) instead of a more realistic process on the cell surface. The main differences w.r.t. the previous modeling are the following: now the (replicated) process on the cell binds both an LDL particle (*LDLpart*) and a receptor (*LDLrec*) to start the process. Moreover, when the vesicle that will be used to carry the receptor to the cell surface is created, the process *recVesicle* releases a "receptor molecule" *LDLrec* inside the vesicle before performing the exocytosis. When the exocytosis is performed, the *LDLrec* molecule is expelled outside the cell and can be bound again by a new copy of the *bindRecPart* process.

The modeling in Table 3 is less intuitive than the modeling in Table 2, as the receptors are not directly connected to the cell. Hence, while the modeling without recursive definitions behaves correctly in case of a system composed by a single cell, in case of multiple cells it could happen that a single cell concurrently makes use of two or more receptors.

Table 3: Modeling the LDL Cholesterol Degradation Pathway in Brane Calculus without Recursive Definitions.

$$\begin{array}{rl}
System_2 = & LDLpart \circ LDLrec \circ \\
& (!bindRecPart)(\!|!LateEndo \circ \\
& \qquad !Lysosome|\!) \\
LateEndo = & mate^{\perp}_{le}(\!|acid|\!) \\
Lysosome = & mate^{\perp}_{lyso}(\!|DestroyEnz|\!) \\
bindRecpart = & LDLrec \circ LDLpart(\diamond) \rightrightarrows \diamond(\diamond). \\
& \circledcirc(ldlRecPart).\circlearrowleft^{\perp}_{rec} \\
ldlRecPart = & mate_{le} \mid \\
& \diamond(acid) \rightrightarrows \diamond(LDLpart). \\
& drip(recVesicle).mate_{lyso} \mid \\
& \diamond(DestroyEnz \circ LDLpart) \rightrightarrows \\
& \diamond(LDLconstituents) \\
recVesicle = & \diamond(\diamond) \rightrightarrows \diamond(LDLrec).\circlearrowleft_{rec}
\end{array}$$

By exploiting the techniques illustrated in Section 2.2 we can perform some analysis of the behaviour of the brane system $System_2$.

For example, by using the inevitability property it is possible to check that all the computations starting from $System_2$ will eventually reach a system not covering the system *LDLpart*; this means that the LDL particle will eventually be consumed by the cell.

To check that the recycling of the receptor behaves correctly, we could check that the above property (i.e., all computations will eventually reach a system not covering *LDLpart*) holds when starting from a system containing more than one LDL particle, such as, e.g., the following system containing three LDL particles:

$$\begin{array}{rl}
System_3 = & LDLpart \circ LDLpart \circ LDLpart \circ \\
& LDLrec \circ \\
& (!bindRecPart)(\!|!LateEndo \circ \\
& \qquad !Lysosome|\!)
\end{array}$$

Moreover, we can check that all computations of both $System_2$ and $System_3$ terminate (or, equivalently, that such systems have no divergent computation), whereas the system containing an unbounded number of LDL particles, i.e.,

$$\begin{array}{rl}
System_4 = & !LDLpart \circ LDLrec \circ \\
& (!bindRecPart)(\!|!LateEndo \circ \\
& \qquad !Lysosome|\!)
\end{array}$$

does not terminate.

The control state maintainability property can be used, e.g., to check that, in all reachable states, no more than a single receptor is contained inside the cell.

We can check what happens in individuals that develop a mutant LDL receptor that is not able to bind an LDL particle. We need to modify the system in Table 3 by replacing the *bindRecpart* process with the following:

$$bindRecpart = LDLwrongr \circ LDLpart(\diamond) \rightrightarrows \diamond(\diamond).\\ \circledcirc(ldlRecPart).\circlearrowleft^{\perp}_{rec}$$

In such a case, it is possible to show that the inevitability property discussed above no longer holds; namely, there are computations starting from this new version of $System_2$ that never reach a system not covering the system *LDLpart*; this means that the LDL particles will not be consumed.

