Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 238 (2009) 5–22

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Rewriting Logic Specification of Membrane Systems with Promoters and Inhibitors [1]

Oana Agrigoroaiei[2]

Romanian Academy, Institute of Computer Science
Blvd. Carol I no.8, 700505 Iaşi, Romania

Gabriel Ciobanu[3]

"A.I.Cuza" University of Iaşi, Faculty of Computer Science
Blvd. Carol I no.11, 700506 Iaşi, Romania

Abstract

Membrane systems represent a new model of computation involving parallel application of rules, communication between membranes and dissolving. Since rewriting logic is a general framework for concurrent systems, we connect it with the operational semantics of membrane systems. We use a new representation given by register membranes which are able to express the evolution involving rules with promoters and inhibitors. The evolution is expressed in terms of both dynamic and static allocation of resources to rules. It is proved that these semantics are equivalent. Dynamic allocation allows translation of the maximal parallel application of membrane rules into sequential rewritings. An implementation in Maude is provided.

Keywords: rewriting logic, membrane system, operational semantics, Maude.

1 Introduction

Membrane computing is a rather young area of *natural computing* aiming to abstract computing ideas and models from the structure and the functioning of living cells. Several extensions comes both from biology, i.e., from the desire to capture more and more biological facts, and from mathematics and computer science, i.e., from the desire to have more powerful or more elegant models.

Natural computing represents a generic name for various fields consisting of ideas, models, and paradigms useful to computer science inspired from nature. Evolutionary computing and neural computing represent branches of natural comput-

[1] Research partially supported by CNCSIS Idei 402/2007

[2] Email: oanaag@iit.tuiasi.ro

[3] Email: gabriel@info.uaic.ro

doi:10.1016/j.entcs.2009.05.010

ing. Closer to membrane computing are the multiset processing languages, the most known of them being *Gamma* [5]. Membrane systems restricts the form of rules in Gamma, looking for both biological roots and mathematical simplicity and elegance. Membranes appear also in the *Chemical Abstract Machine* (CHAM). However the membranes of CHAM are not membranes as in cell biology, but correspond to the contents of membranes (multisets). The description and the goals of CHAM [6] are different from those of membrane systems; they are closer to the process algebras approach. Related formalisms are represented by *mobile ambient calculus* [8,18] and *brane calculus* [9]. A formal relationship between mobile ambients and mobile membranes is given in [1].

Membrane computing deals with distributed and parallel computing models, processing multisets of symbol objects in a localized manner (evolution rules and evolving objects are encapsulated into compartments delimited by membranes). Membrane computing was not initiated as an area aiming to provide models to biology, models of the cell in particular. At this moment, after developments at the theoretical level, the domain is prepared to offer such models to biology, and considerable advances toward such achievements have been reported.

In this paper we present two operational semantics of membrane systems which differ only in the way the maximal parallel application of rules is described. These two operational semantics reflect the fact that resource allocation to rules can be done either statically or dynamically. For membrane systems with promoters and inhibitors dynamical allocation requires the addition of a register to each of the system's membranes. We define an operational semantics of membrane systems by means of three sets of inference rules corresponding to maximal parallel rewriting, sending messages and dissolving. A minimal set of inference rules is defined, and their behaviour is detailed along with the presentation of the rewriting logic implementation. We use a uniform representation of rather complex membrane systems with promoters and inhibitors, and get a flexible interpreter for them in Maude. The main results provide the correspondence between the operational semantics of dynamic allocation and the rewriting theory.

2 Membrane Systems

A *membrane system* is composed of membranes which do not intersect, and which are all contained in a *skin* membrane. Each membrane can contain multisets of objects, evolution rules and other membranes. The objects inside a membrane evolve in a maximal parallel manner according to the evolution rules inside the same membrane. According to [16], maximal parallel "means that we assign objects to rules, non-deterministically choosing the objects and the rules, until no further assignment is possible." Membrane systems are also called P systems.

A multiset w over a set S is a function $w : S \to \mathbb{N}$. To each multiset w we associate its support, denoted by $supp(w)$,which contains those elements of S which have a non-zero image. A multiset is called non-empty if it has non-empty support. We denote the empty multiset by 0_S. If $S' \subseteq S$, we denote by $w|_{S'}$ the multiset over S' which is the restriction of w to S'.

The sum of two multisets w, w' over S is the multiset $w + w' : S \rightarrow \mathbb{N}, (w + w')(s) = w(s) + w'(s)$. The union of two multisets w, w' over S is the multiset $w \uplus w' : S \rightarrow \mathbb{N}, (w \uplus w')(s) = \max\{w(s), w'(s)\}$. For two multisets w, w' over S we say that w is contained in w' if $w \uplus w' = w'$, i.e. $w(s) \leq w'(s), \forall s \in S$. We denote this by $w \leq w'$.

The structure μ of a membrane system is represented by a tree structure (with the *skin* as its root), or equivalently, by a string of correctly matching parentheses, placed in a unique pair of matching parentheses; each pair of matching parentheses corresponds to a membrane. Graphically, a membrane structure is represented by a Venn diagram in which two sets can be either disjoint, or one the subset of the other. The membranes are labelled in a one-to-one manner. A membrane without any other membrane inside is said to be elementary. A special symbol δ is used to dissolve membranes.

A *membrane system* of degree m is a tuple $\Pi = (O, \mu, w_1, \ldots w_m, Rules(1), \ldots, Rules(m), i_o)$ where:

- O is an alphabet of objects;
- μ is a membrane structure, with the membranes labelled by natural numbers $1 \ldots m$, in a one-to-one manner;
- w_i are the initial multisets over O associated with the regions $1 \ldots m$ defined by μ;
- $Rules(1), \ldots, Rules(m)$ are finite sets of rules associated with the membranes $1 \ldots m$; the rules have the form $u \rightarrow v$, where u is a non-empty multiset of objects and v a multiset over messages of the form $(a, here), (a, out), (a, in_j), \delta$ with the condition that δ can appear at most once;
- i_0 is either a number between 1 and m specifying the output membrane of Π, or it is equal to 0 indicating that the output is the outer region.

The *skin* membrane, which is labelled by 1, is not allowed to be dissolved, so we consider that its rules do not involve δ.

For a rule of form $u \rightarrow v$, the message $(a, here)$ in v says that a, once created, remains in the membrane; (a, out) says that a, once created, is sent into the parent membrane (or into the environment, if the rule is inside the *skin* membrane); (a, in_j) says that a is sent into the child membrane with label j - if no such child membrane exists, the rule cannot be applied; if the special symbol δ appears in v, then the membrane is going to be dissolved, all its objects are to be sent to the parent membrane and its rules disappear.

We can associate *promoters* and *inhibitors* with a rule $u \rightarrow v$, in the form $(u \rightarrow v)|_{w_{prom}, \neg w_{inhib}}$, with w_{prom}, w_{inhib} non-empty multisets of objects. Such a rule associated with a membrane i is applied only if w_{prom} is present and w_{inhib} is absent from the region of the membrane i. The promoters and inhibitors of membrane systems formalize the reaction enhancing and reaction prohibiting roles of various substances present in cells. Membrane systems with promoters or with inhibitors provide characterizations of recursively enumerable sets (of vectors of natural numbers) [7]. If the maximal number of rules with promoters present in any membrane is 6, the number of membranes necessary to achieve universal computations can be reduced to 1. If the maximal number of rules with inhibitors present in any membrane is 6, the number of membranes necessary to achieve universal computations can be reduced to 3 (more details are presented in [7]).

