Volume 3, Issue 11, November 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Channel Blockers in Membrane Systems

Lamiaa Hassaan Ahmad
Modern Academy for Computer Science and Management Technology,
Cairo, Egypt

Abstract— Membrane systems are computational models that abstract the structure and functions of a living cell. A membrane system contains sets of several types of rules such as transition rules, transportation rules and membrane manipulation rules. Any membrane surrounding any component of a membrane system should contain communication channels allowing the transportation of system objects among system regions. Communication channels can be unidirectional or bidirectional. This paper is aiming at adding a new type of rules to the theory of membrane computing. The proposed type of rules is channel blocker rules that allow the blocking of a communication channel when necessary.

Keywords— Membrane systems, Channels in membrane systems, membrane manipulation in P systems, membrane structure in P systems, rules in membrane systems.

I. INTRODUCTION

Membrane systems –also called P systems- were introduced by G. Paun [1]. They are computational models that are inspired from the structure and functions of a living cell [1]. As in fig.1 [1], a membrane system consists of a number of membranes that can be nested inside each other or on any other arrangement according to the case. The chemical reactions controlling the change of molecules are represented by evolution rules –also called multiset rewriting rules [1][2] and the chemical reactions controlling transportation of molecules without changing them are represented by communication rules [1][2]. Communication rules can either be symport/antiport rules or rules with carriers [2]. The above different types of rules are employed and implemented by the P system with the target of transforming from a computational status to another.

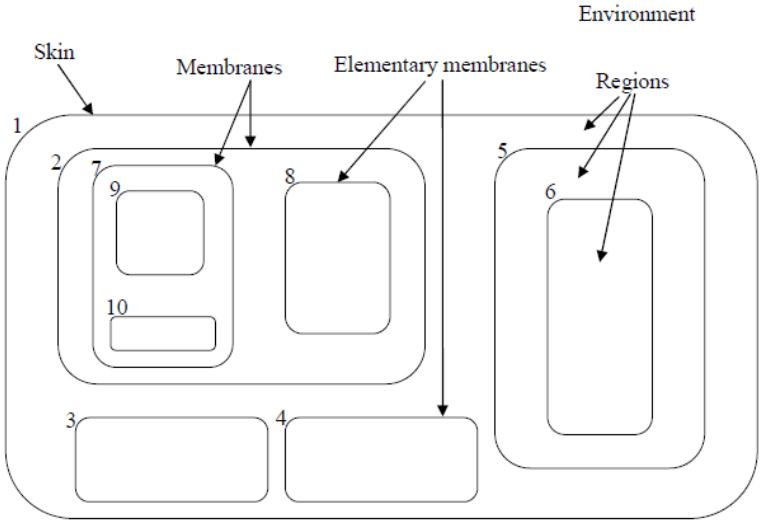

Fig. 1 A membrane system structure

The structure of a membrane system can be represented by using square brackets. Each square bracket will indicate a single membrane meaning that $[_1]_1$ symbolizes that membrane number 1 is opened and closed and it does not contain any other membranes. Thus, fig.1 can be represented by $[_1 [_2 [_7 [_9]_9 [_{10}]_{10}]_7 [_8]_8]_2 [_5 [_6]_6]_5 [_3]_3 [_4]_4]_1$.

A simple transition P system is constructed of the form [2]:

$$\Pi = (O, C, \mu, w_1, w_2, \ldots, w_n, R_1, R_2, \ldots, R_n, i_o)$$

Where:
O: The alphabet of objects, i.e. cellular molecules.
C: The alphabet of catalysts, if any.

μ: The membrane structure. It consists of n membranes labeled with 1,2,3,..,n.
w_1, w_2,…, w_n: The strings over O U C, representing the multisets of objects initially present in all regions of the system membrane structure [2].
R_1, R_2, …, R_n: The set of evolution rules associated with the regions of the system.
i_o: The output region. It will take one of the labels 1,2, ...,n.

