Simulation of Mobile Ambients by P Systems. Part 2

Vladimir ROGOZHIN
The State University of Moldova
60 Mateevich str., MD-2009 Chişinău, Moldova
E-mail: rv@math.md

Elena BOIAN
Institute of Mathematics and Computer Science
Academy of Sciences of Moldova
5 Academiei str., MD-2028, Chişinău, Moldova
E-mail: lena@math.md

Abstract. Ambient calculus is a theory which deals with mobile computing and computation and encompasses such notions as mobile agents, the ambients where the agents interact and the mobility of the ambients themselves. P systems is a formalism which abstracts from the structure and functioning of living cells and describes distributed parallel computing devices with multiset of objects processing. Ambient calculus and membrane computing are based on the same concepts and structures though they are developed in different areas of computer science. The purpose of our work now is to express ambient calculus by means of P systems, namely by tissue P systems with dynamic network of membranes.

1 Introduction

This paper is organized as follows: firstly, basic concepts and ideas of ambient calculus [2] will be described. After that we will give the notion of dynamic networks and will introduce the notion of tissue P systems with dynamic networks of membranes. We will give some ideas how to express systems of mobile ambients using this type of P systems with *tree* topology.

1.1 Ambient Calculus

An ambient is a named bounded place where computation happens. Ambients form systems with hierarchical nested structure. Ambients are allowed to move inside neighbour ambients and to move outside parent ambients. The movement of ambients is controlled by the computation which happens inside them. Furthermore, the process which is running inside an ambient can open (dissolve) the ambient's subambient.

There is a security feature in ambient calculus: the access to an ambient by a computational process is authorized. The authorization is performed using the name of an ambient.

If an ambient with the name to which the process appeals to does not exist, then the process is blocked until an ambient with this name appears in the area of activity of the process.

There are three mobility primitives in ambient calculus:

- *open* n opens the ambient named n. The operation will be blocked until a subambient inside the surrounding ambient with this name appears.
- *in* n makes the enclosing ambient with whole its content to move inside the neighbouring ambient named n. The operation will be blocked until a subambient inside the surrounding ambient with this name appears.
- *out* n makes the enclosing ambient to exit out of its parent ambient named n. The operation will be blocked until the name of the parent ambient of the enclosing ambient will be n.

The following primitives describe the structure of a process in Ambient Calculus:

restriction Expression $(\nu n)P$ means creation of the name n restricted to the scope P.

inactivity Expression 0 denotes the process which does not perform any action.

parallel composition Expression $P|Q$ denotes processes P and Q running in parallel.

replication Expression $!P$ means unbounded replication of P. $!P \Leftrightarrow P = P|!P$.

It is proved that ambient calculus is Turing-complete [2].

1.2 Membrane Computing and Ambient Calculus

Although membrane computing [6] and ambient calculus belong to different branches of computer science, they are based on the same concepts and ideas. The structure of traditional P systems [6] and of mobile ambients is hierarchical. The computation has a high level of parallelism and is distributed over membranes/ambients. The notions of a membrane and of an ambient are similar. The computation in ambient calculus is presented by computational processes which are running inside ambients, but in P systems by the process of application of communication and evolution rules.

In [8] it has been shown that P systems can be expressed in the frame terms of ambient calculus.

2 Tissue-like P Systems with Dynamic Network of Membranes

2.1 Dynamic Networks

We assume we have a complex computer network (as an example we can take Internet) where the possibility to create and remove communication channels or to move communicating nodes exists. We assume that nodes can communicate even during the time of their motion through the network. We call such a type of networks *dynamic networks*.

All problems typical for static networks [5] are typical for dynamic networks, too. Some important issues in this framework are:

- distribution of network addresses,
- addressing of nodes,
- routing of the messages in the network.

The permanent changing of network's structure makes these problems more difficult.

When, for example, a channel has been removed or created, or a node has been moved, then the set of nodes to which the node has direct connection changes, the subnetwork to which the node belongs changes. If we use structured addresses, then in dependence of the strategy of distributing the addresses when changing the network structure we have to redistribute the network addresses between nodes again.

There are at least two reasons which make the routing problem more difficult in dynamic networks: when the structure of the network changes, the necessity to choose the new path for the routing message may appear and even the necessity to change destination network address of the message being sent may appear because of the redistributing network addresses between the nodes.

There are a lot of architectures of dynamic networks: with dynamic and static parts (for example, see [1]), that is when a part of network can be changed (i.e., communication channels and network nodes can be created and removed) and another part remains unchanged. As an example of such networks we can consider the network of cellular telephones, or today's Internet with mobile computing devices – laptops and others. These are networks with mobile nodes-leaves but with static routers (or gateways).

However, one can consider dynamic networks where all nodes can be moved.