Definition 2.1 Formally, the set $\mathcal{M}(\Pi)$ of membranes in a P system Π, and the membrane structure are inductively defined as follows:

- if i is a label and w is a multiset over $O \cup O \times \{here, out\} \cup \{\delta\}$ then $\langle i|w\rangle \in \mathcal{M}(\Pi)$; $\langle i|w\rangle$ is called an *elementary membrane*, and its structure is $\langle\rangle$;
- if i is a label, $M_1, \dots, M_n \in \mathcal{M}(\Pi), n \geq 1$ have distinct labels, each M_k has structure μ_k and w is a multiset over $O \cup O \times \{here, out\} \cup O \times \{in_j / j \text{ label of some } M_k\} \cup \{\delta\}$ then $\langle i|w; M_1, \dots, M_n\rangle \in \mathcal{M}(\Pi)$; $\langle i|w; M_1, \dots, M_n\rangle$ is called a *composite membrane*, and its structure is $\langle \mu_1 \dots \mu_n\rangle$.

Note that the children of $\langle i|w; M_1, \dots, M_n\rangle$ are precisely $M_1, \dots, M_n$, which are not necessarily elementary. In what follows we use $[n]$ to denote the set $\{1, \dots, n\}$.

For a unitary approach, we consider all multisets of objects and all multisets with objects with messages which are in the region determined by a membrane to be over

$$\Omega = O \cup O \times \{here\} \cup O \times \{out\} \cup O \times \{in_j\}_{j \in [n]}$$

meaning that, for example, multisets of objects are considered to be multisets over Ω, with the support included in O. Also, a multiset w over $O \cup O \times \{here, out\} \cup \{\delta\}$ is identified with a multiset w over Ω which has $supp(w) \cap O \times \{in_j\}_{j \in [n]} = \emptyset$. Since the special symbol δ is actually a signal to dissolve the membrane, we consider that $w(\delta)$ is always equal to 0 or to 1 . Also, since the addition of multisets over Ω must respect this convention, we set $(w + w')(\delta) = \min\{w(\delta) + w'(\delta), 1\}$.

We say that a multiset w is *here-free* if $w|_{O \times \{here\}} = 0_{O \times \{here\}}$. Similarly, we define *out-free*, *in-free* and δ-*free*.

3 Membrane Systems Semantics

In this section we present two transition relations which yield the same rewriting semantics for P systems. The reason for considering two such transition systems is that while one of them holds closer to what "maximal parallel rewriting" means, the other is easier to translate in rewriting logic and to implement. The semantics for message passing and dissolving are given separately, in order to be used with each of these transition systems. In what follows we view a rule as a tuple $r = (u, v, w_{prom}, w_{inhib})$ of multisets over Ω, such that:

- $supp(u) \subseteq O$ and $u|_O \neq 0_O$ (a rule has to consume at least one object);
- $supp(v) \subseteq \Omega \backslash O$ (the right hand side of a rule contains only messages);
- $supp(w_{prom}) \subseteq O$, $supp(w_{inhib}) \subseteq O$ (promoters and inhibitors are multisets of objects).

For such a rule r we define

$$lhs(r) = u,\ rhs(r) = v,\ promoter(r) = w_{prom},\ inhibitor(r) = w_{inhib}$$

If a rule r has no associated promoter, we set $promoter(r) = 0_\Omega$; if the rule has no associated inhibitor, we also set $inhibitor(r) = 0_\Omega$ (this convention is used for the sake of uniformity such that we do not have to differentiate between rules with and without inhibitors).

3.1 Dynamic Allocation Semantics

In order to give an operational semantics for membrane systems (operational semantics which can easily be transcribed in a rewriting logic implementation), we present a semantics based on applying rules one by one in a nondeterministic manner until there is no applicable rule left. We call this a *dynamic allocation semantics.*

In a maximal parallel evolution step, a rule's applicability with respect to promoters and inhibitors only depends on the initial content of the membrane. For this reason, when each transition consists of applying only a rule, we need to store somewhere the objects consumed by previous applications of rules. Thus, we need to consider membranes with registers consisting of a multiset of objects which have been consumed previously, to keep track of rule application. In a similar manner to that of Definition 2.1 we construct the set $\mathcal{M}_h(\Pi)$ of *register membranes*:

- if $M \in \mathcal{M}(\Pi)$ is an elementary membrane and $u : O \to \mathbb{N}$ then the pair $(M, u) \in \mathcal{M}_h(\Pi)$ and is called an *elementary register membrane*;
- if $M = \langle i|w; M_1 \dots M_n\rangle$ is a composite membrane and $u, u_1, \dots, u_n : O \to \mathbb{N}$ then $(\langle i|w; (M_1, u_1) \dots (M_n, u_n)\rangle, u) \in \mathcal{M}_h(\Pi)$ is called a *composite register membrane.*

We can see a membrane of $\mathcal{M}(\Pi)$ as an equivalence class of register membranes, i.e., obtained by ignoring registers. Namely, we define inductively a relation $\equiv$ on $\mathcal{M}(\Pi)$ as follows: $(< i|w >, u) \equiv (< i|w >, v), \forall u, v : O \to \mathbb{N}$ and if $H_1 \equiv H'_1, \dots, H_n \equiv H'_n$ then $(\langle i|w; H_1, \dots H_n\rangle, u) \equiv (\langle i|w; H'_1, \dots H'_n\rangle, v)$. Clearly, $\equiv$ is an equivalence relation.

Proposition 3.1 *The set $\mathcal{M}_h(\Pi)/_\equiv$ of equivalence classes is isomorphic with the set $\mathcal{M}(\Pi)$ of membranes of the P system Π.*

Proof. We construct $\phi : \mathcal{M}_h(\Pi) \to \mathcal{M}(\Pi)$ inductively by setting $\phi(\langle i|w\rangle, u) = \langle i|w\rangle$ and $\phi(\langle i|w; H_1, \dots H_n\rangle, u) = \langle i|w; \phi(H_1), \dots \phi(H_n)\rangle$. This map induces a bijection $\hat{\phi} : \mathcal{M}_h(\Pi)/_\equiv \to \mathcal{M}(\Pi)$. □

The states of the following transition system are register membranes. The labels are taken from the set $Rules \cup \{\tau\}$, where $Rules = \cup_{i \in [n]} Rules(i)$, and τ denotes a silent action – namely the evolution of a membrane in which all rewrites take place in the inner membranes while the content of the top membrane stays the same.

The following definition gives a mathematical description of what it means for a rule r to be applicable in a membrane M with register u.

