

# P Systems with Endosomes

Roberto Barbuti      Giulio Caravagna  
Andrea Maggiolo–Schettini      Paolo Milazzo

Dipartimento di Informatica, Università di Pisa  
Largo Pontecorvo 3, 56127 Pisa, Italy.  
Email: {*barbuti, caravagn, maggiolo, milazzo*}@di.unipi.it

April 15, 2009

## Abstract

P Systems are computing devices inspired by the structure and the functioning of a living cell. A P System consists of a hierarchy of membranes, each of them containing a multiset of objects, a set of evolution rules, and possibly other membranes. Evolution rules are applied to the objects of the same membrane with maximal parallelism. In this paper we present an extension of P Systems, called P Systems with Endosomes (PE Systems), in which endosomes can be explicitly modeled. We show that PE Systems are universal even if only the simplest form of evolution rules is considered, and we give one application examples.

## 1 Introduction

P Systems were introduced by Păun in [7] as distributed parallel computing devices inspired by the structure and the functioning of a living cell. A P System consists of a *hierarchy of membranes*, each of them containing a multiset of *objects*, representing molecules, a set of *evolution rules*, representing chemical reactions, and possibly other membranes. For each evolution rule there are two multisets of objects, describing the reactants and the products of the chemical reaction. A rule in a membrane can be applied only to objects in the same membrane. Some objects produced by the rule remain in the same membrane, others are sent *out* of the membrane, others are sent *into* the inner membranes, which are identified by their labels. Evolution rules are applied with *maximal parallelism*, meaning that it cannot happen that some evolution rule is not applied when the objects needed for its triggering are available.

Many variants and extensions of P Systems exist that include features to increase their expressiveness and that are based on different evolution strategies. Among the most common extensions we mention P Systems with dissolution rules that allow a membrane to disappear and release in the environment all

the objects it contains. We mention also P Systems with priorities, in which a priority relationship exists among the evolution rules of each membrane and can influence the applicability of such rules, and P Systems with promoters and inhibitors, in which the applicability of evolution rules depends on the presence of at least one occurrence and on the absence, respectively, of a specific object. See [2, 8] for the definition of these (and other) variants of P Systems and [13] for a complete list of references to the bibliography of P Systems.

In this paper we present another extension of P Systems, called P Systems with Endosomes (PE Systems), with these features:

- objects can be contained both inside and on the surfaces of the membranes (as in P Systems with peripheral proteins [4, 12]);
- rules are contained on the surfaces of the membranes (they can rewrite objects outside/on/into the membranes);
- endosomes can be explicitly created in order to model a biologically inspired transportation mechanism.

The definition of this extension of P Systems has a biological inspiration. In fact, the *endocytosis* of macromolecules is the process by which cells absorb material (molecules such as proteins) from outside the cell by engulfing it with their cell membrane. It is used by all cells of the body because most substances important to them are large polar molecules that cannot pass through the hydrophobic plasma membrane or cell membrane. There exist three kind of endocytosis: *phagocytosis*, *pinocytosis* and *receptor-mediated endocytosis*. In particular, phagocytosis (literally, cell-eating) is the process by which cells ingest large objects, such as cells which have undergone apoptosis, bacteria, or viruses. The membrane folds around the object, and the object is sealed off into a large vacuole known as a phagosome. Pinocytosis (literally, cell-drinking) is concerned with the uptake of solutes and single molecules such as proteins, and, finally, receptor-mediated endocytosis is a more specific active event where the cytoplasm membrane folds inward to form coated pits. These inward budding vesicles bud to form cytoplasmic vesicles. Figure 1<sup>1</sup> summarizes the kinds of endocytosis. By the point of view of the modeler, these three processes are made possible by vesicles (in fact this transportation mechanism is known as *vesicle-mediated transportation*) which, in the most general case, engulf the macromolecules together with molecules from the surface of the membranes (i.e. receptors). This leads to the creation of *endosomes* containing the engulfed molecules. The endosomes transfer their content inside the cell by possibly interacting with other components. The endosomes could also be degraded by the interaction with the lysosomes. We define an extension of P Systems (PE Systems) which can explicitly model the creation of endosomes and their interaction inside the cells and, consequently, can easily model these three kind of endocytosis.

---

<sup>1</sup>Pictures taken from <http://cellbiology.med.unsw.edu.au/units/science/lecture0806.htm>

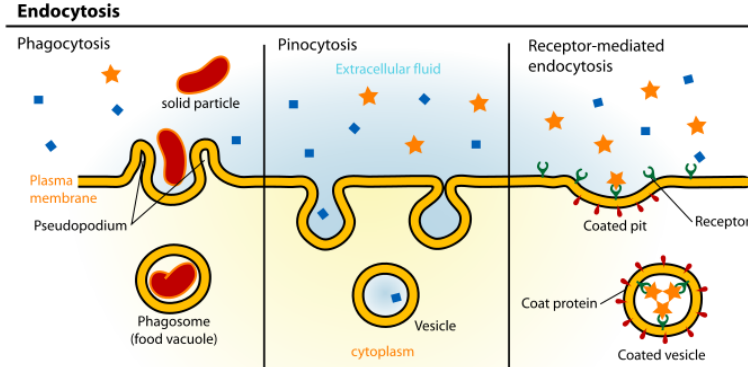


Figure 1: Three kind of endocytosis: *phagocytosis*, *pinocytosis* and *receptor-mediated endocytosis*.