A lot of strategies can be used to solve the problems of distribution of addresses and of routing in dynamic networks.

We will call *dynamic data transmission network* a construct of communicating nodes and communication channels which can be presented as a graph where channels can be created and removed, nodes can be created and removed in *runtime*.

In the model we consider below we will abstract from the strategies of distribution of addresses, of addressing and of routing.

2.2 P Systems with Dynamic Networks of Membranes

We define the type of tissue P systems [3] where the system of membranes represents the dynamic network of membranes and channels.

Definition: A P system with a dynamic network of membranes is a construct

$$\Pi = (O, A, T, S_A, M, \mu, i_0),$$

where:

– O is an alphabet of symbol-objects of the system. It is a finite non-empty set.

– A is the set of valid *names* of membranes. Names can be of arbitrary nature. The set A can be finite or infinite.

– T is the set of types (classes) of membranes. One type of membranes represents the set of communication/evolution rules and priorities among the rules. We will write $T = \{T_1, T_2, \ldots, T_t\}$, where $T_1, \ldots, T_t$ are types of membranes. Type T_i

we will define as $T_i = \{r_1^{(i)}, r_2^{(i)}, \ldots, r_l^{(i)}, p_1^{(i)}, p_2^{(i)}, \ldots, p_m^{(i)}\}$. $r_1^{(i)}, r_2^{(i)}, \ldots, r_l^{(i)}$ are communication/evolution rules. $p_1^{(i)}, p_2^{(i)}, \ldots, p_m^{(i)}$ are priorities between the rules $r_1^{(i)}, r_2^{(i)}, \ldots, r_l^{(i)}$ [7].

– $S_A \subseteq O \times A$ is the list of correspondences of symbol-objects from O and the names A of membranes. We will write $S_A = \{(s_1, a_1), (s_2, a_2), \ldots, (s_k, a_k)\}$, where $s_i \in O$, $a_j \in A$. A pair (s_i, a_i) means that to membrane with the name a_i symbol-object s_i is assigned. The meaning of this item will be explained later.

– M is the set of membranes currently presented in the system. We will write $M = \{\varpi_1, \varpi_2, \ldots, \varpi_n\}$. $\varpi_1, \varpi_2, \ldots, \varpi_n$ are membranes. A membrane $\varpi \in \{\varpi_1, \varpi_2, \ldots, \varpi_n\}$ can be denoted as $\varpi = \{a, t, \omega\}$. $a \in A$ is name of the membrane, $t \in T$ is type of the membrane, $\omega \in O^*$ is the multiset of symbol-objects presented inside the membrane.

– $\mu \subseteq M \times M$ is the set of communication channels between membranes. We denote a communication channel between membranes ϖ_i and ϖ_j by the ordered pair (ϖ_i, ϖ_j). This pair denotes that a multiset of objects can be sent directly from the membrane ϖ_i to the membrane ϖ_j, but it does not mean that objects can be sent from ϖ_j to ϖ_i.

– i_0 is optional. It is the name of the output membrane. The result of a computation is considered to be in this membrane. If we have no *output* membrane i_0, then we will consider as the result of a computation the multiset of objects which appear in the environment of the system.

The evolution/communication rules are of the general general forms

$$r : u \to v \ [_{(a_1,t_1)}\omega_1]_{(a_1,t_1)} \ [_{(a_2,t_2)}\omega_2]_{(a_2,t_2)} \ \ldots [_{(a_k,t_k)}\omega_k]_{(a_k,t_k)}\delta$$

or

$$r : u \to v \ [_{(a_1,t_1)}\omega_1]_{(a_1,t_1)} \ [_{(a_2,t_2)}\omega_2]_{(a_2,t_2)} \ \ldots [_{(a_k,t_k)}\omega_k]_{(a_k,t_k)}$$

where:

- $u \in O^+$. If $|u| = 1$, then the system is called non-cooperative, otherwise if $|u| > 1$ the system is called cooperative.

- $v \in (O_{TAR})^*$. $O_{TAR} \subseteq O \times TAR$ is the set of symbol-objects and communication (or target) commands. $TAR = \{here\} \cup (\{go\} \times A^*)$ is the set of target commands. Target command $(s, here)$ means that after the application of the rule the newly created symbol s must remain in the membrane where the rule was applied. The target command $(s, go\ path)$ means that after the application of the rule the newly created symbol s must be sent to the destination indicated by *path*. $path \in A^*$ is a string of the form $a_1 a_2 \ldots a_k$. In other words, *path* is a route from the membrane where the rule is applied to the membrane with the name a_k and which consists of the sequence of channels $(\varpi_{a_0}, \varpi_{a_1}), (\varpi_{a_1}, \varpi_{a_2}), \ldots, (\varpi_{a_{k-1}}, \varpi_{a_k})$, where $\varpi_{a_0}, \ldots, \varpi_{a_k}$ are membranes with the names $a_0, \ldots, a_k$ and ϖ_{a_0} is membrane where the rule is applied. We will say that the path (or route) *path* is valid in the *current time* if all communication channels from *path* are present in the system in *current time*, otherwise we will say that the path is invalid.

