Membrane Operations in P Systems with Active Membranes

Artiom ALHAZOV[1,2], **Tseren-Onolt ISHDORJ**[1]

[1] Research Group on Mathematical Linguistics
Rovira i Virgili University
Pl. Imperial Tárraco 1, 43005 Tarragona, Spain
E-mail: {artiome.alhazov,tserenonolt.ishdorj}@estudiants.urv.es

[2] Institute of Mathematics and Computer Science
Academy of Science of Moldova
Str. Academiei 5, Chişinău, MD 2028, Moldova
E-mail: artiom@math.md

Abstract. In this paper we define a general class of P systems covering some biological operations with membranes, including evolution, communication, modifying the membrane structure, and we describe and formally specify some of these operations: membrane merging, membrane separation, membrane release. We also investigate a particular combination of types of rules that can be used in solving SAT problem in linear time.

1 Introduction

Such operations as membrane fusion (merging), membrane fission (budding, separation), and release of vesicle contents are well known phenomena of cell biology. Many macromolecules are too large to be transported across membranes through protein channels, that is why they are transported by means of vesicle formation. This process can transport packages of chemicals into the cell or out of the cell, then the contents of the vesicle is released, and the vesicle fuses with the cell membrane.

Informally speaking, in P systems with active membranes without polarizations one uses six types of rules: (a_0) multiset rewriting rules, (b_0) rules for introducing objects into membranes, (c_0) rules for sending objects out of membranes, (d_0) rules for dissolving membranes, (e_0) rules for dividing elementary membranes, and (f_0) rules for dividing non-elementary membranes, see [1]. In these rules, a single object participates during the process. We introduce here some further types of rules: (g_0) membrane merging rules, (h_0) membrane separation rules, and (i_0) membrane release rules, all these in the framework of P systems with active membranes. The common feature of those rules is the transport of multisets of objects among regions of the system.

Some operations with membranes, different from those of dissolution and division, basically considered in membrane computing, were also introduced in [3] and [2].

In P systems, exponential workspace is obtained by using membrane division, membrane creation, and replicating strings. We will see an interesting new way for obtaining exponential workspace in a linear time, by using the membrane separation. Some particular combinations of types of rules in P systems with active membranes can solve hard problems, typically **NP**-complete problems, in linear time. This possibility is illustrated here with `SAT` problem.

2 Operations on Membranes without Polarization

The reader is assumed familiar with fundamentals of membrane computing, e.g., from [4]; details can be also found at `http:// psystems.disco.unimib.it`.

We are considering a P system defined as $\Pi = (O, H, \mu, w_1, \ldots, w_m, R)$, where

1. $m \geq 1$ is the initial degree of the system;
2. O is the alphabet of *objects*;
3. H is a finite set of *labels* for membranes;
4. μ is a *membrane structure*, consisting of m membranes, labeled (not necessarily in a one-to-one manner) with elements of H;
5. $w_1, \ldots, w_m$ are strings over O, describing the *multisets of objects* placed in the m regions of μ;
6. R is a finite set of *developmental rules*, of the following forms:

   (a_0) $[\ a \to v]_h$, for $h \in H, a \in O, v \in O^*$
   (object evolution rules, associated with membranes and depending on the label, but not directly involving the membranes, in the sense that the membranes are neither taking part in the application of these rules nor are they modified by them);

   (b_0) $a[\ \ ]_h \to [\ b]_h$, for $h \in H, a, b \in O$
   (communication rules; an object is introduced in the membrane during this process);

   (c_0) $[\ a\ ]_h \to [\ \ ]_h b$, for $h \in H, a, b \in O$
   (communication rules; an objects sent out of the membrane during this process);

   (d_0) $[\ a\ ]_h \to b$, for $h \in H, a, b \in O$
   (dissolving rules; in reaction with an object, a membrane can be dissolved, while the object specified in the rule can be modified);