The techniques developed in (Busi 2006), and mentioned in Section 2.2, are based on the so-called *tree saturation methods* for well-structured transition systems: such a class of methods essentially consists in representing (an approximation of) all the computations in a finite tree-like structure. Another class of methods, called *set saturation methods*, can be used to decide other properties, such as coverability. Given a state x and an initial state s, the coverability problem consists in checking if there exists a computation starting from s that reaches a state covering x. We plan to exploit the set saturation methods to investigate the decidability of the coverability problem. In such a case, we could verify, e.g., if the LDL particle will eventually float in the cytosol by checking the coverability of system $(\!|LDLpart|\!)$.

6 MODELING THE LDL DEGRADATION PATHWAY IN MEMBRANE SYSTEMS

We show now how the process described in Section 4 can be modeled using Membrane Systems.

Consider the system

$$\Pi = (V, \mu, M_0, \ldots, M_3, R_1, \ldots, R_3, i_0)$$

where:

- $V = \{S, LDLpart, LDLRcpt, Clth, LateEndo, Chol\}$
- $\mu = [_0[_1[_2[_3\ ]_3]_2]_1]_0$
- $M_0 = \{LDLpart\}$
- $M_1 = \{LDLrcpt\}$
- $M_2 = \{Clht, LateEndo\}$
- $M_3 = \emptyset$
- $R_0 = \{LDLpart + LDLrcpt \rightarrow LDLpart\&LDLrcpt\}$
- $R_1 = \{[_1LDLRcpt]_1 \rightarrow LDLRcpt[_1\ ]_1\} \cup \{LDLpart\&LDLRcpt[_1\ ]_1 \rightarrow [_1LDLpart\&LDLRcpt]_1\} \cup \{LDLRcpt[_1\ ]_1 \rightarrow [_1LDLRcpt]_1\}$
- $R_2 = \{[_2Clth]_2 \rightarrow Clth[_2\ ]_2\} \cup \{Clth \rightarrow Clth\} \cup \{Clth, LDLpart\&LDLRcpt[_2\ ]_2 \rightarrow [_2Clth, LDLpart\&LDLRcpt]_2\} \cup \{Clth, LDLRcpt[_2\ ]_2 \rightarrow [_2Clth, LDLRcpt]_2\} \cup \{LateEndo + LDLpart \rightarrow LateEndo\&LDLpart\} \cup \{LDLpart\&LDLrcpt \rightarrow LDLpart + LDLrcpt\} \cup \{[_2LDLrcpt]_2 \rightarrow LDLrcpt[_2\ ]_2\}$
- $R_3 = \{LateEndo\&LDLpart[_3\ ]_3 \rightarrow [_3\ LateEndo\&LDLpart\ ]_3\} \cup \{LateEndo\&LDLpart \rightarrow LateEndo + LDLpart\} \cup \{[_3LateEndo]_3 \rightarrow LateEndo[_3\ ]_3\}$
- $i_0 = 3$

The system works in the following way.

Initially, the molecule *LDLpart* is present in region 0. We assume that from time to time a molecule of this type can enter from the region outside the skin membrane, so that the process can be iterated.

The rule $[_1LDLRcpt]_1 \rightarrow LDLRcpt[_1\ ]_1$ moves LDL receptors from region 1 to region 0. Here, the receptor is eventually bound to *LDLpart* by means of the rule $LDLpart + LDLrcpt \rightarrow LDLpart\&LDLrcpt$ and then the complex is moved back again in region 1, by the rule $\{LDLpart\&LDLRcpt[_1\ ]_1 \rightarrow [_1LDLpart\&LDLRcpt]_1\}$.

If *LDLpart* has not yet appeared in region 0, then the receptor is communicated back to region 1.

At the same time, the clathrin molecules *Clth* in region 2 are rewritten again to *Clth* until they are sent to region 1 by $[_2Clth]_2 \rightarrow Clth[_2\ ]_2$. Here, we have two cases.

Case 1. If *LDLRcpt* is not bound to *LDLpart*, then the rule $Clth, LDLRcpt[_2\ ]_2 \rightarrow [_2Clth, LDLRcpt]_2$ is applied. Then, *Clth* is rewritten again to itself until eventually it will be sent again to region 1, and *LDLRcpt* is immediately sent back to region 1 by the rule $[_2LDLrcpt]_2 \rightarrow LDLrcpt[_2\ ]_2$. The initial situation is restored and the process evolves in the same way.