- $[_{(a,t)}\omega]_{(a,t)}$ is a membrane creation command. It means that after the application of the rule the new membrane $\varpi = \{a, t, \omega\}$ will be created. If the rule contains such kind of commands, then we call it a membrane creation rule.

- δ is the membrane dissolution command. It means that after the application of the rule the membrane where the rule was applied will be removed from the system (dissolved), together with all communication channels of the form (ϖ, ϖ_0) and (ϖ_0, ϖ). ϖ is any membrane of the system and $\varpi_0 = \{a_0, t_0, \omega_0\}$ is the membrane where the rule was applied. In other words, after the application of the rule, the membrane where the dissolution rule was applied will be removed from the system together with all channels connected to it.

The tuple (M, μ) is called the configuration of the system in the *current time*.

The set of membranes (without considering their contents) and the set of communication channels is called the *structure* of the system.

An evolution of the system is any sequence of transitions between configurations. The transitions between configurations are performed by the application of rules.

The rules which modifies the structure of the system are called structure rules.

The rules are applied *synchronously, in a non-deterministic maximally parallel manner*, as in [6].

The application of rules of the form $u \rightarrow v\ [_{(a_1,t_1)}\omega_1]_{(a_1,t_1)}\ [_{(a_2,t_2)}\omega_2]_{(a_2,t_2)}\ \ldots [_{(a_k,t_k)}\omega_k]_{(a_k,t_k)}\delta$ or $u \rightarrow v\ [_{(a_1,t_1)}\omega_1]_{(a_1,t_1)}\ [_{(a_2,t_2)}\omega_2]_{(a_2,t_2)}\ \ldots [_{(a_k,t_k)}\ \omega_k]_{(a_k,t_k)}\delta$ in all membranes is executed according to the following algorithm (all steps for each rule in all membranes are executed synchronously):

Step 0: A condition if a rule can be applied is checked.

Step 1: The multiset u is removed from the membrane where the rule is applied. The rule can be applied only in the case when the multiset u is present in the membrane. This condition is checked at Step 0.

Step 2: The multiset of objects from v is created and all objects are sent to the membranes according to the target commands from v. That is, if we have a target command $(s, here)$, then the object s is created and remains inside the membrane where the rule is being applied. If we have the target command $(s, go\ a)$, then the symbol s is sent directly to membrane ϖ_a with name a through communication channel (ϖ, ϖ_a). We call the target commands of this form *short-distance* target commands. We assume that ϖ_0 is the membrane where the rule is being applied. If the communication channel (ϖ_0, ϖ_a) does not exist, then the rule can not be applied. This condition is checked at Step 0. If we have a target command of the form $(s, go\ a_1 a_2 \ldots a_k)$, then we say that the *remote delivery procedure* starts in parallel for symbol s directed to the membrane named a_k through the sequence of membranes named $a_1, a_2, \ldots, a_{k-1}$. We call the target commands of this form *distant* (or *long-distance*) target commands. The *remote delivery procedure* is not **synchronized** with the process of application of rules. It works in parallel with the process of application of rules and the *time* when the symbol s will be delivered to the destination membrane is not determined. All this means that the execution of the algorithm can continue with the following step (Step 3) without waiting for finishing the *remote delivery procedure*.

Step 3: The membranes $\varpi_{a_1} = \{a_1, t_1, \omega_1\}, \varpi_{a_2} = \{a_2, t_2, \omega_2\}, \ldots, \varpi_{a_k} = \{a_k, t_k, \omega_k\}$ pointed in the membrane creation commands $\lfloor_{(a_1,t_1)}\omega_1\rfloor_{(a_1,t_1)}$ $\lfloor_{(a_2,t_2)}\omega_2\rfloor_{(a_2,t_2)}$- $\ldots \lfloor_{(a_k,t_k)}\ \omega_k\rfloor_{(a_k,t_k)}$ are created. If the rule is a dissolution one, then the membrane where the rule is applied is dissolved, otherwise go to the Step 0.

We call a *macrostep* of the system (or *time unit*) the sequence of execution of Steps 0–3.

We measure the time of evolution of the system in *macrosteps*.

We call *current time* (or *current macrostep*) the *macrostep* being considered.

We call *network of membranes* the set of membranes and communication channels of the system.

