Size and Power of Extended Gemmating P Pystems

Daniela BESOZZI[1], **Erzsébet CSUHAJ-VARJÚ**[2], **Giancarlo MAURI**[3], **Claudio ZANDRON**[3]

[1]Università degli Studi di Milano
Dipartimento di Informatica e Comunicazione
Via Comelico 39, 20135 Milano, Italy
E-mail: besozzi@dico.unimi.it
[2]Computer and Automation Research Institute
Hungarian Academy of Sciences
Kende u. 13-17, H-1111 Budapest, Hungary
E-mail: csuhaj@sztaki.hu
[3]Università degli Studi di Milano-Bicocca
Dipartimento di Informatica, Sistemistica e Comunicazione
Via Bicocca degli Arcimboldi 8, 20136 Milano, Italy
E-mail: {mauri,zandron}@disco.unimib.it

Abstract. In [2] P systems with gemmation of mobile membranes were examined. It was shown that (extended) systems with eight membranes are as powerful as the Turing machines. Moreover, it was also proved that extended gemmating P systems with only pre-dynamical rules are still computationally complete: in this case nine membranes are needed to obtain this computational power. In this paper we improve the above results concerning the size bound of extended gemmating P systems, namely we prove that these systems with at most five membranes (with meta-priority relations and without (in/out) communication rules) form a class of universal computing devices, while in the case of extended systems with only pre-dynamical rules six membranes are enough to determine any recursively enumerable language.

1 Introduction

P systems with gemmation of mobile membranes were introduced in [3], defining a new kind of communication between membranes which is inspired by certain biological processes in living cells. The biological background of the new model can be briefly summarized as follows: the cellular membranes are selectively permeable to small substances as, for example, water and gases, but not to bigger substances as proteins. These bigger substances are communicated among the cells by means of vesicles, encased on their cytosolic face by a specific protein which causes their budding from the membrane. When the vesicle fuses with its target membrane, then the carried proteins are introduced inside it, where they can undergo different chemical reactions. The reader can easily observe

that this process can be modelled by so-called mobile membranes, that is, we can consider some objects in the original membrane to be transported by means of small membranes to a target membrane and then being fused with it.

To simulate these features, [3] introduced P systems with gemmation of mobile membranes. These are variants of P systems with simple membrane structures, where the skin membrane contains only elementary membranes with string objects which correspond to proteins or any other structured bigger substances. These strings evolve according to operations with biochemical motivations, namely mutation, replication and splitting. The mutation in this case corresponds to the application of a context-free rule. Any membrane is provided with a set of classical evolution rules and a set of so-called pre-dynamical rules, which are rules defining the gemmation of the mobile membranes. There is a meta-priority relation defined between the set of classical evolution rules and the set of pre-dynamical rules which is needed to simulate the completion of the maturation path of an object. A pre-dynamical rule is a particular variant of an evolution rule which also indicates the membrane where the string must be communicated. After a pre-dynamical rule is used, the modified string object(s) is (are) transported into the target membrane, and from then it (they) will evolve according to the rules of this membrane. This procedure corresponds to the gemmation and the fusion of the mobile membrane. In particular, the output of the system is due to the fusion of a mobile membrane with the skin membrane: this process causes the release of the objects outside the system and simulates the biological process of exocytosis.

In [3, 2] P systems with gemmation of mobile membranes were examined. It was shown that these systems are as powerful as the Turing machines, in the case of extended systems even with eight membranes [2]. Moreover, it has also been proved that extended gemmating P systems with only pre-dynamical rules are still computationally complete: in this case nine membranes are needed to obtain this computational power [2]. For detailed information on P systems with gemmation of mobile membranes consult also [1].

In this paper we improve the above results concerning the size bound of extended gemmating P systems, namely we prove that these systems consisting of five membranes (with meta-priority relations and without (in/out)-rules) form a class of computationally complete devices, and for extended systems with only pre-dynamical rules six membranes are enough to reach the power of the Turing machines.