Definition 3.2 Consider $H \in \mathcal{M}_h(\Pi)$, $H = (\langle i|w; H_*\rangle, u)$, and $r \in Rules(i)$; H_* is a (possibly empty) set of register membranes. We say that the pair (H, r) is *valid* when

- $lhs(r) \leq w$, $promoter(r) \leq w + u$;
- if $inhibitor(r) \neq 0_\Omega$ then $\exists a \in O$ such that $(w+u)(a) < inhibitor(r)(a)$;
- if H_* is empty then $rhs(r)(a, in_j) = 0, \forall j \in [m]$;
- if $H_* = \{H_1, \dots, H_n\}$ and $j_1, \dots, j_n$ are the labels of $H_1, \dots, H_n$ respectively, then $rhs(r)(a, in_j) = 0, \forall j \notin \{j_1, \dots, j_n\}$.

We say that an elementary register membrane H is *mpr-irreducible* if (H, r) is not valid for all $r \in Rules(i)$. We say that a composite register membrane $\langle i|w; H_*\rangle$ is *mpr-irreducible* if (H, r) is not valid for all $r \in Rules(i)$ and H_i is mpr-irreducible for all $H_i \in H_*$.

We define inductively a transition relation $T_{mpr} \subseteq \mathcal{M}_h(\Pi) \times Rules \cup \{\tau\} \times \mathcal{M}_h(\Pi)$ as follows:

- if $H = (\langle i|w\rangle, u)$ is an elementary register membrane and (H, r) is valid, then

$$\textbf{(seq−elem)}\ \frac{}{(\langle i|w\rangle, u) \xrightarrow{r} (\langle i|w - lhs(r) + rhs(r)\rangle, u + lhs(r))}$$

- if $H = (\langle i|w; H_1, \dots, H_n\rangle, u)$ is a composite register membrane and $\exists j \in [n]$ such that H_j is not mpr-irreducible, then

$$\textbf{(silent)}\ \frac{H_j \xrightarrow{l} H'_j}{(\langle i|w; H_1, \dots, H_n\rangle, u) \xrightarrow{\tau} (\langle i|w; H'_1, \dots, H'_n\rangle, u)}$$

  where $H'_k = H_k, \forall k \neq j$ and $l \in Rules \cup \{\tau\}$;
- if $H = (\langle i|w; H_1, \dots, H_n\rangle, u)$ is a composite register membrane such that H_j are mpr-irreducible $\forall j \in [n]$ and (H, r) is valid, then

$$\textbf{(rewrite)}\ \frac{}{(\langle i|w; H_1, \dots, H_n\rangle, u) \xrightarrow{r} (\langle i|w - lhs(r) + rhs(r); H_1, \dots, H_n\rangle, u + lhs(r))}$$

Note that the rules *seq-elem* and *rewrite* ensure that rules are first applied in elementary membranes until they become irreducible, then in their parents, and so on.

We now present two other transition relations $T_{msg} \subseteq \mathcal{M}(\Pi) \times \{msg\} \times \mathcal{M}(\Pi)$ and $T_{diss} \subseteq \mathcal{M}(\Pi) \times \{diss\} \times \mathcal{M}(\Pi)$ which express the message passing and the dissolving steps in the evolution of a membrane system. We use the isomorphism from Proposition 3.1 to glue together T_{mpr}, T_{msg} and T_{diss} as follows:
A membrane M evolves to a membrane N when

- $(H_M, H) \in T^*_{mpr}$, where H_M is the register membrane with its register and all the registers of its children equal to 0_Ω, $\phi(H_M) = M$, H is a mpr-irreducible register membrane and T^*_{mpr} is the transitive closure of T_{mpr} or H_M is mpr-irreducible and $H := H_M$;

- if $\phi(H)$ is msg-irreducible then $M' := \phi(H)$; otherwise, if $\phi(H)$ is not msg-irreducible, then there is a unique membrane M' such that $(\phi(H), M') \in T_{msg}$;
- if M' is diss-irreducible then $N := M'$; otherwise, if M' is not diss-irreducible, then N is the unique membrane for which $(M', N) \in T_{diss}$.

$$\begin{tikzcd}
H_M \arrow[r, "mpr"'] & {} \arrow[r, "mpr"'] & \cdots \arrow[r, "mpr"'] & H \arrow[d] & & \\
M \arrow[u] & & & \phi(H) \arrow[r, "msg"'] & M' \arrow[r, "diss"'] & N
\end{tikzcd}$$

In what follows, let $w(M)$ denote the multiset contained in the membrane M, and $L(M)$ denote the label of M.

Definition 3.3 We say that an elementary membrane $\langle i|w\rangle$ is *msg-irreducible* if either $i \neq 1$ and w is *here-free*, or $i = 1$ and w is both *here-free* and *out-free*. Recall that the label 1 means that the membrane is the *skin* membrane.
We say that a composite membrane $\langle i|w; M_1, \ldots, M_n\rangle$ is *msg-irreducible* if:

- w is *here-free* and *in-free*;
- $w(M_j)$ is *out-free*, for $j \in [n]$;
- if $i = 1$, then w is also *out-free*;
- $M_1, \ldots, M_n$ are msg-irreducible.

We define the following functions over multisets:

$$cleanup, out, in_j, eraseOut, eraseDelta : [\Omega \to \mathbb{N}] \to [\Omega \to \mathbb{N}]$$

- $cleanup(w)(a) = w(a, here) + w(a)$, $cleanup(w)(a, out) = w(a, out), \forall a \in O$, $cleanup(w)(\delta) = w(\delta)$ and $cleanup(x) = 0, \forall x \in O \times \{here, in_j / j \in [m]\}$;
- $out(w)(a) = w(a, out)$ and $out(w)(x) = 0$ for all $x \in \Omega \backslash O$;
- $in_j(w)(a) = w(a, in_j)$ and $in_j(w)(x) = 0$ for all $x \in \Omega \backslash O$;
- $eraseOut(w)(a, out) = 0$ and $eraseOut(w)(x) = w(x), \forall x \in \Omega \backslash O \times \{out\}$;
- $eraseDelta(w)(delta) = 0$ and $eraseDelta(w)(x) = w(x), \forall x \in \Omega \backslash \{\delta\}$.

The *cleanup* function modifies the multiset by "erasing" objects with messages of the form *in_child* and by "transforming" objects of form $(a, here)$ into objects of form a. The *out* function collects the objects a from the objects with messages of form (a, out); the function in_j is similarly defined. The *eraseOut* function removes objects with messages of form (a, out) from a multiset; similarly, *eraseDelta* removes special symbols δ.

The transition relation T_{msg} is given by the following inference rules:

- if $w \neq cleanup(w)$, or $i = 1$ and $w' \neq eraseOut(cleanup(w))$ then

$$\textbf{(msg1)}\ \frac{}{\langle i|w\rangle \xrightarrow{msg} \langle i|w'\rangle}$$

where $w' = cleanup(w)$ if $i \neq 1$, and $w' = eraseOut(cleanup(w))$ if $i = 1$.