   (e_0) $[\ a]_h \to [\ b]_h [\ c]_h$, for $h \in H, a, b, c \in O$
   (division rules for elementary membranes; in reaction with an object, the membrane is divided into two membranes with the same label; the object specified in the rule is replaced in the two new membranes by possibly new objects);

   (g_0) $[\ ]_h [\ ]_h \to [\ ]_h$, for $h \in H$
   (merging rules for elementary membranes; in reaction of two membranes, they are merged into a single membrane; the objects of the former membranes are put together in the new membrane);

(h_0) $[\,O]_h \to [\,U]_h[\,O-U]_h$, for $h \in H, U \subset O$
(separation rules for elementary membranes, with respect to a given set of objects; the membrane is separated into two membranes with the same labels; the objects from U are placed in the first membrane, those from $U - O$ are placed in the other membrane);

(i_0) $[\,[\,O]_h]_h \to [\,]_h O$, for $h \in H$
(release rule; the objects in a membrane are released out of a membrane, surrounding it, while the first membrane disappears).

The rules are applied non-deterministically, in the maximally parallel manner; among the rules of types $(b_0), \cdots, (i_0)$ at most one can be applied to each membrane at each step.

Rules of types (a_0), (b_0), (c_0), (d_0), and (e_0) were introduced in [1], without polarizations of membranes, and without the capability of changing the label of membranes they involve (or it is the case in [4] with rules of type (b), (c)). Moreover, in [1] one considers rules that can change the label of membranes, calling them of type (a_0'), (b_0'), (c_0'), and (e_0'). We follow this idea and this notation also to rules of types (g_0), (h_0): their primed versions indicate the fact that the labels can be changed. Specifically, these rules are of the following forms:

(g_0') $[\,]_{h_1}[\,]_{h_2} \to [\,]_{h_3}$, for $h_1, h_2, h_3 \in H$.
(h_0') $[\,O]_{h_1} \to [\,U]_{h_2}[\,O-U]_{h_3}$, for $h_1, h_2, h_3 \in H, U \subset O$.

In what follows we will see how a particular combination of types of rules can be used for solving `SAT` in linear time.

3 Efficiency

From [1] we know that P systems with rules of types (a_0), (b_0), (c_0), and (e_0') can solve `SAT` in linear time. In solving `SAT`, we can eliminate membrane division (e_0') at a price of using membrane separation (h_0').

Theorem 3.1 *P systems with rules of types (a_0), (b_0), (c_0), (h_0') can solve* `SAT` *in linear time in a confluent way.*

Proof. Let us consider a propositional formula in the conjunctive normal form:

$$\begin{aligned}\beta &= C_1 \wedge \cdots \wedge C_m,\\ C_i &= y_{i,1} \vee \cdots \vee y_{i,l_i},\ 1 \le i \le m, \text{ where}\\ y_{i,k} &\in \{x_j, \neg x_j \mid 1 \le j \le n\},\ 1 \le i \le m, 1 \le k \le l_i.\end{aligned}$$

The instance β of `SAT` will be encoded in the rules of the P system by multisets v_j and v_j' of symbols, corresponding to the clauses satisfied by *true* and *false* assignment of x_j, respectively:

$$\begin{aligned}v_j &= \{c_i \mid x_j \in \{y_{i,k} \mid 1 \le k \le l_i\}, 1 \le i \le m\}, 1 \le j \le n,\\ v_j' &= \{c_i \mid \neg x_j \in \{y_{i,k} \mid 1 \le k \le l_i\}, 1 \le i \le m\}, 1 \le j \le n.\end{aligned}$$

We construct the P system

$$\Pi = (O, H, \mu, w_0, w_1, w_s, R), \text{ with}$$

$$
\begin{aligned}
O &= \{d_i, d_i' \mid 0 \le i \le n+m+5\} \\
&\cup \{t_{i,j}, t_{i,j}', f_{i,j}, f_{i,j}' \mid 1 \le i \le j \le n\} \\
&\cup \{c_i \mid 1 \le i \le m\} \cup \{t, yes, no\}, \\
\mu &= [\,[\,]_0[\,]_1]_s, \\
w_s &= \lambda, \\
w_0 &= d_0, \\
w_1 &= d_0, \\
H &= \{s, 0, 1, \cdots, m+2\},
\end{aligned}
$$

and the following rules (we accompany them with explanations about their use):