This variant of P Systems, together with other modeling features such as the modeling of exocytosis (the biologically counterpart of endocytosis), and enriched with channel-mediated communication [1], would provide a powerful and complete modeling language for naturally describing transportation mechanism of molecules inside cells.

We show that PE Systems are universal even if only the simplest form of evolution rules is considered, namely non-cooperative rules. Finally, we give one application examples to show that endosomes can ease the description of biological systems when PE Systems are used as a modeling formalism.

## 2 P Systems with Endosomes

In this section we recall the definition of standard P Systems, and then we define their extension with endosomes. We will denote multisets over a finite alphabet as strings of alphabet symbols. More precisely, let  $V^*$  be the set of all strings over an alphabet  $V$ , including the empty one, denoted by  $\lambda$ . For  $a \in V$  and  $x$  in  $V^*$  we denote by  $|x|_a$  the number of occurrences of  $a$  in  $x$ . If  $V = \{a_1, \dots, a_n\}$  (the ordering is important here), then the Parikh mapping of  $x$  is defined by  $\Psi_V(x) = (|x|_{a_1}, \dots, |x|_{a_n})$ . The definition is extended in the natural way to languages. A string  $x$  represents the multiset over  $V$  with the multiplicities of objects  $a_1, \dots, a_n$  as given by  $\Psi_V(x)$ .

### 2.1 P Systems

A P System consists of a *hierarchy of membranes* that do not intersect, with a distinguishable membrane, called the *skin membrane*, surrounding them all. As usual, we assume membranes to be labeled by natural numbers. Given a set of objects  $V$ , a membrane  $m$  contains a multiset of *objects* in  $V^*$ , a set of *evolution*

rules, and possibly other membranes, called *child* membranes ( $m$  is also called the *parent* of its child membranes). Objects represent molecules swimming in a chemical solution, and evolution rules represent chemical reactions that may occur inside the membrane containing them. For each evolution rule there is a multiset of objects representing the reactants, and a multiset of objects representing the products of the chemical reaction. A rule in a membrane  $m$  can be applied only to objects in  $m$ , meaning that the reactants should be precisely in  $m$ , and not in its child membranes. The rule must contain target indications, specifying the membranes where the new objects produced by applying the rule are sent. The new objects either remain in  $m$ , or can be sent out of  $m$ , or can be sent into one of its child membranes, precisely identified by its label. Formally, the products of a rule are denoted with a multiset of *messages* of the forms:

- $(v, here)$ , meaning that the multiset of objects  $v$  produced by the rule remain in the same membrane  $m$ ;
- $(v, out)$ , meaning that the multiset of objects  $v$  produced by the rule are sent out of  $m$ ;
- $(v, in_l)$ , meaning that the multiset of objects  $v$  produced by the rule are sent into the child membrane  $l$ .

Let  $TAR$  be the set of message targets  $\{here, out\} \cup \{in_i \mid i \in \mathbb{N}\}$ . Given a set of objects  $O$  we denote with  $O_{tar}$  the corresponding set of messages  $O \times TAR$ . As a consequence, we denote with  $V_{tar}$  the set of all messages and we can define an evolution rule as a rule  $u \rightarrow v$  such that  $u \in V^*$  and  $v \in V_{tar}^*$ .

The size of the left-hand side  $u$  of an evolution rule is also called the *radius* of such a rule. If a P System contains rules of radius greater than one, then it is called a *cooperative* system. Otherwise, it is called *non-cooperative*.

Application of evolution rules is done with maximal parallelism, namely at each evolution step a multiset of instances of evolution rules is chosen non-deterministically such that no other rule can be applied to the system obtained by removing all the objects necessary to apply all the chosen rules.

A P System has a tree-structure in which the skin membrane is the root and the membranes containing no other membranes are the leaves. We assume membranes labels to be unique. A membrane structure can be represented as a balanced sequence of labeled brackets and, graphically, as a Venn diagram.

**Definition 1.** A P System is a tuple  $(V, \mu, w_1, \dots, w_n, R_1, \dots, R_n)$  where:

- $V$  is a finite alphabet whose elements are called objects;
- $\mu \subset \mathbb{N} \times \mathbb{N}$  is a membrane structure, such that  $(i, j) \in \mu$  denotes that the membrane labeled by  $j$  is contained in the membrane labeled by  $i$ ;
- $w_i$  with  $1 \leq i \leq n$  are strings from  $V^*$  representing multisets over  $V$  associated with the membranes  $1, 2, \dots, n$  of  $\mu$ ;
- $R_i$  with  $1 \leq i \leq n$  are finite sets of evolution rules associated with the membranes  $1, 2, \dots, n$  of  $\mu$ ;

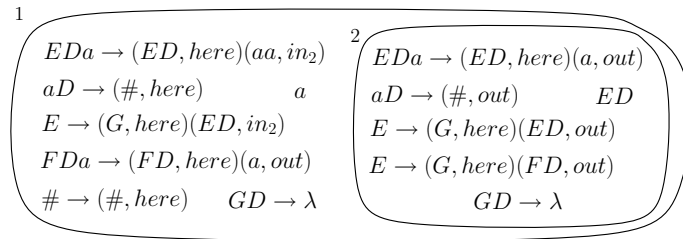


Figure 2: Example of P System that computes  $\{a^{2^n} \mid n \in \mathbb{N}\}$ .