We call *halt*-configuration of the system the configuration in which no evolution/communication rule can be applied. We consider the result of the computation the multiset of objects presented in the membrane named i_0 or in the environment at the *halt*-configuration.

The *remote delivery procedure* delivers symbol-objects from one point of the network of membranes to another point. It works according to the algorithm given below:

Notations (we give them informally):

- *destination* – the destination membrane.
- *source* – the source membrane. It is the membrane where the target command $(s, go\ path)$ was launched.
- *current* – the membrane where the symbol being delivered is situated.
- the function $get\_next(path)$ has as the result the first membrane from the route to the membrane addressed by *path*. In *static networks* (the structure of the system does not change in time) it is $get\_next(a_1a_2\ldots) = a_1$. If the structure of the system changes during the evolution, it is not the case anymore because the path $a_1a_2\ldots$ may become invalid before the application of the function $get\_next(a_1a_2\ldots)$.
- the function $new\_path(path)$ searches for the new path from the membrane *current* to the membrane destination addressed by *path* from the membrane before *current*. That is, when the object s comes to the following membrane, the path from the membrane is searched again. If we have a static network, then $new\_path(a_1a_2\ldots) = \{a_2\ldots\}$, but it is not the case if the network is dynamic. If *path* is invalid or the membrane destination cannot be reached in the *current time*, then the function will wait until the membrane destination can be reached.

Note: in the type of P systems considered in this paper we will abstract from methods and techniques of routing in dynamic networks of membranes and we will not define algorithmic functions *get_next* and *new_path*.

Steps:
Input: $current = source$

Step 1 $current := get\_next(path)$;

Step 2 $if(current == destination) then\ STOP$;
$else\ goto\ Step\ 3$;

Step 3 $path := new\_path(path)$;

Step 4 *goto Step* 1;

Output: $current = destination$

When the *remote delivery procedure* finishes its work the symbol s will find itself inside the membrane destination. We stress that even if membrane destination does not exist on the moment of starting the *remote delivery procedure*, the procedure will start anyway, but it will be finished only when the membrane destination will appear in the system and the symbol s will be successfully delivered to it.

In this way, we have defined the *evolution* and *communications* of the system. Now we will define the operation of modification of the structure of the system (more precisely, the modification of the set of communication channels).

Assume we have membranes $\varpi_a \in M$ and $\varpi_b \in M$ in the system Π. Then, $\varpi_a = \{a, t, \omega\}$ and $\varpi_b = \{b, t', \omega'\}$ and there are no other membranes with the same name b as ϖ_b, that is, $\varpi = (b, t'', \omega'') \in M$ iff $\varpi = \varpi_b$. We have the symbol $s_b \in O$ associated to the name (not membrane!) b, that is, we have $(s_b, b) \in S_A$.

Affirmation: We have a communication channel from the membrane ϖ_a to the membrane ϖ_b $(\varpi_a, \varpi_b) \in \mu$ iff we have inside the membrane ϖ_a the symbol s_b associated to the name b of membrane ϖ_b.

Formally: If we have a P system Π with $\varpi_a = \{a, t, \omega\} \in M$, $\varpi_b = \{b, t', \omega'\} \in M$, $s_b \in O$, $(s_b, b) \in S_A$, then $(\varpi_a, \varpi_b) \in \mu \Leftrightarrow s_b \in \omega$.

If we have in Π more than one membrane with the same name b, that is there are at least two membranes $\varpi_b = \{b, t', \omega'\}$ and $\varpi'_b = \{b, t'', \omega''\}$, and we have a membrane $\varpi_a = \{a, t, \omega\}$ with symbol $s_b \in \omega$, then one of the membranes with the name b is chosen in a non-deterministic way to be connected with the membrane ϖ_a.

There are no restrictions in participating in communication/evolution rules over the objects associated to the names (that is, $s \in O | (s, a) \in S_A \& a \in A$). We define the set of $O_m \subseteq O$ by $O_m = \{s \in O | (s, a) \in S_A \& a \in A\}$.

In this way, by using evolution/communication rules one can create and remove communication channels.

Rules of the form $u \to v \ldots$ with $s \in u$ or $s \in v$ where $s \in O_m$ modify the set of communication channels and they make part of structure rules.

Rules with $s \in u$ where $s \in O_m$ are called channel removing rules.

Rules with $s \in v$ where $s \in O_m$ are called channel creation rules.

For example, assume we have a P system Π with membranes $\varpi_a = \{a, t, \omega\}$ and $\varpi_b = \{b, t', \omega'\}$, $(s_b, b) \in S_A$ and $s_b \notin \omega$. Assume $t = \{r_b : i \to s_b \ldots\}$. It is clear that after the application of the rule r_b inside membrane ϖ_a the symbol s_b and hence channel (ϖ_a, ϖ_b) will be created.