- Generation phase:

G1. $[\ d_i \to d_{i+1}t_{i+1,i+1}d'_{i+1,i+1}f'_{i+1,i+1}]_1, 0 \le i \le n-1,$

G2. $[\ d'_i \to d_{i+1}t_{i+1,i+1}d'_{i+1,i+1}f'_{i+1,i+1}]_1, 1 \le i \le n-1,$

G3. $[\ O]_1 \to [\ U]_1[\ O-U]_1,$
where $U = \{d'_i \mid 1 \le i \le n\} \cup \{t'_{i,j}, f'_{i,j} \mid 1 \le i \le j \le n\},$

G4. $[\ t_{i,j} \to t_{i,j+1}t'_{i,j+1}]_1,$
$[\ f_{i,j} \to f_{i,j+1}f'_{i,j+1}]_1,$
$[\ t'_{i,j} \to t_{i,j+1}t'_{i,j+1}]_1,$
$[\ f'_{i,j} \to f_{i,j+1}f'_{i,j+1}]_1, 1 \le i \le j \le n.$

In n steps, 2^n membranes with label 1 are created, corresponding to all possible 2^n truth assignments of the variables $x_1, x_2, \cdots, x_n$. During this process, objects $t_{i,j}, t'_{i,j}$ correspond to the $true$ value of variables x_i, and objects $f_{i,j}, f'_{i,j}$ correspond to the $false$ value of variables x_i. These 2^n copies of membranes 1 are placed in the skin membrane (the system always has only two levels of membranes).

G5. $[\ t_{i,n} \to v_i]_1,$
$[\ f_{i,n} \to v'_i]_1,$
$[\ t'_{i,n} \to v_i]_1,$
$[\ f'_{i,n} \to v'_i]_1, 1 \le i \le n.$

Every object $t_{i,j}, t'_{i,j}, f_{i,j}, f'_{i,j}$ evolves to $t_{i,n}, t'_{i,n}, f_{i,n}, f'_{i,n}$, respectively. Then these objects evolve into objects c_i, corresponding to clauses C_i, satisfied by the $true$ or $false$ values chosen for x_i.

- Checking phase:

C1. $[\ O]_i \to [\ U_i]_i[\ O-U_i]_{i+1},$
where $U_i = \{c_i\}, 1 \le i \le m.$

Next, starting with $i=1$, in membranes with label i, objects c_i will be separated from the other objects, and the label of the membrane with objects $O - \{c_i\}$ will become $i+1$. The membranes which do not contain objects c_{i+1} will never evolve anymore. If all objects $c_i, 1 \le i \le m$, are present in some membrane, then after m steps this membrane will evolve into a membrane with label $m+1$, containing objects d_n, d'_n, by the rules C1.

C2. $[\ d'_n \to d_n]_{m+1},$

C3. $[\ d_n]_{m+1} \rightarrow [\ ]_{m+1} d_n$,
C4. $[\ d_n \rightarrow tt]_s$.

If β has solutions, then at step $n+m+1$, every membrane corresponding to a solution of β ejects d_n in the skin region, then they will all be rewritten into tt.

C5. $t[\ ]_0 \rightarrow [\ t]_0$,
C6. $t[\ ]_{m+1} \rightarrow [\ t]_{m+1}$,
C7. $[\ O]_0 \rightarrow [\ U']_{m+1}[\ O-U']_{m+2}$,
where $U' = \{t\}$,
C8. $[\ d_i \rightarrow d_{i+1}]_0$,
C9. $[\ d_i \rightarrow d_{i+1}]_{m+2}, 0 \leq i \leq n+m+4$.