A sequence of transitions between configurations of a given P System  $\Pi$  is called a *computation*. A computation is *successful* if and only if it reaches a configuration in which no rule is applicable. The result of a successful computation is the multiset of objects sent out of the skin membrane during the computation. Unsuccessful computations (computations which never halt) yield no result. Given a P System  $\Pi$  whose set of object is  $V$ , the result  $x \in V^*$  of a computation of  $\Pi$  can be represented as the vector of natural numbers  $\Phi_V(x)$ . The set of all vectors of natural numbers computed by  $\Pi$  is denoted  $Ps(\Pi)$ .

In Fig. 2 we show an example of P System  $\Pi_1$  computing  $\{a^{2^n} \mid n \in \mathbb{N}\}$ , namely such that  $Ps(\Pi_1) = \{2^n \mid n \in \mathbb{N}\}$ . Initially, only in membrane 2 there are rules which are applicable and send either objects  $F$  and  $D$  or objects  $E$  and  $D$  into membrane 1. In the former case the object  $a$  in membrane 1 is sent out and the computation halts. In the latter case object  $a$  in membrane 1 is consumed and two occurrences of  $a$  are sent into membrane 2. Subsequently,  $E$  is consumed and sent into membrane 2 together with  $D$ . Note that the rule which sends  $ED$  into membrane 2 cannot be applied while there are still objects  $a$  in membrane 1, otherwise by the maximal parallelism also the rule producing  $\#$  would be applied giving rise to an infinite (unsuccessful) computation. Objects  $a$  sent into membrane 2 are then sent back into membrane 1. The process of doubling and sending into membrane 2 we have explained, could be repeated an arbitrary number of times. Note that all the rules consuming  $a$  act on a single occurrence of  $a$  at a time and hence the time complexity of the computation is proportional to  $2^{n+2}$ .

## 2.2 Extension with Endosomes

In this section we formally define P Systems with endosomes (PE Systems). To this extent, we start by assuming the same membrane structure  $\mu$  of a P System. As regards objects, similarly to P Systems with peripheral proteins [4, 12], we assume that objects can be contained inside a membrane (as in classical P Systems) and on the surface of a membrane. In order to qualify a position of an object with respect to a membrane, we use *in* to identify the object inside the membrane, *out* to identify the object outside the membrane and *here* to identify the object on the surface of the membrane. Let  $TAR$  be the set of

message targets  $\{in, out, here\}$ ; given a set of objects  $O$  we denote with  $O_{tar}$  the corresponding set of messages  $O \times TAR$ , and we denote with  $V_{tar}$  the set of all messages.

We can now introduce the evolution rules of PE Systems; rules are conceptually divided in evolutionary rules (in the same sense of P Systems) and rules for the creation of endosomes. We recall that, differently from P Systems, the rules of PE Systems are conceptually associated with the surfaces of the membranes of the system. The former class of rules are of the form  $u \rightarrow v$  where  $u \in V_{tar}^+$  and  $v \in V_{tar}^*$ . The definition of cooperative and non-cooperative rules are the same as for P Systems.

Notice that this format for evolutionary rules, which are syntactically different from those of P Systems, may seem to be less expressive than the one of P Systems, in particular for rule moving objects into specific membranes (communication rules). In order to show that this is not the case, let us assume an hypothetical membrane structure  $\mu$  such that  $(l, l') \in \mu$ , namely a membrane structure in which  $l'$  is nested into  $l$ . In order to give a rule which moves an object inside membrane  $l'$  we cannot use the identifier  $in_{l'}$  in a rule of the surface of the membrane  $l$  (as in P Systems) because we cannot use the identifier  $l'$  as subscript to  $in$ . However, the same behaviour can be obtained by replacing the rule  $u \rightarrow (v, in_{l'})$  in the membrane  $l$ , as in usual P System, with the PE System rule  $(u, out) \rightarrow (v, in)$  on the surface of the membrane  $l'$ . The behaviour modeled by this rule, which is in some sense an “attraction” by the nested membrane rather than the “sending” from the top membrane, leads to result analogous to those obtained by P Systems, namely to the transportation of the object inside the nested membranes.

The rules for creating endosomes are of the form  $endo_E(u \in V^*, v \in V^*)$  where:

- $E$  is a set of evolutionary rules for the endosome;
- $u$  is the multiset of objects that must appear on the surface of the membrane containing the rule;
- $v$  is the multiset of objects that must appear outside the membrane containing the rule.

Notice that each endome has got its own evolutionary rules in set  $E$ . These rules model the behaviour of the endosome. As regards the creation of an endosome, it is necessary that objects in  $u$  are present on the surface of the membranes (in some sense they can be seen as the receptors) and that objects in  $v$  are present outside of the membrane creating the endosome (in some sense they can be seen as the molecules to be engulfed). More formally, the applicability of the endosome rule is possible in the following general case: let  $(j, i) \in \mu$  and let  $endo_E(u, v)$  be a rule belonging to the surface of the membrane  $i$ , than it can be applied only if  $u$  is a submultiset of the objects contained on the surface of the membrane  $i$ , and only if  $v$  is a submultiset of the objects contained inside the membrane  $j$ . The result of the application of such a rule is a creation of

an endosome inside membrane  $i$  containing  $u$  on its surface and containing  $v$  inside. The endosome itself behaves like a membrane having on its surface rules  $E$ .