2 Basic Definitions

We assume that the reader is familiar with formal language theory; for details and more information we refer to [5]. Throughout the paper we use standard notions and notations: we denote by V^* the set of all words over an alphabet V, including the empty word, λ. The class of recursively enumerable languages is denoted by RE; this is the class of languages accepted by the Turing machines or generated by the class of phrase-structure or 0-type grammars.

In this paper we shall use the notion of a Geffert normal form for the phrase-structure grammars (see [5]). According to this result, for any recursively enumerable language over an alphabet T there exists a generating phrase-structure grammar $G = (N, T, P, S)$, where $N = \{S, A, B, C\}$, is the set of nonterminals, T is the set of terminals, S is the start symbol of G, and the rules in P are of the forms $S \to uSv$, $S \to x$, with $u, v, x \in (T \cup \{A, B, C\}^*)$, and $ABC \to \lambda$.

In the following we recall the definition of extended P systems with gemmation of mobile membranes from [2]. For detailed information about P systems or membrane systems consult [4].

A membrane structure μ is a construction consisting of several membranes hierarchically embedded in a unique membrane, called the skin membrane. A membrane structure can also be identified with a string of correctly matching square parentheses, placed in a unique pair of matching parentheses. Each pair of matching parentheses corresponds to a membrane.

By [2], gemmating P systems use only membrane structures of depth 2, that is $\mu = [_0[_1]_1[_2]_2 \dots [_{n-1}]_{n-1}[_n]_n]_0$. The skin membrane will be always labelled with the number 0, while the inner membranes will be labelled with the numbers $1, \dots, n$.

P systems with gemmation of mobile membranes work with string-objects where the evolution rules are able to multiply the number of the strings. Therefore, a multiset of finite support is associated with every region of the membrane structure. This multiset is a map that associates a multiplicity to every string present in the region, that is we define $M_i : V^+ \to \mathbf{N}$ where $M_i = \{(x_1, M_i(x_1)), \dots, (x_p, M_i(x_p))\}$, for some $x_k \in V^+$ such that $M(x_k) > 0$, for all $k = 1, \dots, p, i = 0, 1, \dots, n$.

Gemmating P systems are defined with three types of rules with biochemical inspiration: mutation, replication, and splitting of a string. In this paper we use only mutation rules.

A mutation rule is a context free rule $r_m : a \to u$, where $a \in V$ and $u \in V^*$. For strings $w_1, w_2 \in V^+$ we write $w_1 \Longrightarrow_{r_m} w_2$ if $w_1 = x_1 a x_2$ and $w_2 = x_1 u x_2$, for some x_1, $x_2 \in V^*$.

When these operations are applied to strings in the membrane systems, target indications are added to the rules, determining the regions where the obtained strings will be communicated at the next step.

With each region $i = 0, 1, \dots, n$ we associate two distinct sets of rules:

- A set C_i of classical evolution rules, that is a set of mutation rules of the form $a \to \alpha$, where $a \in V$ and $\alpha = (u, tar)$, with $u \in V^*$, and $tar \in \{here, out\}$ for $i = 1, \dots, n$, $tar \in \{here, out\} \cup \{in_1, \dots, in_n\}$ for $i = 0$.

- A set D_i of pre-dynamical evolution rules, that is a set of mutation rules of the form $a \to (u, here)$, with $a \in V$, such that given a string $w_1 = x_1 a$ (or $w_1 = a x_2$) we obtain $w_2 = x_1 u$ ($w_2 = u x_2$, respectively), where $u \in V^* \cdot \{@_j\}$ ($u \in \{@_j\} \cdot V^*$, respectively) and $x_1, x_2 \in V^*$. Letter $@_j$ is a special symbol not in V and $j \in \{0, 1, \dots, n\}, j \neq i$.

Notice that a pre-dynamical rule can introduce the special symbol $@_j$ only at the ends of the string. We will always consider the set D_0 as an empty set, that is no pre-dynamical rule will ever be defined inside the skin membrane.