Case 2. If *LDLRcpt* is bound to *LDLpart*, then the rule $Clth, LDLRcpt\&LDLpart[_2\ ]_2 \rightarrow [_2Clth, LDLRcpt\&LDLpart]_2$ is applied, and the two molecules are brought to region 2.

Here, the clathrin molecules start immediately another process of rewriting and then will be sometimes moved again to region 1 to import another molecule of LDL.

The complex *LDLRcpt&LDLpart* enters the process to release cholesterol. In two steps, by means of the bind rule $LateEndo + LDLpart \rightarrow LateEndo\&LDLpart$ and of the release rule $LDLpart\&LDLrcpt \rightarrow LDLpart + LDLrcpt$ the complex *LateEndo&LDLpart* is created and the LDL receptors are left free.

LDL receptors are sent back to region 1 using the rule $[_2LDLrcpt]_2 \rightarrow LDLrcpt[_2\ ]_2$, while the complex *LateEndo&LDLpart* is sent to (lysosome) membrane 3. Here, the complex is separated by the release rule $LateEndo\&LDLpart \rightarrow LateEndo + LDLpart$. LDL is then transformed in cholesterol ($LDLpart \rightarrow Chol$), while *LateEndo* is sent back in region 2.

The system is ready again to start to import a new *LDLpart* molecule, as soon as such a molecule will be present in region 0, so that the evolution of the system can continue.

The model presented here was just intended to give some hints on how a biological process could be modeled within the framework of Membrane Systems. Clearly, a model which has to be then simulated in silico to gain information on the functioning of the process in various situations (e.g., diseases), would require a model considering also many other aspects, such as, e.g., the rates of reaction. Various simulators of Membrane Systems have been developed. Some of them have been developed to be applied to the simulation of biological processes, such as (Pescini, Besozzi and Mauri 2005).

The interested reader can find other simulations of biological processes with membrane systems, for example in (Ardelean and Besozzi 2005) and (Ciobanu, Păun and Perez-Jimenez 2005).

7 DISCUSSION

A comparison of the representations of the LDL cholesterol degradation pathway in Brane Calculus and in Membrane Systems highlights the following aspects. While the explicit modeling of membranes creation and interaction in Brane Calculus permits to provide a representation of the behaviour of vesicles that is faithful to the biological reality, in Membrane Systems it is more difficult to dynamically change the hierarchical membranes structure, as the only available operation on the membrane structure is dissolution. For this reason, in Membrane Systems we cannot provide an explicit representation of vesicles as membranes; hence, either we provide a representation of the vesicle as a membrane in the initial structure of the system, or we use a more abstract representation of vesicles as objects. More precisely, in Brane Calculus the creation of the vesicle containing the LDL particle bound to the LDL receptor is obtained by a *pino* operation, whereas in Membrane Systems such a vesicle is represented by the membrane with label 1, that is always present in the system.

Another key difference is concerned with the fact that Brane Calculus processes provide an abstraction for entities residing on the membrane, while such an abstraction is not supported in Membrane Systems. Thus, in Brane Calculus the LDL receptor is modeled as a process that binds an LDL particle and produces a vesicle – by pinocytosis – decorated with a receptor-particle complex. On the other hand, in Membrane Systems membranes only act as separators between regions, and are not decorated with active entities. Hence, the LDL receptor is represented as an object, initially residing in membrane 1, that at each computational step exits membrane 1, possibly binds to an LDL particle outside membrane 1, and enters again membrane 1.

In Brane Calculus, the recycling of the LDL receptors to the external cell surface is performed by dripping a vesicle – decorated with the LDL receptor – that performs an exocytosis. On the other hand, in Membrane Systems this operation is performed by moving the LDL receptor object from membrane 2 (representing the late endosome) to membrane 1.

We note that the rewriting rules of Membrane Systems permit to obtain an intuitive representation of the chemical reactions and the complexation/decomplexation operations between objects, while in Brane Calculus these reactions need to be simulated with the bind&release operation.

In the comparison carried out in this Section we considered Brane Calculus with recursive definitions. Note that, when moving to Brane Calculus with no recursive definitions, the representation of receptors as molecules is not very intuitive, and can lead to problems when systems with multiple cells are analysed.

8 CONCLUSIONS

In this paper we show how to model the LDL cholesterol degradation pathway in Brane Calculus and in Membrane Systems. We also provided a comparison of the two approaches.