We can now formally define a PE System as follows.

**Definition 2.** A PE System is a tuple  $(V, \mu, w_1, \dots, w_n, z_1, \dots, z_n, R_1, \dots, R_n)$  where:

- $V$  is an alphabet whose elements are called objects;
- $\mu \subset \mathbb{N} \times \mathbb{N}$  is a membrane structure, such that  $(i, j) \in \mu$  denotes that the membrane labeled by  $j$  is contained in the membrane labeled by  $i$ ;
- $w_i$  with  $1 \leq i \leq n$  are strings from  $V^*$  representing multisets over  $V$  associated with the content of membranes  $1, 2, \dots, n$  of  $\mu$ ;
- $z_i$  with  $1 \leq i \leq n$  are strings from  $V^*$  representing multisets over  $V$  associated with the surfaces of membranes  $1, 2, \dots, n$  of  $\mu$ ;
- $R_i$  with  $1 \leq i \leq n$  are finite sets of evolution rules associated with the surfaces of the membranes  $1, 2, \dots, n$  of  $\mu$ .

The notions of (successful) computation and of result of computations of PE Systems are the same as for standard P Systems.

### 3 Universality of PE Systems

In this section we prove a universality result for PE Systems by showing that any matrix grammar with appearance checking can be simulated by a PE System. As a consequence, before giving the result and its proof, we recall from [8] the definition of such variant of matrix grammars and some related notions.

#### 3.1 Matrix grammars with appearance checking

A (context-free) matrix grammar with appearance checking is a tuple  $G = (N, T, S, M, F)$ , where  $N$  and  $T$  are disjoint alphabets of non-terminals and terminals, respectively,  $S \in N$  is the axiom,  $M$  is a finite set of matrices, namely sequences of the form  $(A_1 \rightarrow x_1, \dots, A_n \rightarrow x_n)$  of context-free rules over  $N \cup T$  with  $n \geq 1$ , and  $F$  is a set of occurrences of rules in the matrices of  $M$ . For a string  $w$ , a matrix  $m : (r_1, \dots, r_n)$  can be executed by applying its rules to  $w$  sequentially in the order in which they appear in  $m$ . Rules of a matrix occurring in  $F$  can be skipped during the execution of the matrix if they cannot be applied, namely if the symbol in their left-hand side is not present in the string.

Formally, given  $w, z \in (N \cup T)^*$ , we write  $w \Longrightarrow z$  if there is a matrix  $(A_1 \rightarrow x_1, \dots, A_n \rightarrow x_n)$  in  $M$  and the strings  $w_i \in (N \cup T)^*$  with  $1 \leq i \leq n+1$  such that  $w = w_1$ ,  $z = w_{n+1}$  and, for all  $1 \leq i \leq n$ , either (1)  $w_i = w'_i A_i w''_i$  and  $w_{i+1} = w'_i x_i w''_i$ , for some  $w'_i, w''_i \in (N \cup T)^*$ , or (2)  $w_i = w_{i+1}$ ,  $A_i$  does not

appear in  $w_i$  and the rule  $A_i \rightarrow x_i$  appears in  $F$ . We remark that  $F$  consists of *occurrences* of rules in  $M$ , that is, if the same rule appears several times in the matrices, it is possible that only some of these occurrences are contained in  $F$ .

The language generated by a matrix grammar with appearance checking  $G$  is defined as  $L(G) = \{w \in T^* \mid S \Longrightarrow^* w\}$ , where  $\Longrightarrow^*$  is the reflexive and transitive closure of  $\Longrightarrow$ . The family of languages of this form is denoted by  $MAT_{ac}^\lambda$ , when rules having the empty string  $\lambda$  as right hand side ( $\lambda$ -rules) are allowed, and by  $MAT_{ac}$  when such rules are not allowed. Moreover, the family of languages generated by matrix grammars without appearance checking (i.e. with  $F = \emptyset$ ) is denoted by  $MAT^\lambda$ , when  $\lambda$ -rules are allowed, and by  $MAT$ , when such rules are not allowed. It is known that (i)  $MAT \subset MAT_{ac} \subset CS$ ; (ii)  $MAT^\lambda \subset MAT_{ac}^\lambda = RE$ , where  $CS$  and  $RE$  are the families of languages generated by context-sensitive and arbitrary grammars, respectively.

Let  $ac(G)$  be the cardinality of  $F$  in  $G$  and let  $|x|$  denote the length of the string  $x$ . A matrix grammar with appearance checking  $G = (N, T, S, M, F)$  is said to be in the *strong binary normal form* if  $N = N_1 \cup N_2 \cup \{S, \#\}$ , with these sets mutually disjoint, and the matrices in  $M$  are in one of the following forms:

1.  $(S \rightarrow XA)$ , with  $X \in N_1, A \in N_2$ ;
2.  $(X \rightarrow Y, A \rightarrow x)$ , with  $X, Y \in N_1, A \in N_2, x \in (N_2 \cup T)^*, |x| \leq 2$ ;
3.  $(X \rightarrow Y, A \rightarrow \#)$ , with  $X, Y \in N_1, A \in N_2$ ;
4.  $(X \rightarrow \lambda, A \rightarrow x)$ , with  $X \in N_1, A \in N_2, x \in T^*, |x| \leq 2$ .