When a symbol $@_j$ appears in some string w present in a membrane i, for $j \neq i$, then inside the P system two sequential and dynamical communication processes take place. We say that a mobile membrane, which we write as a couple of well-matching round brackets $(_{i,j})_{i,j}$ carries the string w from the originating membrane i to the target membrane j.

The communication steps are defined by means of the following rules.

- The gemmation of a mobile membrane is defined as follows:

$$[_0 \dots [_i \dots, w@_j, \dots]_i \dots]_0 \to [_0 \dots [_i \dots]_i (_{i,j}\, w\,)_{i,j} \dots]_0$$

  for some $i \in \{1, \dots, n\}, j \in \{0, 1, \dots, n\}, j \neq i, w \in V^+$.

  During this first phase of the step the symbol $@_j$ is removed, its subscript becomes

the second label of the mobile membrane, then string w leaves membrane i and enters the created mobile membrane. If there are several strings of the form $w_1@_j, \ldots, w_k@_j$ inside membrane i, all of them with the same target membrane j, then a single common mobile membrane will be budded off from membrane i:

$$[_0 \ldots [_i \ldots, w_1@_j, \ldots, w_k@_j, \ldots]_i \ldots]_0 \rightarrow$$
$$[_0 \ldots [_i \ldots]_i (_{i,j} w_1, \ldots, w_k)_{i,j} \ldots]_0.$$

If inside membrane i there are strings of the form $w_1@_{j_1}, \ldots, w_{h_1}@_{j_1}, w_{h_1+1}@_{j_2}, \ldots, w_{h_2}@_{j_2}, \ldots, w_{h_{k-1}+1}@_{j_k}, \ldots, w_{h_k}@_{j_k}$ $(h_k \geq k)$ such that $j_1, \ldots, j_k$ are pairwise different, then k different mobile membranes will be gemmated, each one containing the strings directed to the specified membrane:

$$[_0 \ldots [_i \ldots, w_1@_{j_1}, \ldots, w_{h_1}@_{j_1}, \ldots, w_{h_{k-1}+1}@_{j_k}, \ldots, w_{h_k}@_{j_k}, \ldots]_i \ldots]_0 \rightarrow$$
$$[_0 \ldots [_i \ldots]_i (_{i,j_1} w_1, \ldots, w_{h_1})_{i,j_1} \ldots (_{i,j_k} w_{h_{k-1}+1}, \ldots, w_{h_k})_{i,j_k} \ldots]_0.$$

The case when a membrane i, for some $i \in \{1, \ldots, n\}$, contains one or more strings of the form $@_j w$, for some $j \in \{0, 1, \ldots, n\}$ can be analogously handled. Obviously, the same holds when a membrane i contains some strings of both forms.

- The fusion of the mobile membrane is defined in the following way:

$$[_0 \ldots (_{i,j} w)_{i,j} [_j \ldots]_j \ldots]_0 \rightarrow [_0 \ldots [_j \ldots, w, \ldots]_j \ldots]_0,$$

for some $i \in \{1, \ldots, n\}, j \in \{1, \ldots, n\}, j \neq i, w \in V^+$.
During this second phase of the communication step the mobile membrane becomes a part of the target membrane, leaving its contents inside it.
In particular, if $j = 0$ the mobile membrane fuses with the skin membrane (in this way we simulate the biological process of exocytosis) and the objects exit the system:

$$[_0 \ldots (_{i,0} w)_{i,0} \ldots]_0 \rightarrow [_0 \ldots]_0 w.$$

To keep the construction closer to the functioning of real cells, we define a meta-priority relation between the whole set C_i and the whole set D_i, for all $i = 1, \ldots, n$, meaning that all applicable classical rules in C_i must be used before any applicable pre-dynamical rule in D_i. We remark that we do not define any priority relation between rules in the set C_i neither between rules in the set D_i.