Objects are assigned to rules by choosing rules and objects non-deterministically. Also, the chosen multiset of rules should be applicable to the chosen multiset of objects currently available. When no other rules can be applied on the current multiset of objects, the multiset of rules is said to be maximal. Different rules can be applied on different objects in parallel. We can conclude that P systems run in a maximal parallel non-deterministic manner [2].

A sequence of transitions constitutes a computation [2]. A computation is successful if it halts, reaching a configuration where no rule can be applied to the existing objects, and the output region i_o still exists in the halting configuration (in the case when i_o is the label of a membrane, it can be dissolved during the computation) [2]. The result of a successful computation can be represented in different ways. For a pre specified output region i_o, where this region is an internal one, an internal output can be defined. This means that the objects present in the output regions when the system halts are counted and this will be the result of the computation. In the case of external output, where $i_o = 0$, the objects that leave the system during the computation are counted and this will be the result of the system. In both cases the result is a number. When we differentiate objects from one another, then the final result can be a vector of natural numbers. The objects which leave the system can also be arranged in a sequence according to the moments when they exit the skin membrane, and in this case the result is a string (if several objects exit at the same time, then all their permutations are accepted as a substring of the result) [2]. When a computation does not arrive to the halting status, no output exists and therefore there is no way to know a completely computed number before halting. In the case of the dissolving of the internal output membrane during the computation, the system aborts and there is no result produced by the system. Several successful computations may be obtained by a P system. The reason is the non-determinism of the application of rules, starting from an initial configuration. Therefore, a P system computes a set of numbers, or a set of vectors of numbers, or a language, depending on the way the output is defined [2]. The case when we get a language is important in view of the qualitative difference between the "loose" data structure we use inside the system (vectors of numbers) and the data structure of the result, strings, where we also have a "syntax", a positional information [2].
When a set of numbers computed by a P system in the above way, we denote the P system by N(II). When the vector of multiplicities of objects from the output region is the main concern, we write Ps(II). In the case when the external output is the target of the P system, we denote the language of these strings by L(II).

Consider the following P system, this example from book [2]:
$II = (O, \mu, w_3, R_1, R_2, R_3, i_o)$
Where:
$O = \{a, b, c, d, e, f\}$;
$\mu = [1[2[3]3]\ 2]1$;
$w_3 = a\ f\ c$;
$R_1 = \{e \rightarrow (e, out), f \rightarrow f\}$,
$R_2 = \{b \rightarrow d, d \rightarrow de, ff \rightarrow f, cf \rightarrow cd\delta\}$,
$R_3 = \{a \rightarrow ab, a \rightarrow b\delta, f \rightarrow ff\}$
The structure of this example is shown in fig.2. from book [2].