2.3 Modified Tissue-like P Systems with Dynamic Network of Membranes

Here we consider a more restricted type of tissue P systems with a dynamic network of membranes, more suitable for representing tree-like networks of membranes.

Specifically, we consider constructs of the form

$$\Pi = (O, A, T, S_P, S_C, M, \mu, i_0),$$

where:

– O, A, T, M, μ, i_0 have the same meaning as in tissue-like P systems with a dynamic network of membranes (see above).

– $S_P \in O \times A$ is the list of correspondences between object-symbols and names of membranes-parents.

– $S_C \in O \times A$ is the list of correspondences between object-symbols and names of membranes-children.

Modifications:

We describe only the modified components of this type of P systems. The rest of components is the same as in *tissue-like P systems with a dynamic network of membranes* described above.

2.3.1 Communication Channels

In this type of P systems we will use only bi-directional parent-child channels. We say that the *bi-directional parent-child channel* (ϖ_a, ϖ_b) which connects membranes $\varpi_a = \{a, t, \omega\}$ and $\varpi_b = \{b, t', \omega'\}$ allows the transmission of objects between ϖ_a and ϖ_b in both directions. That is, the *short-distance* target commands of the form $(s, go\ a)$ inside ϖ_b and viceversa $(s, go\ b)$ inside ϖ_a can be executed. We say that the ordered pair (ϖ_a, ϖ_b) denotes a *parent-child* communication channel between membranes ϖ_a and ϖ_b, where ϖ_a is parent and ϖ_b is a child (or, in other words, ϖ_a is a parent for ϖ_b and ϖ_b is a child for ϖ_a). Let us define functions *children* and *parents*: $children(\varpi) = \{\varpi_c | (\varpi, \varpi_c) \in \mu\}$ and $parents(\varpi) = \{\varpi_p | (\varpi_p, \varpi) \in \mu\}$. We impose the restriction $|parents(\varpi)| <= 1\ for\ all\text{-}\varpi \in M$, which means that every membrane can have at most one parent.

This definition of channels is very suitable for building *tree-like* networks, but it can be used only to build networks with at most one cycle. In this way, we cannot use this type of channels to build networks with an arbitrary topology.

2.3.2 Addressing

We introduce two special types of names:

1. *up*. Used for addressing the parent. According to the definition of *parent-child* channels, at most one membrane-recipient can be addressed by *up*. That is, assume we have membranes $\varpi_p = \{a, t, \omega_p\} \in M$ and $\varpi_c = \{a', t', \omega_c\}$, communication channel (ϖ_p, ϖ_c) and target command $(s, go\ up)$ being applied inside ϖ_c. Then, after the application of the command, the symbol s will be placed inside membrane ϖ_p, that is, $\omega_p = \{s\} \cup \omega_p$.

2. *all*. Used to broadcast (send simultaneously) objects to all children of the membrane. Assume we have membranes $\varpi_p = \{p, t, \omega_p\}$, $\varpi_{c1} = \{c_1, t', \omega_{c1}\}$, ..., $\varpi_{cn} = \{c_n, t^{(n)}, \omega_{cn}\}$, and *parent-child* channels $(\varpi_p, \omega_{c1}), \ldots, (\varpi_p, \omega_{cn})$. Then, after the application of the target command $(s, go\ all)$, copies of s will be found inside membranes ϖ_{c1}, ..., ϖ_{cn}.

In this way, the set of target commands will be of the form: $TAR = \{here\} \cup (\{go\} \times \{A \cup \{up\} \cup \{all\}\}^*)$.

Using names of membranes from A one can address children membranes, but addressing of a parent is not possible. That is, if we have two membranes $\varpi_p = \{a, t, \omega\}$ and $\varpi_c =$

$\{b, t', \omega'\}$, and the parent-child channel (ϖ_p, ϖ_c), then the target command $(s, go\ b)$ can be applied inside ϖ_p, but the target command $(s, go\ a)$ cannot be applied inside ϖ_c. Instead of this command one should use $(s, go\ up)$.

2.3.3 Controlling the Set of *parent-child* Channels

Assume we have membranes $\varpi_a = \{a, t, \omega\} \in M$ and $\varpi_b = \{b, t', \omega'\} \in M$. Assume $(s_a, a) \in S_P$ and $(s_b, b) \in S_C$.

Affirmation: We have the *parent-child* channel in Π $(\varpi_a, \varpi_b) \in \mu$ iff we have the parent symbol s_a associated to the name a inside membrane ϖ_b and the child symbol s_b associated to the name b inside membrane ϖ_a. Formally: If we have a P system Π with $\varpi_a = \{a, t, \omega\} \in M$, $\varpi_b = \{b, t', \omega'\} \in M$, $(s_a, a) \in S_P$, $(s_b, b) \in S_C$, then $(\varpi_a, \varpi_b) \in \mu \Leftrightarrow s_a \in \omega' \& s_b \in \omega$.