Now we give the formal definition of an extended P system Π with gemmation of mobile membranes (or an extended gemmating P system Π, in short) of degree $n + 1$, $n \geq 0$, as follows:

$$\Pi = (V, T, \mu, M_0, \ldots, M_n, (C_0, \emptyset), (C_1, D_1), \ldots, (C_n, D_n)),$$

where

– V is an alphabet not containing the symbols $@_0, @_1, \ldots, @_n$;

– $T \subseteq V$ is the output (terminal) alphabet;

– $\mu = [_0[_1]_1[_2]_2 \dots [_{n-1}]_{n-1}[_n]_n]_0$ is a membrane structure of depth 2 and degree $n+1$;

– $M_0, \dots, M_n$ are multisets of finite support over V^+;

– (C_i, D_i), for all $i = 0, 1, \dots, n$, are a set of classical evolution rules and a set of pre-dynamical evolution rules, respectively. The set C_i, $i = 1, \dots, n$, has a meta-priority above D_i as far as the application of all of its rules is concerned. The set D_0 is empty.

An extended gemmating P system works as follows: the regions are processed simultaneously, that is, in every step, inside each region, all the strings which can be the subject of an evolution rule are simultaneously rewritten. The rules to be applied can be nondeterministically chosen among all the applicable rules, in accordance with the meta-priority defined over the set of classical rules and the set of pre-dynamical rules. At each step of a computation a string can be rewritten by one rule only. The strings resulting after the application of a rule can remain inside the membrane where they are placed, or can be communicated by mobile membranes or by (in/out) communication to the regions specified by the target indications.

The membrane structure at a given moment, together with all multisets of objects associated with the regions defined by the membrane structure, form the configuration of the system at that moment.

For two configurations $\sigma_1 = (\mu, M'_0, \dots, M'_n)$ and $\sigma_2 = (\mu, M''_0, \dots, M''_n)$ of Π, we say that σ_2 is obtained from σ_1 in one transition by applying the rules in (C_i, D_i), $0 \leq i \leq n$, in accordance with the meta-priority relation.

A sequence of transitions forms a computation. A computation halts when there is no rule which can be further applied in the current configuration. On the contrary, we say that a computation is non-halting if there is at least one rule which can be applied forever.

The output of the P system Π (or the language of Π) is the set of strings over T expelled from the system during the computation. The language generated by Π is denoted by $L(\Pi)$. Non-halting computations provide no output.

3 Size and Power of Extended Gemmating P Systems

In this section we improve the result of [2], namely, we show that extended gemmating P systems consisting of five membranes with meta-priority relations and without the use of (in/out)-rules are as powerful as the Turing machines, and extended systems with only pre-dynamical rules and without any other feature need six membranes for this purpose.

We denote by $EGemP_m(MPri, \alpha)$, for $\alpha \in \{(in/out), n(in/out)\}$, the family of languages generated by extended gemmating P systems of degree at most m, for $m \geq 1$, with relation of meta-priority and with the use (if $\alpha = (in/out)$) or without the use (if $\alpha = n(in/out)$) of communication rules of type (in/out). If we use $*$ instead of m, then we refer to the whole class of languages of extended gemmating P systems with relation of meta-priority and with the use (if $\alpha = (in/out)$) or without the use (if $\alpha = n(in/out)$) of communication rules of type (in/out).

Furthermore, let us denote by $EGemP_m(Dyn)$ the family of languages generated by extended gemmating P systems of degree m, for $m \geq 1$, with membranes having only pre-dynamical rules. Analogously to the previous notations, $EGemP_*(Dyn)$ denotes the

whole class of languages of extended gemmating P systems with membranes having only pre-dynamical rules.

We start with the case of extended gemmating P systems with only pre-dynamical rules; the proof of the other statement can easily be obtained by modifying the proof of the following theorem.

Theorem 1 $EGemP_6(Dyn) = EGemP_*(Dyn) = RE$.