- if M_j are msg-irreducible, $\forall j \in J \subseteq [n]$, then

$$(\mathbf{msg2})\ \frac{M_k \xrightarrow{msg} M'_k, \forall k \in [n]\backslash J}{\langle i|w; M_1, \ldots, M_n\rangle \xrightarrow{msg} \langle i|w'; M''_1, \ldots, M''_n\rangle}$$

where M''_l have the same label and structure as M_l, but $w(M''_j) = eraseOut(w(M_j) + in_{L(M_j)}(w))$,$\forall j \in J$, and $w(M''_k) = eraseOut(w(M'_k) + in_{L(M_k)}(w))$,$\forall k \notin J$, and either $w' = cleanup(w) + \sum_{l\in[n]} out(w(M_l))$ whenever $i \neq 1$, or $w' = eraseOut(cleanup(w)) + \sum_{l\in[n]} out(w(M_l))$ if $i = 1$.

In order to define the transition relation T_{diss}, we first define the notion of diss-irreducibility.

Definition 3.4 Any elementary membrane is *diss-irreducible.* A composite membrane $\langle i|w, M_1, \ldots, M_n\rangle$ is *diss-irreducible* if $w(M_j)(\delta) = 0$ and M_j is diss-irreducible for all $i \in [n]$.

In what follows, M_* and N_* range over (possibly empty) sets of membranes. The transition relation T_{diss} is given by the following inference rules:

- if $M_s = \langle i_s|w_s; M_{*s}\rangle$ are diss-irreducible $\forall s \in [n]$, and there exists a non-empty $J \subset [n]$ such that $w_j(\delta) = 1$, $j \in J$ and $w_k(\delta) = 0$ for $k \notin J$, then

$$(\mathbf{diss1})\ \frac{}{\langle i|w; M_1, \ldots, M_n\rangle \xrightarrow{diss} \langle i|w'; M_*\rangle}$$

where $w' = w + \sum_{j\in J} eraseDelta(w_j)$ and $M_* = (\cup_{k\notin J}\{M_k\}) \cup (\cup_{j\in J} M_{*j})$;

- if M_j are diss -irreducible for all $j \in J$ where $J \subseteq [n]$, then

$$(\mathbf{diss2})\ \frac{M_k \xrightarrow{diss} M'_k, \forall k \notin J}{\langle i|w; M_1, \ldots, M_n\rangle \xrightarrow{diss} \langle i|w; M'_1, \ldots, M'_n\rangle}$$

where $M'_j = M_j$ for all $j \in J$.

3.2 Static Allocation Semantics

Let R be a multiset over $Rules(i)$ for some label i. We denote by $lhs(R)$ the multiset over Ω given by $lhs(R)(a) = \sum_{r\in Rules(L)} R(r) \cdot lhs(r)(a)$. The multiset $rhs(R)$ is defined in the same way. We also define $promoter(R) = \biguplus_{r\in supp(R)} promoter(r)$.

Definition 3.5 For a membrane $M =< i|w; M_* >$ and for R a multiset of rules over $Rules(i)$ we say that the pair (M, R) is *valid* when

- $lhs(R) \leq w$, $promoter(R) \leq w$
- for all $r \in supp(R)$, either $inhibitor(r) = 0_\Omega$ or there exists $a_r \in \Omega$ such that $w(a_r) < inhibitor(r)(a_r)$;
- if $rhs(R)(a, in_j) > 0$ then there exists a membrane with label j in M_*.

We say that the pair (M, R) is *maximally valid* if it is valid, and for any multiset R' over $Rules(i)$ such that $R \leq R'$ and (M, R') is valid it follows that $R = R'$.

Note that the multiset R is not required to be non-empty, therefore the pair $(M, 0_{Rules(i)})$ is valid, and can even be maximally valid (when no rule from $Rules(i)$ can be applied).

We also point out why the notions of *inhibitor* and *promoter* are not dual: every rule in R is applicable to w with respect to promoters if and only if $promoter(R) = \biguplus_{r \in supp(R)} promoter(r) \leq w$, yet the negation of $\biguplus_{r \in supp(R)} inhibitor(r) \leq w$ is the statement "*there is* $a \in \Omega$ *such that* $w(a) < \biguplus_{r \in supp(R)} inhibitor(r)(a)$". This is stronger than "*for all* $r \in supp(R)$ *there exists* $a_r \in \Omega$ *such that* $w(a_r) < inhibitor(r)(a_r)$" which is equivalent to "*every rule* r *in* R *is applicable to* w *with respect to inhibitors*".

We define a transition system over the set of membranes by the following rules:

- if $(\langle i|w\rangle, R)$ is maximally valid then

$$\textbf{(mpr1)}\ \frac{}{\langle i|w\rangle \xrightarrow{R} \langle i|w - lhs(R) + rhs(R)\rangle}$$

- if $(\langle i|w; M_1, M_2, \ldots, M_k\rangle, R)$ is maximally valid then

$$\textbf{(mpr2)}\ \frac{M_1 \xrightarrow{R_1} N_1, \ldots, M_k \xrightarrow{R_1} N_k}{\langle i|w; M_1, M_2, \ldots, M_k\rangle \xrightarrow{R} \langle i|w - lhs(R) + rhs(R); N_1, N_2, \ldots, N_k\rangle}$$

We say that M is mpr-irreducible if $M \xrightarrow{0_{Rules}} M$, and all its children are also mpr-irreducible.

Note that even if $M \xrightarrow{R} N$, it does not mean that N is mpr-irreducible: since inhibitors can be consumed by the multiset of rules R, it may be the case that there is a non-empty multiset S of rules such that (N, S) is maximally valid.

We prove now the equivalence between static and dynamic allocation semantics. In what follows, let $w(H)$ denote the content of a register membrane H, and $reg(H)$ its register, i.e. $H = (\langle i|w(H); H_*\rangle, reg(H))$.

Proposition 3.6 *Consider a membrane* M *which is mpr-reducible. Let* $H_M \in \phi^{-1}(M)$ *denote the register membrane corresponding to* M*, which has its register and all the registers of inner membranes equal to* 0_Ω*. If* $M \xrightarrow{R} N$ *then there exist* $l_1, \ldots, l_k \in Rules \cup \{\tau\}$ *such that* $H_M \xrightarrow{l_1} \ldots \xrightarrow{l_k} H$*,* H *is mpr-irreducible and* $card\{i \in [k] / l_i = r\} = R(r)$*. Moreover,* $N = \phi(H)$*.*

$$\begin{tikzcd}
H_M \arrow[r, "l_1"] \arrow[d, dotted, bend right, "\phi"'] & \ldots \arrow[r, "l_k"] & H \arrow[d, dotted, bend left, "\phi"] \\
M \arrow[rr, "R"'] \arrow[ur, phantom, "\Uparrow"] & & N
\end{tikzcd}$$

Proof. By induction over the structure of M. First, note that if $H \xrightarrow{l_1} \ldots \xrightarrow{l_k} H'$ then $reg(H') = reg(H) + \sum_{l_j \neq \tau} lhs(l_j)$.