It is necessary to note that, unlike in the previous section, the communication channel is controlled by both membranes.

Like in the previous section, if we have more than one membrane with the name a and/or more than one membrane with the name b and the conditions of the existence of the communication channel are respected in some of these membranes, then the membranes to be connected are chosen in the nondeterministic way. In other words, if we have membranes $\varpi_{a1} = \{a, t_1, \omega_1\}$, $\varpi_{a2} = \{a, t_2, \omega_2\}$, ..., $\varpi_{al} = \{a, t_l, \omega_l\}$ and $\varpi_{b1} = \{b, t'_1, \omega'_1\}$, $\varpi_{b2} = \{b, t'_2, \omega'_2\}$, ..., $\varpi_{bk} = \{b, t'_k, \omega_k\}$, $s_b \in \omega_i$, $i \in \{1, \ldots, l\}$ and $s_a \in \omega'_i$, $i \in \{1, \ldots, k\}$, then $s = min(l, k)$ parent-child channels $(\varpi_{ai}, \varpi_{aj})$ are created, where $i \in \{1, \ldots, l\}$ and $j \in \{1, \ldots, k\}$ are chosen in the nondeterministic manner.

3 Expressing Ambient Calculus by Means of P Systems

In this section we will give some ideas how to express ambient calculus by means of modified tissue P systems with dynamic network of membranes.

3.1 Current Configuration

We call current configuration of mobile ambients the expression $n[P_1| \ldots |P_p|m_1[\ldots]| \ldots |m_q[\ldots]]$, where $P_1, \ldots, P_p$ are processes running in parallel inside the ambient n, $m_1, \ldots, m_q$ are subambients nested by ambient n.

We distinguish two groups of membranes in P systems:

- Group A will serve as a model of an ambient from ambient calculus.
- Group P will model a process from ambient calculus.

To represent a hierarchy of ambients we will use modified tissue P systems with a dynamic network of membranes with a tree-topology. To all membranes from the group A we will connect by *parent-child* channels all corresponding membranes from the group A and all corresponding membranes from the group P. I.e., for the ambient $n[P_1| \ldots |P_p|m_1[\ldots]| \ldots |m_q[\ldots]]$ we will create the following network of membranes:

Membranes *Group A:* $\varpi_n = \{n, a_n, \omega_n\} \in M$, $\varpi_{m1} = \{m_1, a_{m1}, \omega_{m1}\} \in M$, ..., $\varpi_{mq} = \{m_q, a_{mq}, \omega_{mq}\} \in M$;
Group P: $\varpi_{P1} = \{P_1, t_{P1}, \omega_{P1}\} \in M$, ..., $\varpi_{Pp} = \{P_p, t_{Pp}, \omega_{Pp}\} \in M$

Channels $(\varpi_n, \varpi_{P1}), \ldots, (\varpi_n, \varpi_{Pp}), (\varpi_n, \varpi_{m1}), \ldots, (\varpi_n, \varpi_{mq}) \in \mu$; $(s_n, n) \in S_P$, $(s_{P1}, P_1), \ldots, (s_{Pp}, P_p), (s_{m1}, m1), \ldots, (s_{mq}, m_q) \in S_C$; $s_{P1}, \ldots, s_{Pp}, s_{m1}, \ldots, s_{mq} \in \omega_n$, $s_n \in \omega_{P1}$, $\ldots$, $s_n \in \omega_{Pp}$, $s_n \in \omega_{m1}$, $\ldots$, $s_n \in \omega_{mq}$

3.2 Mobility

As it has been mentioned above, there are three elementary kinds of mobility of ambients:

1. move into one's neighbour
2. move out of one's parent
3. open (dissolve) an ambient

The motion of ambients through the hierarchy is presented as a commutation of communication channels.