Another paper investigating such a pathway is (Pilegaard, Nielson and Nielson 2005), where the LDL degradation pathway is modeled in BioAmbients (Regev, A., et al. 2004) and analyzed by static analysis techniques. A first look at these modelings shows that the representations in Brane Calculus and in Membrane Systems are more intuitively appealing than the representation in BioAmbients. This is mainly due to the fact that interaction in BioAmbients is mainly based on channels, and channels do not have a clear biological counterpart.

Regarding the automatic verification of systems, our research group is developing a tool for Brane Calculus, consisting of an animator and of an analyser based on the analysis techniques illustrated in Section 2.2.

ACKNOWLEDGMENTS

We thank he anonymous referees for their comments and suggestions.

REFERENCES

Ardelean, I.I., and D. Besozzi. 2005. On modelling ion fluxes across biological membranes with P systems. In Proceedings of the Third Brainstorming Week on Membrane Computing, Sevilla (Spain), 35-42.

Busi, N. 2006. Deciding behavioural properties in Brane Calculi. In Proceedings Computational Methods in System Biology 2006 (CMSB 2006), to appear.

Busi, N., M. Gabbrielli and G. Zavattaro. 2003. Replication vs. recursive definitions in channel based calculi. In Proceedings of 30th International Colloquium on Automata, Languages and Programming (ICALP'03), LNCS 2719, 133–144, Springer-Verlag.

Busi, N., and R. Gorrieri. 2006. On the computational power of Brane Calculi. *Transactions on Computational Systems Biology*, LNCS, Springer, to appear.

Cardelli, L. 2004. L. Cardelli. Brane Calculi - Interactions of biological membranes. In Proceedings Computational Methods in System Biology 2004 (CMSB 2004), LNCS 3082, Springer, 2005.

Ciobanu, G., Gh. Păun and M.J. Perez-Jimenez, eds. 2005. Applications of membrane computing. Springer-Verlag, Berlin, 43–80.

Finkel, A. and Ph. Schnoebelen. 2001. Well-structured transition systems everywhere! *Theoretical Computer Science*, 256:63–92, Elsevier.

Lodish, H., A. Berk, P. Matsudaira, C. A. Kaiser, M. Krieger, M. P. Scott, S. L. Zipursky and J. Darnell. 1999. Molecular cell biology. W.H. Freeman and Company, 4th edition.

Milner, R.. 1989. Communication and concurrency. Prentice-Hall.

Milner, R., J. Parrow and D. Walker, 1992. A calculus of mobile processes. *Journal of Information and Computation*, 100:1–77. Academic Press.

Păun, G.. 2002. Membrane computing. An introduction. Springer.

Păun, G.. 2000. Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143.

Pescini, D., D. Besozzi and G. Mauri. 2005. Investigating local evolutions in dynamical probabilistic P systems in G. Ciobanu, Gh. Păun, Pre-Proceedings of First International Workshop on Theory and Application of P Systems, Timisoara, Romania, 83-90

Pilegaard, H., F. Nielson and H. Riis Nielson. 2005. Static analysis of a model of the LDL degradation pathway. in Proceedings Computational Methods in System Biology 2005 (CMSB 2005).

Regev, A., E. M. Panina, W. Silverman, L. Cardelli and E. Shapiro. 2004. BioAmbients: an abstraction for biological compartments. *Theoretical Computer Science*, 325(1):141–167, Elsevier.

AUTHOR BIOGRAPHIES

NADIA BUSI got the PhD degree in Computer Science in 1998. From February 1998 to September 2001 she has been Researcher, and since October 2001 she is Associate Professor at the Computer Science Department of the University of Bologna, Italy. Her research interests cover the area of the theory of languages and models for the description of concurrent systems, in particular coordination languages, Petri nets and (biologically inspired) process calculi. Her e-mail address is `<busi@cs.unibo.it>`, and her web page is `<http://www.cs.unibo.it/~busi>`.

CLAUDIO ZANDRON got the PhD degree in Computer Science in 2002. From December 2002 he works as a researcher in Computer Science at the University of Milano-Bicocca. His research interests concern Formal Languages, Molecular Computing, and DNA Computing. His e-mail address is `<zandron@disco.unimib.it>`, and his web page is `<http://www.bio.disco.unimib.it/~zandron>`.