If M is an elementary membrane and $supp(R) = \{r_1, \ldots, r_s\}$, then we set $l_{N_j+1} = \ldots = l_{N_{j+1}} = r_{j+1}$ where $N_0 = 0$ and $N_{j+1} = R(r_j) + N(j)$ for $j \in \{0, \ldots, s-1\}$. In other words, we set the first $R(r_1)$ labels to be r_1, then the next $R(r_2)$ labels to be r_2 and so on. Let $H_1 := H_M = (\langle i|w(M)\rangle, 0_\Omega)$. We prove by induction on $p < N(s)$ that (H_p, l_p) is valid, where H_p is given by $H_{p-1} \xrightarrow{l_{p-1}} H_p$. This follows from the fact that $(w(H_p)+reg(H_p))|_O = (w(H_{p-1})+reg(H_{p-1}))|_O = \ldots = w(M)|_O$. If there exist some rule r' in $Rules(i)$ such that $(H_{N(s)}, r')$ is valid, then it follows that $(M, R+r')$ is also valid, and this contradicts the maximal validity of (M, r).

If M is a composite membrane with children $M^1, \ldots, M^n$, then it follows that for each $j \in [n]$ there exist $l^j_1, \ldots, l^j_{k_j} \in Rules \cup \{\tau\}$ such that $H_{M_j} \xrightarrow{l^j_1} \ldots \xrightarrow{l^j_k} H^j$. If M^j is mpr-irreducible we just set $k_j = 1$ and $H^j = H_{M^j}$. Then there exist s such that $t_1 = t_2 = \ldots t_s = \tau$ and

$$H_M = \langle i|w(M); H_{M^1} \ldots H_{M^n}\rangle \xrightarrow{t_1} \ldots \xrightarrow{t_s} \left\langle i|w(M); H^1 \ldots H^n\right\rangle$$

and all H^j are mpr-irreducible. From here on we repeat the procedure of setting rules as labels similarly to the case of the elementary membrane.

□

Proposition 3.7 *If H_0 is a register membrane with its register and all registers of inner membranes equal to 0_Ω and $H_0 \xrightarrow{l_1} \ldots \xrightarrow{l_k} H_k$ in T_{mpr} such that H_k is mpr-irreducible, then there exists a multiset R over the rules in $\phi(H_0)$ such that $\phi(H_0) \xrightarrow{R} \phi(H_k)$. Moreover, $R(r) = card\{i \in [k]/l_i = r\}$.*

$$\begin{tikzcd}
H_0 \arrow[r, "l_1"] \arrow[d, dotted, "\phi"'] & \cdots \arrow[r, "l_k"] \arrow[d, Rightarrow] & H_k \arrow[d, dotted, "\phi"] \\
\phi(H_0) \arrow[rr, "R"'] & & \phi(H_k)
\end{tikzcd}$$

Proof. By induction over the structure of H_0. If H_0 is an elementary register membrane with label i then $l_j \in Rules(i)$ for all $j \in [k]$. Let R be the multiset over $Rules(i)$ given by $R(r) = card\{i \in [k]/l_i = r\}$. Since $reg(H_k) = \sum_{j\in[k]} lhs(l_j)$ it follows that $(w(H_0)+reg(H_k))|_O = w(M)|_O$. Therefore $(\phi(H_0), R)$ is valid. If there exists another valid multiset of rules $R' \supsetneq R$, then let consider a rule $r \in R' \backslash R$. It follows that (H_k, r) is valid, and thus there exists H_{k+1} such that $H_k \xrightarrow{r} H_{k+1}$ which contradicts the mpr-irreducibility of H_k. Clearly, $\phi(H_0) \xrightarrow{R} \phi(H_k)$ since $w + \sum_j (rhs(l_j) - lhs(l_j)) = w - lhs(R) + rhs(R)$ (with some abuse of the notation).

If H_0 is composite, consider $J = \{j \in [k]/l_j = \tau\}$. Then $H_0 \xrightarrow{\tau} \ldots \xrightarrow{\tau} H$, where the number of silent actions τ is equal to the cardinal of J. We repeat for H the above reasoning about the multiset of rules. □

These two propositions show that the static allocation can be considered to be a *big-step* semantics [11] for the rewriting stage of the evolution of a membrane, while dynamic allocation provides an equivalent *small-step* semantics [17]. We use static allocation semantics together with the previously defined T_{msg} and T_{diss} to describe the evolution of a membrane system analogously to the manner in which we used together T_{mpr}, T_{msg} and T_{diss}.

4 Rewriting Specification for Dynamic Allocation

A certain familiarity of the reader with rewriting logic is assumed. A good reference is [14]. Rewriting logic is a computational logic which combines equational logic with term rewriting. More precisely, a *rewrite theory* is a triple (Σ, E, R) , where Σ is a signature of function symbols, E a set of (possibly conditional) Σ-equations, and R a set of (possibly conditional) Σ-rewrite rules. The conditions for a rewrite rule can involve both equations and rewrite rules. Generally, a typed setting is used in the form of a membership equational logic (Σ, E) [13] which has sorts, subsort inclusions and kinds (connected components of sorts). The notation $\mathcal{R} \vdash t \to t'$ is used to express that $t \to t'$ is provable in the theory $\mathcal{R}$ using the inference rules of rewriting logic.

In what follows we use the syntax of the rewriting engine Maude [10] to describe a rewriting theory which corresponds to the semantics presented in Section 3.1.

In the previous sections all multisets of objects and messages were considered to be over Ω. This inspires us to describe in rewriting logic a multiset of objects and messages as consisting of four "bags" of which three are multisets of objects (standing for objects which are actually in the membrane, objects with message *here*, objects with message *out*), and the fourth containing a multiset of pairs of objects and labels i which stand for objects with message in_i. This representation facilitates the rewriting logic specification because in this way there is no need for additional sorts with respect to messages. This representation is equivalent to the representation of multisets over Ω because we have the following bijection

$$[\Omega \to \mathbb{N}] \simeq [O \to \mathbb{N}] \times [O \cup \{\delta\} \to \mathbb{N}] \times [O \to \mathbb{N}] \times [O \times \{m\} \to \mathbb{N}]$$

We first consider the following sorts:

```
sorts Obj ObjMultiset ObjAddressMultiset Label Rule RuleSet .
subsort Obj < ObjMultiset . subsort Rule < RuleSet .
```

By `emptyMO` and `emptyMAO` we denote the empty multiset of objects, respectively of objects with labels, and use `+` to denote the addition on both `ObjMultiset` and `ObjAddressMultiset`. The multiset of objects with addresses is constructed through the operator `op in : ObjMultiset Label -> ObjAddressMultiset`. A rule is constructed through the operator

```
op _->_|_|_|_|_ : ObjMultiset ObjMultiset ObjMultiset
 ObjAddressMultiset ObjMultiset ObjMultiset -> Rule [ctor] .
```

The first slot is for the objects to be consumed (it is the left hand side of the rule); the second slot is for the objects produced with label "here"; the third slot is for the objects produced with label "out"; the fourth slot is for the objects produced with label "in_child"; the last two slots are for promoters respectively inhibitors. The operators which are used to manipulate the components of a rule are

```
ops lhs rhsHere rhsOut promoter inhibitor : Rule -> ObjMultiset .
op rhsIn : Rule -> ObjAddressMultiset .
```

and `rulesIn : Label -> RuleSet` is used to present the rules inside a membrane.