The motion inside one's neighbour Assume we have the configuration $n[m_1[\ldots]|m_2[\ldots]|\ldots]$. Then we will have the corresponding P system with the structure $\varpi_n = \{n, t_n, \omega_n\}, \varpi_{m1} = \{m_1, t_{m1}, \omega_{m1}\}, \varpi_{m2} = \{m_2, t_{m2}, \omega_{m2}\} \in M$; $(\varpi_n, \varpi_{m1}), (\varpi_n, \varpi_{m2}) \in \mu$; $(s_n, n) \in S_P$, $(s_{m1}, m_1), (s_{m2}, m_2) \in S_C$, $(s'_{m1}, m_1) \in S_P$; $s_{m1}, s_{m2} \in \omega_n$, $s_n \in \omega_{m1}$, $s_n \in \omega_{m2}$. Assume that the ambient m_1 moves inside the ambient m_2. The new configuration will be $n[m_2[m_1[\ldots]|\ldots]|\ldots]$ and the corresponding structure of the P system $\varpi_n = \{n, t_n, \omega_n\}, \varpi_{m1} = \{m_1, t_{m1}, \omega_{m1}\}, \varpi_{m2} = \{m_2, t_{m2}, \omega_{m2}\} \in M$; $(\varpi_n, \varpi_{m2}), (\varpi_{m2}, \varpi_{m1}) \in \mu$; $s_{m2} \in \omega_n$, $s'_{m2} \in \omega_{m1}$, $s_n, s_{m1} \in \omega_{m2}$. The channel (ϖ_n, ϖ_{m1}) will be removed.

The motion outside one's parent Assume we have the configuration $n[m_2[m_1[\ldots]|\ldots]|\ldots]$. Then we will have the corresponding P system with the structure $\varpi_n = \{n, t_n, \omega_n\}, \varpi_{m1} = \{m_1, t_{m1}, \omega_{m1}\}, \varpi_{m2} = \{m_2, t_{m2}, \omega_{m2}\} \in M$; $(\varpi_n, \varpi_{m2}), (\varpi_{m2}, \varpi_{m1}) \in \mu$; $(s_n, n) \in S_P$, $(s_{m1}, m_1), (s_{m2}, m_2) \in S_C$, $(s'_{m1}, m_1) \in S_P$; $s_{m2} \in \omega_n$, $s'_{m2} \in \omega_{m1}$, $s_n, s_{m1} \in \omega_{m2}$. Assume that the ambient m_1 moves outside the ambient m_2. The new configuration will be $n[m_1[\ldots]|m_2[\ldots]|\ldots]$ and the corresponding structure of the P system $\varpi_n = \{n, t_n, \omega_n\}, \varpi_{m1} = \{m_1, t_{m1}, \omega_{m1}\}, \varpi_{m2} = \{m_2, t_{m2}, \omega_{m2}\} \in M$; $(\varpi_n, \varpi_{m1}), (\varpi_n, \varpi_{m2}) \in \mu$; $s_{m1}, s_{m2} \in \omega_n$, $s_n \in \omega_{m1}$, $s_n \in \omega_{m2}$. The channel $(\varpi_{m2}, \varpi_{m1})$ will be removed.

The opening Assume we have the configuration $n[m_1[\ldots]|m_2[P_1|P_2|m_3[\ldots]|m_4[\ldots]]]$. The corresponding structure in the P system is $\varpi_n = \{n, t_n, \omega_n\}, \varpi_{m1} = \{m_1, t_{m1}, \omega_{m1}\}$, $\varpi_{m2} = \{m_2, t_{m2}, \omega_{m2}\}, \varpi_{p1} = \{p_1, t_{p1}, \omega_{p1}\}$, $\varpi_{p2} = \{p_2, t_{p2}, \omega_{p2}\}, \varpi_{m3} = \{m_3, t_{m3}, \omega_{m3}\}$, $\varpi_{m4} = \{m_4, t_{m4}, \omega_{m4}\} \in M$; $(\varpi_n, \varpi_{m1}), (\varpi_n, \varpi_{m2}), (\varpi_{m2}, \varpi_{p1}), (\varpi_{m2}, \varpi_{p2}), (\varpi_{m2}, \varpi_{m3}), (\varpi_{m2}, \varpi_{m4}) \in \mu$; $(s_n, n), (s_{m1}, m_1), (s_{m2}, m_2) \in S_P$, $(s'_{m1}, m_1), (s'_{m2}, m_2), (s'_{p1}, p_1), (s'_{p2}, p_2), (s'_{m3}, m_3), (s'_{m4}, m_4) \in S_C$; $s'_{m1}, s'_{m2} \in \omega_n$, $s_n \in \omega_{m1}$, $s_n, s'_{p1}, s'_{p2}, s'_{m3}, s'_{m4} \in \omega_{m2}$, $s_{m2} \in \omega_{p1}$, $s_{m2} \in \omega_{p2}$, $s_{m2} \in \omega_{m3}$, $s_{m2} \in \omega_{m4}$ Assume that the ambient m_2 opens. Then we obtain the configuration $n[m_1[\ldots]|P_1|P_2|m_3[\ldots]|m_4[\ldots]]$. In the P system we obtain the structure $\varpi_n = \{n, t_n, \omega_n\}, \varpi_{m1} = \{m_1, t_{m1}, \omega_{m1}\}$, $\varpi_{p1} = \{p_1, t_{p1}, \omega_{p1}\}$,