At step $n+m+3$, one copy of t enters the membrane with label 0, and (suppose β has s solutions, $1 \leq s \leq 2^n$) s copies of t enter the s membranes with label $m+1$, at step $n+m+4$, $s-1$ copies of t enter the membranes with label $m+1$, or $s-2$ copies of t enter the $s-2$ membranes with label $m+1$, and 1 copy of t enters the membrane with label 0. Using rule C7, membrane with label 0 is separated into two membranes, which contain object t and object d_{n+m+4}, respectively. If β has no solution, then no object enters membrane labeled by 0 and rule C7 is not applied.

- Output phase:

O1. $[\ d_{n+m+5}]_0 \rightarrow [\ ]_0 no$,
O2. $[\ d_{n+m+5}]_{m+2} \rightarrow [\ ]_{m+2} yes$,
O3. $[\ no]_s \rightarrow [\ ]_s no$,
O4. $[\ yes]_s \rightarrow [\ ]_s yes$.

If β has solutions, then at step $n+m+5$, object d_{n+m+5} in membrane with label $m+2$ eject *yes* into skin and then into the environment. It is the $(n+m+7)$th step of the computation. If β has no solution, then after $n+m+5$ steps object d_{n+m+5} ejects object *no* into skin and then into the environment. □

The following theorem shows how membrane merging (g_0) can be used instead of rules (b_0) to solve `SAT`.

Theorem 3.2 *P systems with rules of types (a_0), (c_0), (g_0), (h'_0) can solve* `SAT` *in linear time in a confluent way.*

Proof. We construct the P system

$$
\begin{aligned}
\Pi &= (O, H, \mu, w_0, w_1, w_s, R), \text{ with} \\
O &= \{d_i, d'_i \mid 0 \leq i \leq m+2n\} \\
&\cup \{t_{i,j}, t'_{i,j}, f_{i,j}, f'_{i,j} \mid 1 \leq i \leq j \leq n\} \\
&\cup \{c_i \mid 1 \leq i \leq m\} \cup \{d', t, yes, no\}, \\
\mu &= [\ [\ ]_0 [\ ]_1]_s, \\
w_s &= \lambda, \\
w_0 &= d_0, \\
w_1 &= d_0, \\
H &= \{s, 0, 1, \cdots, m+2\}.
\end{aligned}
$$

We reuse rules of the generation phase and rule C1 in Theorem 1, and we replace the remaining part of the construction with:

- Checking phase (continued):

C2. $[\ ]_{m+1}[\ ]_{m+1} \rightarrow [\ ]_{m+1}$.

Using merging rule as above, in at most n steps, all membrane corresponding to a solution of β (suppose β has s solutions, $1 \leq s \leq 2^n$) are merged into a single "solution" membrane with label $m+1$, which will contain s copies of objects d_n and d'_n.

C3. $[\ d_{m+2n} \rightarrow d_0 d']_0$,
C4. $[\ O]_0 \rightarrow [\ U'']_0 [\ O - U'']_{m+1}$,
where $U'' = \{d'\}$.

The counter object d_{m+2n} from membrane with label 0 rewritten into $d_0 d'$, and separated into two membranes with labels 0 and $m+1$, containing objects d' and d_0, respectively. By using rule C4, the latter membrane is merged with the "solution" membrane, if β has solutions.

C5. $[\ O]_{m+1} \rightarrow [\ U''']_{m+1} [\ O - U''']_{m+2}$,
where $U''' = \{d_n, d'_n\}$.

If membrane $m+1$ contains at least one object d_n or d'_n, it means there is a solution for β, and then we can separate into two membranes, with label $m+1$ which contains objects d_n, d'_n, and one with label $m+2$, which contains object d_0. The object d_0 evolves into d_1.

C6. $[\ d_0 \rightarrow d_1]_{m+1}$.

If there is no solution for β, then the merging rule C5 is not applied. In this case, rule C6 will be applied, and object d_0 evolves d_1.