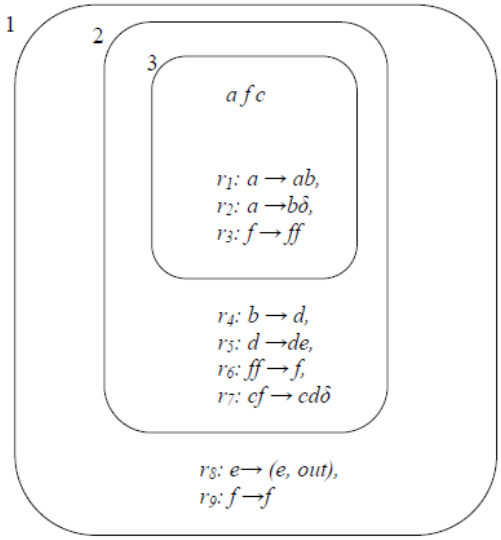

Fig. 2 An example of a membrane system

In membrane 3, there are three objects *a, f and c.* There are no other objects in the other membranes, therefore, the only membrane we can start applying rules is membrane 3. The rule r_1 can be repeated in parallel with r_3, resulting in the growth of the number of copies of *b* and the doubling of copies of *f* in each step. In case of applying r_2 in parallel with r_3 then the system will not achieve a halting status. Thus, in order to ever halt, membrane 3 must dissolve. When membrane 3 is dissolved, its contents (n + 1 copies of *b*, 2^{n+1} copies of *f* and one copy of the catalyst *c*) are left free in membrane 2, which now can start using its rules [2]. In the next step, all objects *b* become *d*. When r_7 dissolves membrane 2, the contents of this membrane are passed to membrane 1. When the objects of membrane 2 arrive to membrane 1, and there is at least one copy of *f*, then the rule r_9 from region 1 can be used forever and the computation never stops. Also, if the rule r_6 is used at least once in parallel with the rule r_7, then at least one copy of *f* is present. Therefore, the rule r_7 should be used only if region 2 contains only one copy of *f* [2]. This means that the rule r_6 was always used for all the available pairs of *f*, that is, in each step the number of copies of *f* is divided by 2. This is already done once in the step when all copies of *b* become *d*, and will be done from now on as long as at least two copies of *f* are present. Also, in each step each *d* causes the production one copy of *e*. This process can continue until we get a configuration with only one copy of *f* present; it will be the time to use the rule r_7 because rule r_6 is no longer applicable, and membrane 2 is also dissolved. The parallel application of rule r_5 for all copies of *d* (there are n + 1 such copies), during n + 1 steps, we have (n + 1)(n + 1) copies of *e*, n + 2 copies of *d* (one of them was produced by the rule r_7), and one copy of *c* present in the skin membrane of the system [2]. The objects *e* are sent out, and the computation halts. Therefore, we compute in this way the number $(n + 1)^2$, for some $n \geq 0$, that is, $N(\Pi) = \{n^2 \mid n \geq 1\}$ [2].

II. COMMUNICATION CHANNELS IN MEMBRANE SYSTEMS

In a living cell, cellular substances could move from a compartment to another under certain conditions. Changing places of molecules which is achieved through the communication channels[2]. There are some benefits, stated in [2], of changing the places of objects rather than changing them:

1- These communication rules will correspond directly to well-known chemical processes.
2- The system will preserve the conservation law.
3- The communication framework is powerful.
4- Objects can be brought into the system from the environment.

There are two famous types of communication rules which are symport/antiport rules and rules with carriers. *symport* represents the passage of two molecules through a channel in the same direction, while *antiport* represents the passage of two molecules through a channel in opposite directions [2]. In a more formal way, the rules *(xy, in)* or *(xy, out)* are symport rules, where *x* and *y* pass together through a membrane while the rule *(x, out; y, in)* is an antiport rule, where *x* goes out at the same time *y* goes in the same membrane. Neither *x* nor *y* can pass the membrane alone unless there is a rule *(x, in)* or *(x, out)* which is called *uniport* rule [2]. Symport/antiport rules can use the target indications *here, in, out* [2]. These systems are called *evolution-communication* P systems [2]. Another way of communication in P systems is carrier rules. Objects can be carried from a region to another by specific type of objects called *carriers*. This will make the carried objects called *passengers*. In a living cell, carrier proteins carry molecules to help them passing through a membrane to another. In P systems, a specific type of rules will attach passengers to carriers, move them through membranes and disassembles them by releasing the carriers and passengers available for further attachments and moves [2].

III. CHANGING THE MEMBRANE STRUCTURE

Membrane *creation* has been investigated where rules can have the form $a \rightarrow [_h v]_h$, which specify that an object of the alphabet can produce a new membrane labeled *h* and in this process it is eventually translated into a multiset of objects *v* over the same alphabet [3].

Membrane *merging* can be obtained by rules of the form $[_{h1} a]_{h1} [_{h2} b]_{h2} \rightarrow [_{h3} c]_{h3}$ where *a, b, c* elements of the alphabet *O* , while h_1, h_2, h_3 are membrane labels. Note that, in this general case, the fusion of two membranes originates a new one and the content of the membrane is changed as well.

The reverse of merging is the operation of *separation*: given a set of objects *K* which is a subset of *O*, a separation of objects in *K* has the form $[_{h1}]_{h1} [_{h2} K]_{h2} \rightarrow [_{h3} \neg K]_{h3}$. The meaning of the operation is to separate into two different membranes (i.e. membrane h_2 and h_3) the content of membrane h_1 in such a way to put in membrane h_2 all those elements belonging to the set *K* while in h_3 all the rest[3].