Moreover, there is only one matrix of type 1,  $F$  consists exactly of all rules  $A \rightarrow \#$  appearing in matrices of type 3 and  $ac(G) \leq 2$ . We remark that  $\#$  is a trap symbol, namely once introduced it cannot be removed, and a matrix of type 4 is used only once, in the last step of a derivation.

For each matrix grammar (with or without appearance checking) there exists an equivalent matrix grammar in the strong binary normal form. Consequently, for each language  $L \in RE$  there exists a matrix grammar with appearance checking  $G$  satisfying the strong binary normal form and such that  $L(G) = L$ .

**Conventions** A matrix grammar with appearance checking in the strong binary normal form is always given as  $G = (N, T, S, M, F)$ , with  $N = N_1 \cup N_2 \cup \{S, \#\}$  and with  $n + 1$  matrices in  $M$ , injectively labeled with  $m_0, m_1, \dots, m_n$ . The matrix  $m_0 : (S \rightarrow X_{init}A_{init})$  is the initial one, with  $X_{init}$  a given symbol from  $N_1$  and  $A_{init}$  a given symbol from  $N_2$ ; the next  $k$  matrices are without appearance checking rules,  $m_i : (X \rightarrow \alpha, A \rightarrow x)$ , with  $1 \leq i \leq k$ , where  $X \in N_1, \alpha \in N_1 \cup \{\lambda\}, A \in N_2, x \in (N_2 \cup T)^*, |x| \leq 2$  (if  $\alpha = \lambda$ , then  $x \in T^*$ ); the last  $n - k$  matrices have rules to be applied in the appearance checking mode,  $m_i : (X \rightarrow Y, A \rightarrow \#)$ , with  $k + 1 \leq i \leq n, X, Y \in N_1$ , and  $A \in N_2$ .

Since the grammar is in the strong binary normal form, we have (at most) two symbols  $B^{(1)}$  and  $B^{(2)}$  in  $N_2$  such that the rules  $B^{(j)} \rightarrow \#$  appear in matrices  $m_i$  with  $k + 1 \leq i \leq n$ . For  $j \in \{1, 2\}$ , we denote with  $\ell_j$  the set

$\{i \mid \text{the matrix } m_i \text{ contains the rule } B^{(j)} \rightarrow \#\}$ . For uniformity, we also denote  $\ell_0 = \{1, 2, \dots, k\}$  and  $\ell = \ell_0 \cup \ell_1 \cup \ell_2 = \{1, 2, \dots, n\}$  (note that  $0 \notin \ell$ ). Clearly, the sets  $\ell_0, \ell_1$  and  $\ell_2$  are disjoint.

We remark that in matrix grammars in strong binary normal forms we can assume that all symbols  $X \in N_1$  appear as the left-hand side of a rule from a matrix: otherwise, the derivation is blocked after introducing such a symbol, hence we can remove these symbols and the matrices involving them.

### 3.2 Universality

We prove that PMC Systems are universal by showing that the family, denoted  $PsE_2(ncoo)$ , of sets  $Ps(\Pi_e)$  of results computed by PE Systems with at least two membranes and with non-cooperative rules is equivalent to the family, denoted  $PsRE$ , of the images of all the languages in  $RE$  obtained through the Parikh mapping (this is the family of recursively enumerable sets of vectors of natural numbers). As P Systems with non-cooperative rules are not universal, our result implies that universality is due to the presence of endosomes.

**Theorem 1.**  $PsE_2(ncoo) = PsRE$ .

*Proof.* It is enough to show that for a  $G$  in strong binary normal form there is a PE System  $\Pi_G$  such that  $Ps(\Pi_G) = \Psi_T(L(G))$ . We assume that the output of this PE System is given by the objects sent out from the skin membrane. The alphabet we take into consideration is given by  $T \cup N_1 \cup N_2 \cup \{c\} \cup \{c_i, d_i, d'_i \mid i = 1, 2\}$ . We build  $\Pi_G$  as a system with a root membrane, labeled 1, and one child membrane labeled 2. All the objects encoding the grammar will be stored inside membrane 1 and the matrixes will be simulated by membrane 2. The initial configuration is given by the objects corresponding to  $X_{init}$  and  $A_{init}$  contained in membrane 1, namely objects of  $w_1$ , and by the token  $c$  contained on the surface of membrane 2, namely  $z_2 = \{c\}$ .