- Output phase:

O1. $[\ d_1]_{m+1} \rightarrow [\ ]_{m+1} no$,
O2. $[\ d_1]_{m+2} \rightarrow [\ ]_{m+2} yes$,
O3. $[\ no]_s \rightarrow [\ ]_s no$,
O4. $[\ yes]_s \rightarrow [\ ]_s yes$.

If β has no solutions, then at step $m+2n+2$ the object d_1 from membrane with label $m+1$ ejects object *no* into skin and then into the environment. If β has solutions, then after $m+2n+4$ steps object d_1 in membrane with label $m+2$ ejects *yes* into skin and then into the environment. This is $(m+2n+6)$th step of the computation. Thus, the satisfiability problem is solved. □

Also rules for the release of vesicle contents (i_0) can be used instead of rules (c_0) in the following way.

Theorem 3.3 *P systems with rules of types* (a_0), (g_0), (h'_0), (i_0) *can solve* `SAT` *in linear time in a confluent way.*

Proof. Following the generation phase of Theorem 1, and checking phase of Theorem 2, we replace the output phase of the construction by:

- Output phase:

O1. $[\ d_1 \to d'\ no]_{m+1}$,

O2. $[\ d_1 \to d'\ yes]_{m+2}$,

O3. $[\ O]_{m+1} \to [\ U'']_{m+1}[\ O - U'']_s$,
where $U'' = \{d'\}$,

O4. $[\ O]_{m+2} \to [\ U'']_{m+2}[\ O - U'']_s$,
where $U'' = \{d'\}$,

O5. $[\ [\ O]_s]_s \to [\ ]_s O$.

If β has no solution, then the counter object d_1 in membrane with label $m+1$ is rewritten into $d'no$ and separated into two membranes, with label $m+1$, which contains object d', and one with label s, which contains object no. At the $(m+2n+4)$th step, rule O5 is applied, releasing object no into the environment. If β has solutions, then the counter object d_1 in membrane with label $m+2$ is rewritten into $d'yes$, and then separated into two membranes. Membrane with label s will contain object yes. After $m+2n+6$ steps, the object yes is released into the environment, by applying rule O5. □

4 Conclusions

We have considered several new types of rules for membrane handling: (g_0) membrane merging, (h_0) membrane separation, and (i_0) membrane release, known from cell biology.

These types of rules could be used also in neural-like networks of membranes because naturally crowded chemicals in a neuron are transmitted through an axon, and released to the cleft of the synaptic connections of neurons package by package in vesicle formation and uptake to neurons from the cleft.

The following problems are expecting future work: What is the power of P systems using particular combinations of rules of types (g_0), (h_0), and (i_0) with other rules, and primed versions of these rules? For instance, can P systems with rules $(a_0, b_0, c_0, d_0, e_0, g_0',$ $h_0, i_0)$ solve `SAT` in linear time? What are the versions of the rules of types (g_0), (h_0), and (i_0) for non-elementary membranes?

Acknowledgements. The authors acknowledge IST-2001-32008 project "MolCoNet". The first author acknowledges also the Moldovan Research and Development Association (MRDA) and the U.S. Civilian Research and Development Foundation (CRDF), Award No. MM2-3034 for providing a challenging and fruitful framework for cooperation. The second author acknowledges The State Training Fund of the Ministry of Science, Technology, Education and Culture of Mongolia.

References

[1] A. Alhazov, L. Pan, Gh. Păun, Trading Polarizations for Labels in P Systems with Active Membranes, submitted, 2003.

[2] T. Head, Aqueous Simulations of Membrane Computations, *Romanian J. of Information Science and Technology*, to appear.

[3] M. Margenstern, C. Martín-Vide, Gh. Păun, Computing with Membranes: Variants with an Enhanced Membrane Handling, In *Proc. 7th Intern. Meeting on DNA Based Computers* (N. Jonoska, N.C. Seeman, eds.), Tampa, Florida, 2001, 53–62.

[4] Gh. Păun, *Computing with Membranes: An Introduction*, Springer-Verlag, Berlin, 2002.