We work with register membranes even when implementing message passing and dissolving. This does not modify in any way the semantics, since we identify a membrane M with the register membrane H_M, as in the Propositions 3.6 and 3.7. In what follows, all the membranes are register membranes, even when not explicitly stated. A register membrane is constructed through the operator

```
op <_‘[_|_|_|_‘]_>_ : Label ObjMultiset ObjMultiset ObjMultiset
 ObjAddressMultiset MembraneSet ObjMultiset -> Membrane [ctor] .
```

The first slot is for the label; the second slot is for the objects inside the membrane; the third slot is for the objects with label "here"; the fourth slot is for the objects with label "out"; the fifth slot is for the objects with label "in_child"; the sixth slot is for the set of children membranes; the last slot is for the register. The operators which are used to manipulate the components of a rule are

```
op labelOf : Membrane -> Label .
ops register here : Membrane -> ObjMultiset .
ops content out : MembraneSet -> ObjMultiset .
op inChildren : Membrane -> ObjAddressMultiset .
op children : Membrane -> MembraneSet .
```

Other operators are `_isIn_` which evaluates whether a multiset is contained in another multiset, `mprIrred`, `msgIrred`, `dissIrred`, `eraseDelta`, `emptyOut` and `emptyReg` whose names are self-explaining. We also use `labelsOf` to gather the membrane labels which appear in the right hand side of a rule, `membraneSetLabels` for the same purpose with respect to the membrane sets, and `subsetOf` to compare them. These last three functions are used only when evaluating whether a pair formed of a membrane M and a rule R is valid:

```
op valid : Membrane Rule -> Bool .
ceq valid(M, R) = true if lhs(R) isIn content(M) /\ promoter(R) isIn
(content(M) + register(M)) /\ labelsOf(rhsIn(R)) subsetOf
membraneSetLabels(children(M)) /\ if (inhibitor(R) =/= emptyMO)
then (inhibitor(R) isIn (content(M) + register(M)) == false) else true fi .
eq valid(M, R) = false [owise] .
```

To separate the three stages of evolution of a membrane we use four tags:

```
sorts evolutionType State .
ops mpr msg diss end : -> evolutionType [ctor] .
op _;_ : MembraneSet evolutionType -> State [ctor] .
```

where `end` is used to stop the rewriting once the membrane has stopped evolving.

The maximal parallel rewriting of a membrane is given by the following rules:

```
crl [1] : M , MM ; mpr => M1 , MM ; mpr if
MM =/= null /\ M ; mpr => M1 ; mpr /\ M =/= M1 .

crl [2] : < L [ W1 | W2 | W3 | A ] MM > W4 ; mpr =>
```

```
< L [ W1 - lhs(R) | W2 + rhsHere(R) | W3 + rhsOut(R) | A + rhsIn(R) ]
MM > (W4 + lhs(R) ) ; mpr if mprIrred(MM) /\ R RR := rulesIn(L)
/\ valid(< L [ W1 | W2 | W3 | A ] MM > W4, R) .

crl [3] : < L [ W1 | W2 | W3 | A ] MM > W4 ; mpr =>
< L [ W1 | W2 | W3 | A ] MM1> W4 ; mpr if
mprIrred(MM) == false /\ MM ; mpr => MM1 ; mpr /\ MM =/= MM1 .
```

These rules impose the following evolution: if in a membrane there is some mpr-reducible child membrane, then the membrane is replaced by a similar membrane which has that child rewritten (rules `crl [3]` and `crl [1]`); if a membrane has only mpr-irreducible children, all valid rules are applied one by one (rule `crl [2]`). When even the *skin* membrane is mpr-irreducible, the following rule is applied

```
crl [4] : M ; mpr => emptyReg(M) ; msg if labelOf(M) == 1 /\ mprIrred(M) .
```

in order to empty the register and to begin the next evolution stage, that of the message sending.

This message sending stage is governed by the following rules:

```
crl [5] : M , MM ; msg => M1 , MM ; msg if
MM =/= null /\ M ; msg => M1 ; msg /\ M =/= M1 .

crl [6] : < L [ W1 | W2 | W3 | A ] MM > W4 ; msg => if L == 1 then
< L [ W1 + W2 + out(MM1) | emptyMO | emptyMO | emptyMAO ]
emptyOut(sendIn(A, MM1)) > W4 ; msg else
< L [ W1 + W2 + out(MM1) | emptyMO | W3 | emptyMAO ]
emptyOut(sendIn(A, MM1)) > W4 ; msg fi
if msgIrred(MM) == false /\ MM ; msg => MM1 ; msg /\ msgIrred(MM1) .

crl [7] : < L [ W1 | W2 | W3 | A ] MM > W4 ; msg => if L == 1 and
W3 =/= emptyMO then < L [ W1 + W2 + out(MM) | emptyMO | emptyMO | emptyMAO ]
emptyOut(sendIn(A, MM)) > W4;msg else
< L [ W1 + W2 + out(MM) | emptyMO | W3 | emptyMAO ]
emptyOut(sendIn(A, MM)) > W4 ; msg fi if msgIrred(MM)
/\ (A =/= emptyMAO) or (W2 =/= emptyMO) or out(MM) =/= emptyMO .

crl [8] : M ; msg => M ; diss if labelOf(M) == 1 /\ msgIrred(M) .
```

In this stage a membrane evolves in a single rewriting step: if the set MM of children membranes is msg-reducible, then MM rewrites to a msg-irreducible $MM1$(rule `crl [5]`); the membrane M with objects $W1$ which contains MM is rewritten to the membrane $M1$ with objects $W1 + W2 + out(MM1)$ (i.e. the objects with messages of form $(a, here)$ are transformed in objects of form a, and the objects sent out by the set $MM1$ of membranes are added), and children $emptyOut(sendIn(A, MM1))$ (i.e. the objects of form (a, in_j) are sent into the membrane with label j and then the objects with messages of form (a, out) are erased from every child membrane). The result is msg-irreducible, because the only objects with messages are in the membrane $M1$, and they are of the form (a, out) (if $M1$ is the *skin* not even those objects remain). If the set MM of children membranes is msg-irreducible, then the same process takes place, except that instead of $MM1$ it is still MM(rule [7]).

Rules `crl [5]`, `crl [6]` and `crl [7]` correspond to inference rules $msg1$ and $msg2$. In defining the transition relation T_{msg} we treat the case of an elementary membrane separately, since we prefer to avoid extending T_{msg} to sets of membranes.

Although rules `crl [6]` and `crl [7]` look almost identical, we cannot include them in a single rule with the conditional part `if MM ; msg => MM1 ; msg` because it would lead to an infinite loop of identical rewritings. This happens because $MM; msg \rightarrow MM; msg$ is provable in rewriting logic.

When the entire membrane system is msg-irreducible, the rule

```
crl [8] : M ; msg => M ; diss if labelOf(M) == 1 /\ msgIrred(M) .
```

is applied. This rule starts the next evolution stage, that of dissolving.