Two other operations that can accompany the P systems are *endocytosis* and *exocytosis*. The *endocytosis* operation can be described and formed using the rules in the form $[_{h1} a]_{h1} [_{h2}]_{h2} \rightarrow [_{h2} [_{h1} b]_{h1}]_{h2}$ for h_1*;* h_2 membrane labels and *a, b* elements of the alphabet. According to *endocytosis* rules, an elementary membrane h_1 enters membrane h_2 under the control of the object *a*, that is possibly changed, whilst membranes remain unaltered [3]. On the other hand, *exocytosis* rules are described and formed using the rule $[_{h2} [_{h1} a]_{h1}]_{h2} \rightarrow [_{h1} b]_{h1} [_{h2}]_{h2}$, where, h_1 and h_2 are membrane labels while *a, b* are objects of the alphabet [3]. In the *exocytosis* process, an elementary membrane h_1 is sent out from a membrane h_2 in a way mediated by the element *a* that can possibly be changed by the process.

In communication, there is the *gemmation* operation which is used as a particular form of communication. *Gemmation* rules have the form $a \rightarrow [_{@h2} u]_{@h2}$ where *a* is an object and *u* is a multiset of objects. The rule means that a particular membrane, labeled $@h_2$, is created, starting from an object *a*, and it enters membrane h_2 then, it is dissolved and the

multiset *u* is released inside h_2 [2]. Objects are treated as atomic entities in all the cases above. There is another possible set of extensions to P systems deals with *structured objects*.

IV. BIOLOGICAL BACKGROUND ON CHANNEL BLOCKERS

In a living cell, chemical substances and molecules should be always maintained in a certain balance in order not to cause cellular failure. An example of keeping the cell in a good shape is the Calcium Channel Blocker (CCB). In the body's tissues, the concentration of calcium ions (Ca2+) outside of cells is normally about ten thousand times higher than the concentration inside of cells. Embedded in the membrane of some cells are calcium channels. When these cells receive a certain signal, the channels open, letting calcium rush into the cell. The resulting increase in intracellular calcium has different effects in different types of cells. Calcium channel blockers prevent or reduce the opening of these channels and thereby reduce these effects [5]. Fig.3 illustrates the mechanism of CCB [6].

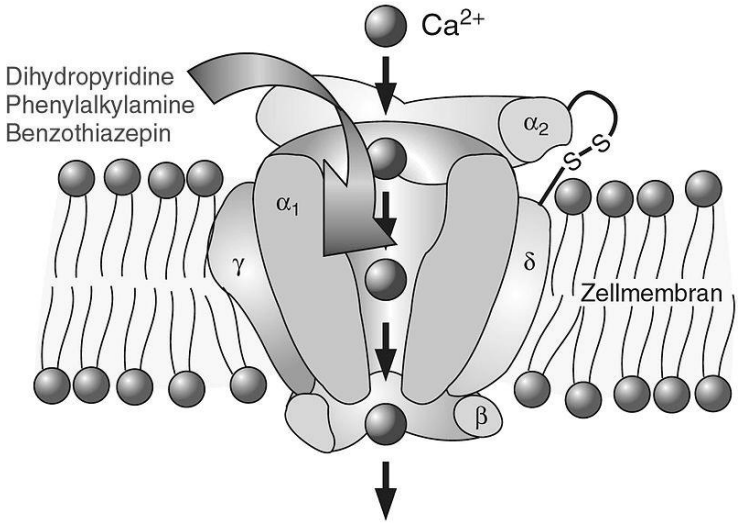

Fig. 3: Calcium channel within cellular membrane

V. CHANNEL BLOCKERS IN MEMBRANE SYSTEMS

In membrane systems, a computation is achieved by counting the number of the objects of interest at the moment. In order to achieve accurate results, some experiments tend to deal with one variable at a time. Meaning, when an object *x* is exponentially produced while running a membrane model, and when another object *y* is the one of interest, it is easier to stop producing object *x*. One way is to block the channel transporting the unneeded object. The membrane construct would be:

$$\Pi = (O, \mu, CH_{n,m}, w_1, w_2, \ldots, w_n, R_{e1}, R_{e2}, \ldots, R_{en}, R_{c1}, R_{c2}, \ldots, R_{cn}, R_{b1,m}, R_{b2,m}, \ldots, R_{bn,m}, i_o)$$