This PE System works as follows: it has a cyclic behaviour such that, at the beginning of the cycle, at most one endosome in membrane 2 can be created and, if possible, all terminal symbols inside membrane 1 are sent out as output symbols. The created endosome can start a series of steps resulting in the interpretation of the application of a matrix or, differently, it can start a checking phase to model the fact that, if there exist non terminal symbols which cannot be rewritten by any grammar, then the computation has not to halt. In the case in which it starts the interpretation of a matrix of type 2 or 4 (a matrix  $m_i$  with  $1 \leq i \leq k$ ), the involved non terminals are taken by the endosome which contains as rules the ones interpreting the matrix. Objects will be sent into membrane 2 by these rules creating the result of applying the corresponding matrix to the non terminals. Subsequently, these objects are sent out to membrane 1 to restart the cyclic behaviour. We recall that during this process no other endosomes can be created, so no other matrixes can be simulated. Differently, in the case in which a matrix of type 3 (a matrix  $m_i$  with  $k + 1 \leq i \leq n$ ) is applied, the single non terminal of  $N_1$  is taken into the endosome. The endosome will work in the same sense of the endosomes interpreting matrixes of type 2 and 4 even

though, at the end of the application of this matrix, instead of restarting with the cyclic behaviour, a checking process is started. This process checks, by creating endosomes, the presence of the proper non terminal symbol  $B^{(j)}$ . If this symbol is found, a special endosome is created which will introduce a trap symbol in this PE System so that the computation will not halt. Analogously, if it is not found, an endosome will restore the configuration of this PE System so that the cyclic behaviour can start again.

We list now the rules of  $\Pi_G$ . Membrane 1 contains just one single set of rules to create the output of the PE System:

1.  $\{(a, here) \rightarrow (a, out) \mid \forall a \in T\}$ . All terminal objects in membrane 1 are sent out as output.

All the interpretation of any matrix is done by the rules of the membrane 2 which are the following:

1.  $\forall X \in N_1 \cup N_2. endo_{(X, in) \rightarrow (\#, out)}(c, X)$ . If any non terminal is present in membrane 1,  $\Pi_G$  will always be able to create, by using an endosome, a trap symbol inside membrane 2. This will ensure that, if a derivation of  $G$  reaches a deadlock configuration, then  $\Pi_G$  can always enter an endless configuration.
2.  $\forall a \in N_1 \cup N_2 \cup T. (a, in) \rightarrow (a, out)$ . Every terminal and non terminal present inside membrane 2 is sent out to membrane 1.
3.  $(c, in) \rightarrow (c, here)$ . Object  $c$  inside membrane 2 is restored on the surface of membrane 2 so that other endosomes can be created.
4.  $(\#, in) \rightarrow (\#, in)$ . The trap symbol lets this computation not to be recognized such.
5.  $\forall (X \rightarrow \alpha, A \rightarrow x) \in \{m_i \mid 1 \leq i \leq k\}$ .  
 $endo_{(X, in) \rightarrow (\alpha, out), (A, in) \rightarrow (x, out), (c, here) \rightarrow (c, out)}(c, XA)$ . For any rule of type 2 and 4, we create an endosome by taking  $XA$  from membrane 1 and  $c$  from the surface of membrane 2 (this locks the creation of other endosomes). The endosome contains rules to rewrite  $X$  and  $A$  with the result of applying the matrix. Object  $c$  is not consumed and sent out to membrane 2 together with  $\alpha$  and  $x$ .
6.  $\forall (X \rightarrow Y, A \rightarrow \#) \in \{m_i \mid k + 1 \leq i \leq n\}$ .  
 $endo_{(X, in) \rightarrow (Y, out), (c, here) \rightarrow (c_i, out)}(c, X)$ . For any rule with appearance checking, we create an endosome by taking only  $X$  from membrane 1 and  $c$  from the surface of membrane 2 (this locks the creation of other endosomes). The endosome contains rules to rewrite  $X$  with  $Y$  and  $c$  with  $c_i$ . Both the objects are sent out to membrane 2.
7.  $(c_i, in) \rightarrow (c_i, here)(d_i, here)$ . Object  $c_i$ , together with a new object  $d_i$ , is moved on the surface of membrane 2.

8.  $endo_{(B^{(i)},in)\rightarrow(\#,out)}(c_i, B^{(i)})$ . This implements the appearance checking of grammar  $G$ . We create, if possible, an endosome by taking only  $B^{(i)}$  from membrane 1 and  $c_i$  from the surface of membrane 2. The endosome creates a trap symbol in membrane 2; this will make  $\Pi_G$  start an endless computation.
9.  $(d_i, here) \rightarrow (d'_i, here)$ . The symbol  $d_i$  is rewritten in the same place as  $d'_i$ . This is done even if also rule 8 can be applied. However, in the case that rule 8 cannot be applied (namely  $B^{(i)}$  was not present), this completes the appearance checking operation and lets  $\Pi_G$  start an operation which will restart its cyclic behaviour.
10.  $endo_{(c_i,here)\rightarrow(c,out),(d'_i,here)\rightarrow\lambda}(c_i d'_i, \emptyset)$ . This endosome lets  $\Pi_G$  restart its cyclic behaviour. We create an endosome by simply taking both the control symbols only  $c_1$  and  $d'_i$  from the surface of membrane 2. The endosome destroys  $d'_i$  and rewrites  $c_i$  with  $c$  in membrane 2 (restarting  $\Pi_G$  will be obtained by applying rule 3).

It is clear that these rules, applied in a proper order, provide the correct interpretation of the application of any matrix to the starting symbols of the grammar and, consequently, we get  $Ps(\Pi_G) = \Psi_T(L(G))$  which concludes the proof.  $\square$

## 4 An Application: the EGF Signaling Pathway

In this section we give an application of PE systems to the description of the initial phases of the EGFR signalling cascade.