$\varpi_{p2} = \{p_2, t_{p2}, \omega_{p2}\}, \varpi_{m3} = \{m_3, t_{m3}, \omega_{m3}\}$, $\varpi_{m4} = \{m_4, t_{m4}, \omega_{m4}\} \in M$; $(\varpi_n, \varpi_{m1}), (\varpi_n, \varpi_{p1}), (\varpi_n, \varpi_{p2})$, $(\varpi_n, \varpi_{m3}), (\varpi_n, \varpi_{m4}) \in \mu$; $s'_{m1}, s'_{p1}, s'_{p2}, s'_{m3}, s'_{m4} \in \omega_n$, $s_n \in \omega_{m1}$, $s_n \in \omega_{p1}$, $s_n \in \omega_{p2}$, $s_n \in \omega_{m3}$, $s_n \in \omega_{m4}$. The channels $(\varpi_{m2}, \varpi_{p1})$, $(\varpi_{m2}, \varpi_{p2})$, $(\varpi_{m2}, \varpi_{m3})$, $(\varpi_{m2}, \varpi_{m4})$ will be removed.

In this way we have to elaborate the set of evolution/communication and structure rules in order to modify the configuration of the P system as shown above.

3.3 Security

As we mentioned above, there is a security feature in ambient calculus: the access to an ambient is authorized using the correct ambient name. The ambient name serves as some kind of password for accessing the ambient.

In the P system considered above, the membranes from the group A have the same names as the ambients they are staying for.

In this way, when a process is going to access some ambient, for example, the ambient named n, the corresponding membrane from the group P in the P system sends some multiset of objects to the membrane with name n. If the ambient with name n exists, that is the corresponding membrane with the name n exists, then with respect to the definition of modified tissue P systems with dynamic networks of membranes the multiset will be delivered to the membrane with address n. Otherwise, the action will be blocked until a membrane with name n appears.

4 Final Remarks

In this paper we have introduced a class of tissue P systems with da ynamic network of membranes.

The modified tissue P systems with a dynamic network of membranes could be used as an intermediate type to represent ambient calculus. The final scope of our work is to express ambient calculus by means of traditional hierarchical P systems with active [4] or mobile [8] membranes. Using these types of P systems it is possible to express tissue P systems with a dynamic network of membranes with *tree* topology.

A part of this work has already been completed in hierarchical P systems with communication controlled by concentration [9]. Namely, the architecture of communication between neighboring membranes has been implemented, and it was shown that the submembranes of a membrane communicates in the same manner as computer-nodes do in computer electronic networks with *common data transmission medium*. As an example of such kind of networks the Ethernet network technology was modelled. That model will be used as a communication instrument between mneighboring embranes in the model of mobile ambients in hierarchical P systems.

Several questions remain to be explored:

- the generative power of tissue P systems with a dynamic network of membranes and their relationships with other classes of P systems,
- the minimal class of P systems (P systems with minimal features) in which it is possible to express ambient calculus,
- the complexity of the model of mobile ambients in different types of P systems.

Acknowledgements. Work is supported by "MolCoNet" project IST-2001-32008, the Moldavian Research and Development Association (MRDA) and the U.S Civilian Research and Development Foundation (CRDF) Award No. MM2-3034.

References

[1] P.S. Alvarado, Defining an adaptable mobile transaction service, *Lecture Notes in Computer Science*, Vol. 2490, Springer, Berlin.

[2] L. Cardelli, A.G. Gordon, Mobile ambients, *Proc. FoSSaCS'98*, M. Nivat (Ed.), *Lecture Notes in Computer Science*, Vol. 1378, Springer, Berlin, 1998, 140–155.

[3] C. Martín-Vide, Gh. Păun, J. Pazos, A. Rodríguez-Patón: Tissue P systems. Turku Center for Computer Science TUCS, *Technical Report* 421, September 2001, and *Theoretical Computer Science*, 296, 2 (2003), 295–326.

[4] C. Martín-Vide, Gh. Păun, A. Rodríguez-Patón, On P systems with membrane creation. *Computer Science Journal of Moldova*, **9**, 2 (2001), 134–145.

[5] V. Olifer, N. Olifer, *Computing Networks*, Piter-Press, 2001.

[6] Gh. Păun, Computing with membranes, *Journal of Computer and System Sciences*, **61**, 1 (2000), pp. 108–143.

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

[8] I. Petre, L. Petre, Mobile ambients and P systems, *Journal of Universal Computer Science*, **5**, 9 (1999), 588–598.

[9] V. Rogozhin, E. Boian, Simulation of mobile ambients by P systems. Part 1, in *Proc of WMC 2003, Lecture Notes in Computer Science*, Vol. 2933, Springer, Berlin, 2004, 304–319.