Where:

O: The alphabet of objects, i.e. cellular molecules.
μ: The membrane structure. It consists of n membranes labelled with 1,2,3,..,n.
$CH_{n,m}$: The formed number of *m* channels within membrane *n*, [4]
$w_1, w_2, \ldots, w_n$: The strings over O U C, representing the multisets of objects initially present in all regions of the system membrane structure [2],
$R_{e1}, R_{e2}, \ldots, R_{en}$: The set of evolution rules associated with the regions of the system,
$R_{c1}, R_{c2}, \ldots, R_{cn}$: The set of channel creation rules associated with membrane *n*, [4]
$R_{b1,m}, R_{b2,m}, \ldots, R_{bn,m}$: The set of blocker rules which block channel *m* within membrane *n*,
i_o: The output region. It will take one of the labels 1,2, ...,n.

The blocker rules will be in the form: $R_{bn,m}$: $CH_{n,m} \rightarrow CH_{n,\infty}$
Where $R_{bn,m}$ is the blocker rule that causes the blockage of channel *m* within membrane *n* by transforming *m* into ∞. This means that channel *m* is currently unavailable for transportation.

What about unblocking the channel? Sure it is convenient with the biological and logical points of view. We can add another set of rules, called unblocking rules $R_{u1,m}, R_{u2,m}, \ldots, R_{un,m}$. The job of these rules is to unblock the blocked channels by transforming ∞ into *m* as follows:

$R_{un,m}$: $CH_{n,\infty} \rightarrow CH_{n,m}$

We can observe the following:

- Multiple channels within a single membrane may be blocked. The reason is that all types of rules are executes in a maximally parallel manner.
- One or more channels may be blocked because we are concentrating on a certain object or we are attempting to save the system from a critical status such as deadlocks.
- In addition to membrane dissolving, merging, division and channel formation rules, a new type of rules, which is channel blocker rules, is capable of manipulating the membrane structure itself.

VI. CONCLUSIONS

Membrane systems are computational models that have been proposed 1998. The power of membrane systems has been investigated in different research fields such as data encryption, image processing and evolutionary algorithms. This paper is intended as a representation of a membrane system with channel blocker rules. One of the observations was that one channel or more were blocked simultaneously in the same membrane according the current number of cellular molecules. Also, channel blocker rules could result in causing the modification of the membrane structure itself. Some serious problems –such as deadlocks- can be avoided or overcame using the channel blocker rules. The proposed work above might be both a useful contribution and an efficient addition to the field of membrane systems.

REFERENCES

[1] G. Paun, *Computing with membranes*, Journal of Computer and System Sciences, 2000, pp. 108-143.
[2] G. Paun, *Membrane computing. An introduction*, Springer-Verlag, 2002.
[3] G. Paun, C. Martin-Vide and A. Rodriguez-Paton, *On P systems with membrane creation*, Computer Science J. of Moldova, 9, 134—145, 2001.
[4] L. H. Ahmed, A. Badr, I. Farag, *Protein Channel Formation in P Systems* " STUDIA UNIV. BABES_{BOLYAI, INFORMATICA, Volume LVI, Number 4, 2011
[5] I. Bezprozvanny , Tsien RW, *Voltage-Dependent Blockade of Diverse Types of Voltage-Gated Ca^{2+} Channels Expressed in Xenopus Oocytes by the Ca^{2+} Channel Antagonist Mibefradil (Ro 40-5967)* . Molecular Pharmacology **48** (3): 540–549, 1995.
[6] http://en.wikipedia.org/wiki/File:Calciumkanal_F%C3%B6rstermann.jpg