In Biology, signal transduction refers to any process by which a cell converts one kind of signal or stimulus into another. Signals are typically proteins that may be present in the environment of the cell. In order to be able to receive the signal, namely to recognize that the corresponding protein is available in the environment, a cell exposes some receptors on its external membrane. A receptor is a transmembrane protein that can bind to a signal protein on its extracellular end. When such a binding is established, the intracellular end of the receptor undergoes a conformational change that enables interaction with other proteins inside the cell. This typically causes an ordered sequence of biochemical reactions inside the cell, usually called signalling pathway, that are carried out by enzymes and may produce different effects on the cell behaviour.

A complex signal transduction cascade, that modulates cell proliferation, survival, adhesion, migration and differentiation, is based on a family of receptors called epidermal growth factor receptors (EGFRs). While EGFR signalling is essential for many normal morphogenic processes, the aberrant activity of these receptors has been shown to play a fundamental role in proliferation of tumor cells. Epidermal growth factor receptors (*EGFR*) are produced by specific genes in the DNA (through the RNA) and they are located on the cell

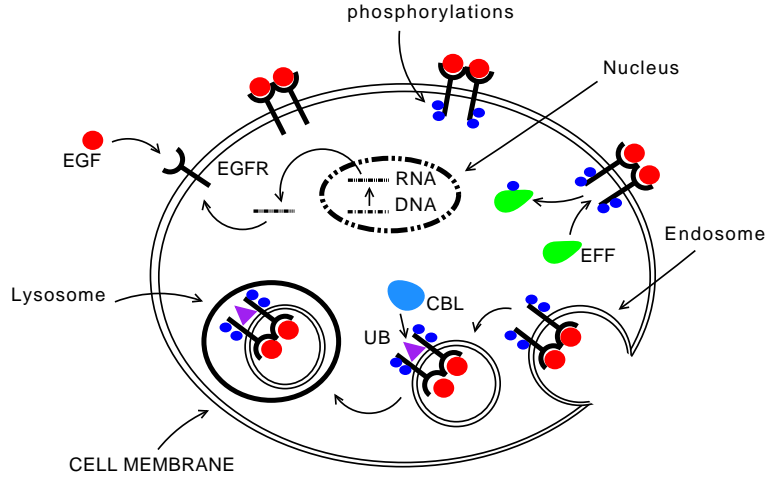


Figure 3: The EGF signaling pathway.

surface. Receptors are activated by the binding with a specific ligand (epidermal growth factor, *EGF*) to form a EGFR (ligand-receptor) complex (*COM*). Upon activation, EGFR undergoes a transition from a monomeric form to an active dimeric one (*DIM*). EGFR dimerization stimulates its intracellular phosphorylation (*DIMp*) which activates signalling proteins. These activated signalling proteins (effector proteins) initiate several signal transduction cascades, leading to DNA synthesis and cell proliferation. After the activation of effector proteins, ligand-receptor dimers are internalized in endosomes. An ubiquitin ligase, known as Cbl, binds an ubiquitin protein (*UB*) to the dimer (ubiquitination). The ubiquitin protein targets the dimers for lysosomal degradation (see Figure 3).

The PE system modeling the EGF is given in Figure 4. The rules which describe its behaviour are the following for membrane 2:

1.  $(EGFR, in) \rightarrow (EGFR, here)$
2.  $(EGF, out)(EGFR, here) \rightarrow (COM, here)$
3.  $(COM, here)(COM, here) \rightarrow (DIM, here)$
4.  $(DIM, here)(P, in) \rightarrow (DIMp, here)$
5.  $(DIMp, here)(SHC, in) \rightarrow \dots$
6.  $(rna, in) \rightarrow (EGFR, in)$
7.  $endo_{(DIMp, here)}(UB, out) \rightarrow (P, out), (DIMp, here) \rightarrow (EGFR^2, out)(DIMp, \emptyset)$

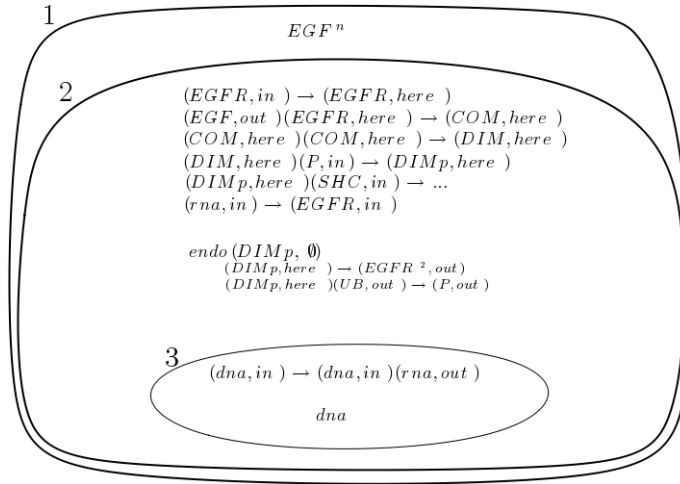


Figure 4: A PE systems model of the EGF signaling pathway.