The rules for dissolving membranes are:

```
crl [9] : M , MM ; diss => M1 , MM ; diss if
MM =/= null /\ M ; diss => M1 ; diss /\ M =/= M1 .

crl [10] : < L [ W1 | W2 | W3 | A ] MM > W4 ; diss =>
< L [ W1 + eraseDelta(content(M)) | W2 | W3 | A ]
children(M) , MM1 > W4 ; diss if dissIrred(MM)
/\ M , MM2 := MM /\ delta isIn content(M) /\ MM1 := children(M) , MM2 .

crl [11] : < L [ W1 | W2 | W3 | A ] MM > W4 ; diss =>
< L [ W1 | W2 | W3 | A ] MM1 > W4 ; diss
if dissIrred(MM) == false /\ MM ; diss => MM1 ; diss /\ dissIrred(MM1) .
```

If the set MM of children membranes for a membrane M is diss-reducible and it rewrites to a diss-irreducible set of membranes $MM1$, then M is rewritten to the similar membrane $M1$ which has children membranes $MM1$ (rules `crl [9]` and `crl [11]`). When the set MM of children membranes is diss-irreducible and at least one of the membranes in MM contains the special symbol δ, then all the membranes from MM which contain δ are dissolved (rule `crl [10]`). Note that a top membrane M does not dissolve even when it does contain δ. This happens because the rewriting rules are given with the purpose of describing the evolution of the *skin* membrane, which can never dissolve. Rules `crl [9]`, `crl [10]` and `crl [11]` correspond to inference rules $msg1$ and $msg2$. Again, we have used the first rule in this group as a stepping stone towards the rewriting of a set of sibling membranes, while avoiding to include the rewriting of a set of sibling membranes in the transition relation T_{diss}.

When even the *skin* membrane is diss-irreducible but is mpr-reducible, the rule

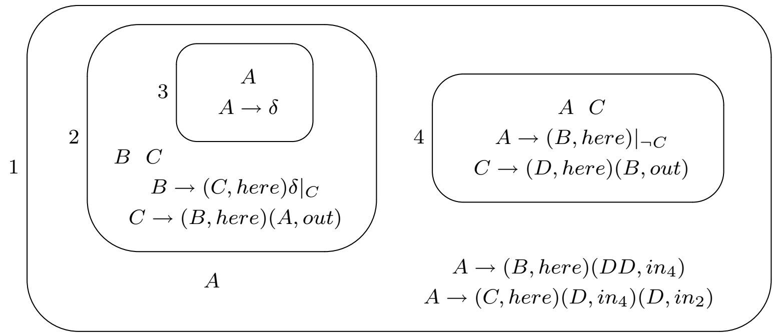
```
crl [12] : M ; diss => M ; mpr if labelOf(M) == 1
/\ dissIrred(M) /\ mprIrred(M) == false .
```

is applied; it starts once more the maximal parallel rewriting stage of the evolution. However, if the *skin* membrane is also mpr-irreducible, rule

```
crl [13] : M ; diss => M ; end if labelOf(M)==1/\dissIrred(M)/\mprIrred(M).
```

is applied; in this case it ends the rewriting. We do not need to evaluate the msg-irreducibility of the *skin* membrane, because the dissolving stage can only be reached by msg-irreducible membranes.

We give an example of membrane system, chosen such that its evolution will be non-trivial in all three stages:



1
2
3
A
$A \to \delta$
$B\ C$
$B \to (C, here)\delta|_C$
$C \to (B, here)(A, out)$
4
$A\ C$
$A \to (B, here)|_{\neg C}$
$C \to (D, here)(B, out)$
A
$A \to (B, here)(DD, in_4)$
$A \to (C, here)(D, in_4)(D, in_2)$

It is described in rewriting logic in the form of the following Maude module:

```
mod EXAMPLE is
inc TRANZ .

ops A B C D : -> Obj .
ops Q Q1 Q2 Q3 : -> Membrane .
ops R1 R2 R3 R4 R5 R6 R7 : -> Rule .
ops 2 3 4 : -> Label .
eq R1 = A -> B | emptyMO | emptyMAO | emptyMO | C .
eq R2 = C -> D | B | emptyMAO | emptyMO | emptyMO .
eq R3 = A -> B | emptyMO | in(D + D, 4) | emptyMO | emptyMO .
eq R4 = A -> C | emptyMO | in(D, 4) + in(D, 2) | emptyMO | emptyMO .
eq R5 = B -> C + delta | emptyMO | emptyMAO | C | emptyMO .
eq R6 = C -> B | A | emptyMAO | emptyMO | emptyMO .
eq R7 = A -> delta | emptyMO | emptyMAO | emptyMO | emptyMO .

eq Q1 = < 4 [ A + C | emptyMO | emptyMO | emptyMAO ] null > emptyMO .
eq rulesIn(4) = R1 R2 .
eq Q3 = < 3 [ A | emptyMO | emptyMO | emptyMAO ] null > emptyMO .
eq rulesIn(3) = R7 .
eq Q2 = < 2 [ B + C | emptyMO | emptyMO | emptyMAO ] Q3 > emptyMO .
eq rulesIn(2) = R5 R6 .
eq Q = < 1 [ A | emptyMO | emptyMO | emptyMAO ] Q1 , Q2 > emptyMO .
eq rulesIn(1) = R3 R4 .
endm
```

When entering the rewrite command `rew Q ; mpr .` Maude presents the following output:

```
result State: < 1[B + B + B + B + C | emptyMO | emptyMO | emptyMAO]<
4[B + D + D + D + D + D | emptyMO | emptyMO | emptyMAO]null >
emptyMO > emptyMO ; end
```

The correspondence between the operational semantics given by the transition relations T_{mpr}, T_{msg} and T_{diss} on one hand, and the rewriting logic implementation on the other hand is presented as follows.
Consider the map $\psi : \mathcal{M}_h(\Pi) \to$`Membrane` and the induced $\Psi : \mathcal{M}(\Pi) \to$`Membrane`, where ψ is defined inductively by $\psi(\langle i|w\rangle, u) =< i[w1|w2|w3|w4]null > u$ with $w1, w3$ multisets over O, $w2$ multiset over $O \cup \{\delta\}$ and $w4$ multiset over $O \times [m]$ such that $w1 = w|_0$, $w2(a) = w(a, here)$, $w2(\delta) = w(\delta)$, $w3(a) = w(a, out)$ and $w4(a, i) = w(a, in_i)$. Moreover, $\psi(\langle i|w; H_1 \dots H_n\rangle, u) =< i[w1|w2|w3|w4]\psi(H_1)\dots\psi(H_n) > u$. Since in rewriting logic the multisets are not represented as functions but by bags, we use the equivalence between multisets as functions and multisets as "bags" (i.e. classes of equivalence of strings). The function Ψ is defined by using a register membrane with empty registers $H_M \in \phi^{-1}(M)$: $\Psi(M) = \psi(H_M)$.

Let $\mathcal{R}_{\mathcal{M}}$ denote the rewriting theory given by the rewriting rules `crl [1]..[13]` as well as operators and equations defining them. Let *tag* range over the set of tags$\{mpr, msg, diss\}$. The following theorem will formalise the correspondence between the dynamic allocation semantics (together with the message passing and dissolving semantics) and the rewriting theory.