Proof. By [2] we should prove only the inclusion $RE \subseteq EGemP_6(Dyn)$. For this purpose, we modify the proof of the statement $RE \subseteq EGemP_9(Dyn)$ in [2], where for any phrase-structure grammar $G = (N, T, S, P)$, given in the Geffert normal form, a simulating extended gemmating P system with nine membranes and with only pre-dynamical rules is constructed. The basic idea of this proof is the so-called "rotation-and-simulation", which is a technique widely used in formal language theoretic models of molecular computing. According to this method, to simulate the application of a production to a symbol A occurring somewhere in the middle of the string, we move (we "rotate") one symbol step by step from the right end to the left end of the string, until the symbol A appears on the right end. Then we apply the production to A. To guarantee the correct simulation of a derivation in the grammar, a special symbol \$ is introduced for marking the position where the original (unrotated) string begins. Since pre-dynamical rules can be applied only at the right end or at the left end of the string, this technique is well applicable.

Let $L \subseteq T^*$ be a recursively enumerable language generated by a phrase-structure grammar $G = (N, T, S, P)$ given in the Geffert normal form. Let $N' = (N \setminus \{S\})$, and let us denote the elements in $(N' \cup T)$ by $E_1, \ldots, E_n$, $n \geq 1$. Furthermore, let $\$, X, Y \notin (N \cup T)$ be auxiliary symbols. Symbol \$, also denoted by E_{n+1}, is used for marking the beginning of the string.

We construct the simulating extended gemmating P system of degree 6 as follows. Let

$$\Pi = (V, T, \mu, M_0, \ldots, M_5, \emptyset, D_1, \ldots, D_5)$$

with:

$$\begin{aligned}
V &= N \cup T \cup \{X, \$, Y\} \\
&\cup \{(E_i, j) \mid E_i \in N' \cup T \cup \{\$\}, 1 \leq i \leq n+1, 0 \leq j \leq n+1\} \\
&\cup \{(X_i, j) \mid 1 \leq i \leq n+1, 0 \leq j \leq n+1\}, \\
\mu &= [_0[_1]_1[_2]_2[_3]_3[_4]_4[_5]_5]_0, \\
M_1 &= \{X\$S \mid S \text{ is the axiom in } G\}, \\
M_i &= \emptyset, \text{ for all } i = 0, 2, \ldots, 5.
\end{aligned}$$

Let Π be given with the following sets of pre-dynamical rules:

$$\begin{aligned}
D_1 &= \{S \to wY@_2 \mid S \to w \in P\} \cup \{C \to \lambda@_4, \$ \to \lambda@_2\} \\
&\cup \{E_i \to (E_i, 0)@_3 \mid E_i \in N' \cup T \cup \{\$\}, 1 \leq i \leq n+1\}; \\
D_2 &= \{Y \to \lambda@_1, A \to \lambda@_1, X \to @_0\lambda\}; \\
D_3 &= \{X \to @_4(X_i, 0) \mid 1 \leq i \leq n+1\} \\
&\cup \{(X_i, j) \to @_4(X_i, j+1) \mid 0 \leq j < i \leq n+1\}; \\
D_4 &= \{(E_i, j) \to (E_i, j+1)@_3 \mid 0 \leq j < i \leq n+1\} \\
&\cup \{(E_i, i) \to \lambda@_5 \mid 1 \leq i \leq n+1\} \cup \{B \to \lambda@_2\}; \\
D_5 &= \{(X_i, i) \to @_1 X E_i \mid 1 \leq i \leq n+1\}.
\end{aligned}$$

The system works as follows. Membranes 1, 2, 4 are used for simulating the productions in P, and membranes 3, 4, and 5 are used for performing the rotation of the rightmost symbol in the current string. Membrane 2 is also used to send the received strings outside the system. Notice that membrane 4 takes part both in rotating the symbols and in simulating the rule $ABC \to \lambda$. No rules are given for the skin membrane.

Symbols (E_i, j) and (X_i, j) are used in making the rotation of $E_i \in N' \cup T \cup \{\$\}$ from the right end to the left end of the string. Number j in (E_i, j) is used as a counter with value j under the rotation, to guarantee that if we delete (E_i, i) from the right end of the string, then we correctly append the same symbol, E_i, to its left end.