Notice that rule 5 is not complete in the sense that it should rewrite the  $DIMp$  on the surface of the membrane and the contained  $SHC$  into a complex in order to start a sequence of intra-cellular events leading to the duplication of the cell. However, for the explanatory aim of this application example, it is not of interest to fully describe this part of the model.

Finally, membrane 3 contains just one rule  $(dna, in) \rightarrow (dna, in)(rna, out)$ .

The behaviour of this PE System is straightforward, membrane 1 models the environment external to the cell, membrane 2 represents the cell surface and membrane 3 is the nucleus. In the external environment  $EGF$  corresponds to the epidermal growth factor EGF which can bind the receptor on the surface of the cell. The receptor is modeled by  $EGFR$  in membrane 2, which can move on the surface of the membrane. The complex of  $EGF$  with the receptor is obtained by rewriting  $EGF$  and  $EGFR$  with the complex  $COM$  on the surface of membrane 2. After the binding of two complexes we can bind them leading to a dimer  $DIM$ . Such a dimer, present on the surface of the membrane, can be phosphorylated by a phosphorus  $P$  inside the cell. Such phosphorylated dimer  $DIMp$  could start a chain of actions we do not model here. Furthermore, it can be enclosed in an endosome which could, in presence of ubiquitin  $UB$ , reproduce the phosphorus inside the cell or, differently, the two receptors. The nucleus of the cell (membrane 3) is responsible for the production of  $EGFR$  through the DNA and RNA ( $dna$  and  $rna$ ). The  $rna$  reaches the cell cytoplasm and there it produces  $EGFR$  which is sent, again, to the cell surface.

## 5 Future Work and Conclusions

In this paper we presented an extension of P Systems, called P Systems with Endosomes (PE Systems), in which endosomes can be explicitly modeled. P Systems are computing devices inspired by the structure and the functioning of a living cell. A P System consists of a hierarchy of membranes, each of them containing a multiset of objects, a set of evolution rules, and possibly other membranes. Evolution rules are applied to the objects of the same membrane with maximal parallelism. PE Systems extend P Systems maintaining the main features of the formalism but adding the possibility of explicitly modeling endosomes. Modeling endosomes is the basis for modeling vesicle-mediated transportation mechanisms, in particular endocytosis, which can be divided in three main forms (pynocytosis, phagocytosis and receptor-mediated endocytosis) can be clearly modeled by using PE Systems. PE Systems uses some ideas taken from other variants of P Systems, in particular as regards objects which can be stored on the surface of the membranes we got inspiration by the works on P System with peripheral proteins [4, 12]. Furthermore, as regards other calculi, operations for modeling transportation mechanism already have been introduced in Brane Calculi [3]. Although similar, PE Systems permit to model in a clearer way these mechanisms. An analysis of PE Systems and Brane Calculi [3] (and also some of their variants like projective Brane Calculi [5]) could be done along the line of the one done in [9, 10] for P Systems and Brane Calculi.

As regards expressiveness of this formalism, we showed that PE Systems are universal even if only the simplest form of evolution rules is considered, namely non-cooperative rules. This expressiveness is achieved by the use of endosomes as classical P Systems with this kind of rules are shown not to be universal [7].

At the end of the paper we have given an application example describing the modeling of the of the initial phases of the EGFR signalling cascade.

## References

- [1] Barbuti, R., Maggiolo-Schettini, A., Milazzo, P. Tini, S.: P Systems with Transport and Diffusion Membrane Channels. Int. Workshop on Concurrency, Specification and Programming (CS&P'08), Gross Vaeter, Germany, September, 2008.
- [2] Bottoni, P., Martin-Vide, C., Păun, G., Rozemberg., G.: Membrane systems with promoters/inhibitors. *Acta Informatica* **38** (2002), pp. 695–720.
- [3] Cardelli, L.: Brane calculi. Interactions of biological membranes. In: Danos, V., Schachter, V. (Eds.), *LNCS 3082*, pp. 257-280.
- [4] Cavaliere, M., Seawards, S.: Membrane systems with peripheral proteins: transport and evolution. *ENTCS* **171** (2007), pp. 37–53.

- [5] Danos, V. Pradalier, S.: Projective Brane Calculus. LNCS **3082**, pp. 134–148.
- [6] Freund, R., Oswald, M.: P systems with activated/prohibited membrane channels. Proc. of WMC 2002, LNCS **2597** (2003), pp. 261–269.
- [7] Păun, G.: Computing with membranes. Journal of Computer and System Sciences **61** (2000), pp. 108–143
- [8] Păun, G.: Membrane Computing. An Introduction. Springer (2002).
- [9] Cardelli, L., Păun, G.: An universality result for a (mem)brane calculus based on mate/drip operations. Internat. J. Found. Comput. Sci. **17**(1), pp. 49–68.
- [10] Păun, G.: Membrane computing and brane calculi. Old, new, and future bridges. Theoretical Computer Science **404**(1-2), pp. 19–25.
- [11] Păun, A., Păun, G.: The power of communication: P systems with symport/antiport. New Generation Computing **20** (2002), pp. 295–306.
- [12] Păun, A., Popa, B.: P Systems with Proteins on Membranes. Fundamenta Informaticae **72**(4) (2006), pp. 467 – 483.
- [13] P Systems, web page. <http://ppage.psystems.eu/>.