First, consider $H \neq H' \in \mathcal{M}_h(\Pi)$ and such that if the label of H is 1 then there does not exist H'' such that $\mathcal{R}_{\mathcal{M}} \vdash \psi(H); mpr \rightarrow \psi(H''); tag$ and $\mathcal{R}_{\mathcal{M}} \vdash \psi(H''); tag \rightarrow \psi(H'); mpr$ with $tag \neq mpr$. Consider $M \neq M' \in \mathcal{M}(\Pi)$ with the analogue property for *msg* and *diss*. We impose this additional condition for the *skin* (register) membrane in order to ensure that we are precisely in the stage *mpr* (respectively *msg* or *diss*), since the *skin* membrane is the only one which can change its tag. A tag change signifies that the membrane (carrying with it all its children membranes) has entered another stage of evolution. We prefer to use this method for the separate study of the three stages, instead of removing **transitivity** from the inference rules in rewriting logic deduction.

Theorem 4.1 (i) *If H and its inner membranes do not contain any objects with messages or the special symbol δ then*

$$(H, H') \in T^*_{mpr} \textit{ iff } \mathcal{R}_{\mathcal{M}} \vdash \psi(H); mpr \rightarrow \psi(H'); mpr$$

(ii) *If M is mpr-irreducible then*

$$(M, M') \in T_{msg} \textit{ iff } \mathcal{R}_{\mathcal{M}} \vdash \Psi(M); msg \rightarrow \Psi(M'); msg$$

(iii) *If M is mpr and msg-irreducible then*

$$(M, M') \in T^*_{diss} \textit{ iff } \mathcal{R}_{\mathcal{M}} \vdash \Psi(M); diss \rightarrow \Psi(M'); diss.$$

Proof. By structural induction. The proof follows the reasoning presented above when describing each stage (marked by the tags *mpr*,*msg*,*diss*) through rewriting rules. □

5 Conclusion and Related Work

Previous papers [2,3,4] describe rewriting logic specifications of the operational semantics of general P systems. Note that the difference between general P systems and P systems with promoters and inhibitors is not trivial. In general P systems it makes no difference from the point of view of a membrane's final content whether a multiset of rules are applied at once or one rule at a time. When promoters and inhibitors are associated to rules, it may happen that in a sequential application of rules some promoters and/or inhibitors are consumed, forbidding/allowing the application of some rules which could not have been applied at the start of the maximal parallel rewriting. Thus the need for the definition of register membranes.

The rewriting logic theory describing this class of membrane systems (in the syntax of Maude) differs substantially from those in our previous work. The differences appear both in the nature of the membrane systems representation and in the nature of the rewriting rules which use tags only on the top membrane to impose a certain evolution. In [4] the rewriting rules used markings both on objects and on membranes which traversed the tree of the membrane structure both from leaves to root(*skin*) and in reverse. In this article we prefer to use a more complex configuration (the membrane is implemented with 4 compartments for objects, instead of 1) instead of more complex operators.

Theorem 4.1 shows the faithfulness of the rewriting logic representation of the dynamic allocation semantics. This result connects through a rather sequential semantics a parallel/concurrent model of computation (membrane systems) to a logic whose theories can specify parallel/concurrent systems.

In [2] we have presented a *big-step* operational semantics for membrane systems. This was translated into rewriting logic [12], obtaining a *small-step* operational description. *Big-step* and *small-step* semantics were also described in rewriting logic for a small imperative programming language in [19], part of the rewriting logic semantics project [15] which aims at unifying algebraic denotational semantics and structural operational semantics.

Acknowledgement

We thank the referees for their useful suggestions which improved the paper.

References

[1] B. Aman, G. Ciobanu. Translating Mobile Ambients into P Systems. *Electronic Notes in Theoretical Computer Science*, vol. 171, 11–23, 2007.

[2] O. Andrei, G. Ciobanu, D. Lucanu. Executable Specifications of the P Systems. *Membrane Computing. WMC5*, Lecture Notes in Computer Science vol. 3365, 127–146, 2005.

[3] O. Andrei, G. Ciobanu, D. Lucanu. Operational Semantics and Rewriting Logic in Membrane Computing, *Electronic Notes of Theoretical Computer Science* vol.156, 57–78, 2006.

[4] O. Andrei, G. Ciobanu, D. Lucanu. A Rewriting Logic Framework for Operational Semantics of Membrane Systems, *Theoretical Computer Science* vol. 373, 163–181, 2007.

[5] J.-P. Banâtre, A. Coutant, D. Le Métayer. A Parallel Machine for Multiset Transformation and Its Programming Style. *Future Generation Computer Systems*, vol. 4, 133–144, 1988.

[6] G. Berry, G. Boudol. The Chemical Abstract Machine. *Theoretical Computer Science*, vol. 96, 217–248, 1992.

[7] P. Bottoni, C. Martín-Vide, G. Paun, G. Rozenberg. Membrane systems with promoters/inhibitors. *Acta Informatica*, vol. 38(10), 695–720, 2002.

[8] L. Cardelli, A. Gordon. Mobile Ambients. *Foundations of Software Science and Computation Structures*, *Lecture Notes in Computer Science*, vol. 1378, 140–155, 1998.

[9] L. Cardelli. Brane Calculus. *Computational Methods in Systems Biology*, *Lecture Notes in Computer Science*, vol. 3082, 257–280, 2005.

[10] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, J.F. Quesada. Maude: Specification and Programming in Rewriting Logic. *Theoretical Computer Science*, vol. 285, 187–243, 2002.

[11] G. Kahn. *Natural semantics*, Technical Report 601, INRIA Sophia Antipolis, 1987.

[12] N. Marti-Oliet, J. Meseguer. Rewriting logic as a logical and semantical framework. *Handbook of Philosophical Logic*, 2nd.edn, Kluwer Academic, 1–87, 2002.

[13] J. Meseguer. Membership algebra as a logical framework for equational specification. *WADT 1997*, *Lecture Notes in Computer Science*, vol. 1376, 18–61, 1997.

[14] J. Meseguer, G. Rosu. Rewriting Logic Semantics: From Language Specifications to Formal Analysis Tools. *IJCAR 2004*, *Lecture Notes in Computer Science*, vol. 3097, 1–44, 2004.

[15] J. Meseguer, G. Roşu. The Rewriting Logic Semantics Project. *Theoretical Computer Science*, vol. 373, 213–237, 2007.

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

[17] G. Plotkin. Structural Operational Semantics. *Journal of Logic and Algebraic Programming*, vol. 60, 17–139, 2004.

[18] A. Regev, E.M. Panina, W. Silverman, L. Cardelli, E. Shapiro. BioAmbients – An Abstraction for Biological Compartments. *Theoretical Computer Science*, vol. 325, 141–167, 2004.

[19] T.F. Şerbănuţă, G. Roşu, J. Meseguer. A Rewriting Logic Approach to Operational Semantics. *Electronic Notes in Theoretical Computer Science*, vol. 192(1), 125–141, 2007.