Let us assume now that at some moment a string of the form $X\alpha\$z$ can be found in membrane 1, for $\alpha \in (N' \cup T)^*$ and $z = z_1 S z_2$ or $z = z_3$, with $z_1, z_2, z_3 \in (N' \cup T)^*$. At the first step of the functioning, we have $\alpha = \lambda$ and $z = S$.

Then the following cases are possible:

1. $r = S$. We have to use a rule $S \to wY@_2$, which simulates the corresponding production $S \to w$ in P and sends the string to membrane 2. Here we can apply the rule $Y \to \lambda@_1$, which sends the string back to membrane 1. Then, if the rightmost symbol of the new string is S, the process is repeated, otherwise the rotation of a symbol or the deletion of symbol C can follow. Note that at any time, in membrane 2, also the rule $X \to @_0\lambda$ can be used causing the current string to exit the skin membrane. Anyway, since the system is extended, only the strings over the terminal alphabet T will contribute to the generated language.

2. $r = E_i$, for $E_i \in N' \cup T$. In this case we can obtain the string $X\alpha\$z'(E_i, 0)$, with z' such that $z = z'r$, and we send it to membrane 3, where the rotation of E_i, the rightmost symbol of the string will start. If $r = C$, then the simulation of the rule $ABC \to \lambda$ can also follow, since we can use the rule $C \to \lambda@_3$. After the application of this rule, the string is forwarded to membrane 4, where the only rule applicable at this moment is $B \to \lambda@_2$. If this rule is successfully applied, then the string is sent to membrane 2, where $A \to \lambda@_1$ can be used at this step (again, if we apply the other applicable rule $X \to @_0\lambda$, then the string will not be part of the generated language). After applying this rule, the string arrives at membrane 1. If these steps cannot be performed after each other, then the computation halts and no terminal string is generated.

3. $r = \$$. In this case there are two possibilities: by applying the rule $E_{n+1} \to (E_{n+1}, 0)@_5$ we start the move of \$ from the end of the string to its beginning (a rotation), or by using $\$ \to \lambda@_2$ we finish the computation in two steps. In the latter case symbol \$ is erased and the string is sent to membrane 2, where X is erased and the string is sent outside the system.

Let us explain in more details how the symbols are rotated. Suppose that at some computation step string $X\alpha\$z'(E_i, 0)$ can be found in membrane 3. Then only rule $X \to @_4(X_j, 0)$ can be applied at this step and after the application the string $(X_j, 0)\alpha\$z'(E_i, 0)$ is sent to membrane 4. In this membrane the only rule applicable at this moment is $(E_i, j) \to (E_i, j+1)@_3$, then the obtained string $(X_j, 0)\alpha\$z'(E_i, 1)$ returns to membrane 3 where we increment the counter in $(X_j, 0)$. After that, the string $(X_j, 1)\alpha\$z'(E_i, 1)$ returns to membrane 4. Repeating the procedure, the counter is incremented. After some steps the following three cases can occur:

1. *Case* $i > j$. When $(X_j, j)\alpha\$z'(E_i, j)$ is sent to membrane 3, then the computation stops without generating a string because its rule $(X_j, i) \to @_4(X_j, i+1)$ can be applied only if $i < j$.

2. *Case* $i < j$. When the string $(X_j, i)\alpha\$z'(E_i, i)$ reaches membrane 4, then symbol

(E_i, i) is erased by using rule $(E_i, i) \to \lambda@_5$ and the string $(X_j, i)\alpha\$z'$ is sent to membrane 5. Here no rule can be applied because $j \neq i$ and thus the computation aborts.

3. *Case* $i = j$. At some moment in membrane 4 a string of the form $(X_i, i)\alpha\$z'(E_i, i)$ can be found. Then only rule $(E_i, i) \to \lambda@_5$ can be applied which erases the rightmost symbol and sends the string $(X_i, i)\alpha\$z'$ to membrane 5. In this membrane, by applying the rule $(X_i, i) \to @_1 XE_i$, XE_i is appended to the left end of the string (corresponding to the previously erased symbol (E_i, i)) and then $XE_i\alpha\$z'$ is sent to membrane 1. Thus, the rotation of symbol E_i is completed.

The process can be iterated. When a string $X\alpha, \alpha \in (N' \cup T)^*$, is sent to membrane 2 (after using the rule $\$ \to \lambda@_2$ on the string $X\alpha\$$ in membrane 1), then X is erased and the string exits the system. If such string is a terminal one, that is $\alpha \in T^*$, then it will be a member of the generated language. Hence, $L(\Pi) = L(G)$. □

The next statement demonstrates that by using both classical evolution rules and pre-dynamical rules, a smaller number of membranes is needed to obtain the computational completeness.

Theorem 2 $EGemP_5(MPri, n(in/out)) = EGemP_*(MPri, n(in/out)) = RE$.

Proof. Analogously to the previous statement, by [2] we should prove only the inclusion $RE \subseteq EGemP_5(MPri, n(in/out))$. To this aim, we modify the construction used in the proof of Theorem 1, that is, for any phrase-structure grammar $G = (N, T, S, P)$, given in the Geffert normal form, we define a simulating extended gemmating P system consisting of five membranes, where both classical evolution rules (with target *here* only) and pre-dynamical rules can be present and also the meta-priority relation is defined.

We construct the simulating extended gemmating P system of degree 5 as follows:

$$\Pi = (V, T, \mu, M_0, \ldots, M_4, (C_0, D_0), (C_1, D_1), \ldots, (C_4, D_4)),$$

where:

$$\begin{aligned}
V &= N \cup T \cup \{X, \$\} \\
&\cup \{(E_i, j) \mid E_i \in N' \cup T \cup \{\$\}, 1 \leq i \leq n+1, 0 \leq j \leq n+1\} \\
&\cup \{(X_i, j) \mid 1 \leq i \leq n+1, 0 \leq j \leq n+1\}, \\
\mu &= [_0[_1]_1[_2]_2[_3]_3[_4]_4]_0, \\
M_1 &= \{X\$S \mid S \text{ is the axiom in } G\}, \\
M_i &= \emptyset, \text{ for all } i = 0, 2, 3, 4.
\end{aligned}$$

Let Π be given with the following sets of rules:

$$\begin{aligned}
C_1 &= \{S \to (w, here) \mid S \to w \in P\}; \\
C_i &= \emptyset \text{ for } i = 0, 2, 3, 4; \\
D_0 &= \emptyset; \\
D_1 &= \{C \to \lambda@_3, \$ \to \lambda@_2\} \\
&\cup \{E_i \to (E_i, 0)@_2 \mid E_i \in N' \cup T \cup \{\$\}, 1 \leq i \leq n+1\}; \\
D_2 &= \{X \to @_3(X_i, 0) \mid 1 \leq i \leq n+1\} \\
&\cup \{(X_i, j) \to @_3(X_i, j+1) \mid 0 \leq j < i \leq n+1\}
\end{aligned}$$

$$
\begin{aligned}
&\cup \quad \{X \to @_0\lambda\};\\
D_3 &= \{(E_i, j) \to (E_i, j+1)@_2 \mid 0 \le j < i \le n+1\}\\
&\cup \quad \{(E_i, i) \to \lambda@_4 \mid 1 \le i \le n+1\} \cup \{B \to \lambda@_4\};\\
D_4 &= \{(X_i, i) \to @_1 X E_i \mid 1 \le i \le n+1\} \cup \{A \to \lambda@_1\}.
\end{aligned}
$$

Symbols $X, (X_i, j), (E_i, j)$, and \$, as in the previous proof, are auxiliary symbols used in the rotation of the symbols, \$ is the marker symbol indicating the beginning of the simulated string of G. Analogously, let us denote the elements in $(\{A, B, C\} \cup T)$ by $E_1, \ldots, E_n$, $n \ge 1$, and let E_{n+1} be a notation for \$.

Now we explain how the system Π works and how the strings generated by G are generated by Π.

It is easy to see that in the first phase of the functioning of Π, membrane 1 uses its classical evolution rules and generates a word of the form $X\$\alpha$, where $\alpha \in (N' \cup T)^*$. During these steps, no action is performed in the other membranes. Then, a second phase of the functioning follows, with the following possibilities:

1. If the rightmost symbol of α is C, then the simulation of the rule $ABC \to \lambda \in P$ can start in membrane 1, by applying the pre-dynamical rule $C \to \lambda@_3$. Then C is deleted from the right-end of α and the string is forwarded to membrane 3. In this membrane the only applicable rule is $B \to \lambda@_4$, which deletes the rightmost symbol, B, of the string. Then, the new string is sent to membrane 4, where $A \to \lambda@_1$ can be applied: after removing symbol A from its right-end, the string returns to membrane 1 and the process can be repeated. If any of these steps fails, then the P system halts without a terminal word as an output. Note that in membrane 2, at any time, we can also use the rule $X \to @_0\lambda$ which sends the string outside the skin membrane; anyway, if the string is not consisting of terminal symbols only, then it will not contribute to the generated language.

2. Irrespectively from whether or not the rightmost symbol of α is C, a rotation of a symbol from $(\{A, B, C, \$\} \cup T)$ can start in membrane 1, by applying a rule $E_i \to (E_i, 0)@_2$, for $1 \le i \le n+1$. Then, exactly in the way as in the previous proof, by the interplay of membranes 2, 3, and 4, symbol E_i is rotated. Notice that the procedures of deleting substring ABC, from the right-end of the string and the rotation of symbols E_i do not interfere each other.

Combining the two procedures, that is, deleting substring ABC from the right-end of the string and rotating the symbols, either we obtain a string of the form $Xw\$$ with $w \in T^*$ in membrane 1 or the system halts without a terminal output. If the first case holds, then the string is forwarded to membrane 2, and after then outside the system, and the successful computation ends.

By the argumentation above, we can easily see that $L(\Pi) = L(G)$ holds. □

4 Conclusions and Open Problems

In the previous section we improved the known size bounds concerning two types of extended gemmating P systems. It is an open question whether or not these bounds are sharp. Moreover, it would be interesting to give sharp bounds on the number of membranes in extended gemmating P systems determining the class of matrix languages, the class of ET0L languages, that is, proper subclasses of the recursively enumerable language class. Similarly, we can ask what can we say about the size and the power of non-extended

gemmating P systems with a restricted number of objects (alphabet of two symbols, for example), or what can we say about the role of splitting and replication features?

Acknowledgements. Research supported in part by Project "MolCoNet" – A thematic network on Molecular Computing, European Commission, Information Society Technologies Programme, IST 2001-32008, and Hungarian Scientific Research Fund "OTKA" Grant. No T 042529.

References

[1] D. Besozzi, *Computational and modelling power of P systems*. PhD dissertation, Università degli Studi di Milano, 2003.

[2] D. Besozzi, G. Mauri, Gh. Păun, C. Zandron, Gemmating P systems: collapsing hierarchies. *Theoretical Computer Science*, 296, 2 (2003), 253–267.

[3] D. Besozzi, C. Zandron, G. Mauri, N. Sabadini, P systems with gemmation of mobile membranes. In A. Restivo, S. Ronchi Della Rocca, L. Roversi, Eds., *Proceedings of the 7th Italian Conference of Theoretical Computer Science* 2001, volume 2202 of *Lecture Notes in Computer Science*, pages 136–153, Springer, Berlin, 2001.

[4] Gh. Păun. *Membrane Computing. An Introduction*. Springer, Berlin, 2002.

[5] G. Rozenberg, A. Salomaa (Eds.), *Handbook of Formal Languages*, Springer, Berlin, 1997.