A Recursive Algorithm for Describing Evolution in Transition P Systems

Fernando Arroyo1, Angel V. Baranda2, Juan Castellanos2, Carmen Luengo1, Luis F. Mingo3

1Dpto. de Lenguajes, Proyectos y Sistemas Informáticos
Escuela Universitaria de Informática, Universidad Politécnica de Madrid
Ctra. de Valencia Km. 7, 28031 Madrid, Spain
E-mail: {farroyo, cluengo}@eui.upm.es

2Dpto. de Inteligencia Artificial, Facultad de Informática
Universidad Politécnica de Madrid
Campus de Montegancedo, 28660 Boadilla del Monte, Madrid, Spain
E-mail: jcastellanos@fi.upm.es

3Dpto. de Organización y Estructura de la Información
Escuela Universitaria de Informática, Universidad Politécnica de Madrid
Ctra. de Valencia Km. 7, 28031 Madrid, Spain
E-mail: lfmingo@eui.upm.es

Abstract. This paper presents a recursive algorithm for simulating the execution of transition P systems. In order to do it, it is necessary to synchronise and to communicate objects among regions. In order to implement the simulated execution of transition P systems on digital computers, it is necessary to first characterize the parallel execution of evolution rules inside regions (this has been done in previous papers), and then to study how it can be possible to synchronise communication among regions in transition P systems.

1 Introduction

The reader is supposed to already be familiar with P systems. To simulate the execution of transition P systems on digital computers it is necessary to represent them into abstract data structures[2]. Here we summarise the main data structures in which the basic components of static structure of transition P system have been represented. This previous work is necessary for parsing the static structure of transition P systems in term of data able to be manipulated by a computer. The main component of static structure of transition P systems is the membrane structure [3], [5]. Basically, a membrane structure is defined by a language MS over the alphabet {[,]}, whose strings are recurrently defined as follows:

• \([\ ] \in MS\),
• if \(\mu_1 \cdots \mu_n \in MS, n \geq 1\), then \([\mu_1, \cdots, \mu_n] \in MS\),
• Nothing else is in \(MS\).
Over $MS$ we define the equivalence relation $\sim$ as follows: Let $x, y \in MS$, then $x \sim y$ if and only if $x = \mu_1 \mu_2 \mu_3 \mu_4$ and $y = \mu_1 \mu_2 \mu_3 \mu_4$, where $\mu_1, \mu_2, \mu_3, \mu_4 \in MS$. Let us denote $MS/\sim = MS$. The elements of $MS$ are called membrane structures. Each matching pair of parentheses $[]$ is called a membrane. The external membrane is called the skin. A membrane without internal membranes (i.e., having the form $[]$) is called an elementary membrane. This recursive structure can be parsed to the abstract data structure of tree.

Membranes from a membrane structure delimit regions. The Venn diagram representation of a membrane structure helps to clarify this notion. A region is any closed space delimited by membranes. It is also clear that a membrane structure of degree $n$ contains $n$ regions, one associated with each membrane. Moreover, we will say that two regions are adjacent if and only if there is only one membrane between them $[3], [4]$. This fact is important because communication between two regions is possible if they are adjacent. In order to define a computational device starting from a membrane structure it is necessary to add contents to regions. In particular, in transition $P$ systems, region content is represented in terms of a multiset of objects from a given set of symbols, represented by a set $U$. Moreover, multisets in regions are changed by application of some evolution rules, in parallel, from a given set of rules associated with each region. Until now we have considered the basic components of transition $P$ systems. Let us now define them in term of data structure and with associate them some operations and functions.

1.1 Multisets

A multiset over a given object set $U$ is defined by a mapping from $U$ to the natural number set $\mathbb{N}$. Every mapping can be also be represented in form of a set of coordinate pairs as follows:

$$M = \{(a, Ma) \mid a \in U\}. \quad (1)$$

It is important to know which objects are really present in the region with at least one copy. That subset of a multiset is called the Support of the multiset: $\text{Supp} M = \{a \in U \mid Ma > 0\}$. Moreover, there is the empty multiset represented by the map $0$, and defined as follows:

$$0 : U \to \mathbb{N}, \quad a \to 0. \quad (2)$$

Therefore, in order to represent a multiset in term of a set of coordinate pairs we do not need to refer to all the elements of $U$, but only to those elements belonging to the support of the multiset.

Multisets can be also represented by a polynomial with exponents in the natural number set; this second representation is more compact and natural, and it is used in most of the papers about $P$ systems. In this representation, objects which has the image equal to zero are not represented.

$$a^{n_a} b^{n_b} \cdots z^{n_z} = \{(a, n_a), (b, n_b), \ldots, (z, n_z)\} \quad (3)$$

1.1.1 Definitions:

Let $M$, $M_1$ and $M_2$ be multisets over $U$. Then: **Multiset Inclusion**: $M_1 \subseteq_u M_2$ if and only if $\forall a \in U, (M_1) a < (M_2) a$. **Multiset Union**: $\forall a \in U, (M_1 \cup M_2) a = (M_1 a) + (M_2 a)$. **Multiset Difference**: $\forall a \in U, (M_1 - M_2) a = (M_1 a) - (M_2 a)$, if $M_1 \subseteq M_2$. 

20
1.2 Evolution Rules

Evolution rules make evolve the region they are associated with. Moreover, they can make disappear the external membrane of the region. An evolution rule is formed by one antecedent and by one consequent. Both of them are represented by multisets over different sets. Formally, an evolution rule is defined as follows. Let $L$ be a set of labels, let $U$ be a set of objects, and let $D = \{\text{out}, \text{here}\} \cup \{\text{in } j \mid j \in L\}$ be the set of regions where a rule can send objects. An evolution rule with label in $L$ and objects in $U$ is a triple $(u, v, \delta)$, where $u$ is a multiset over $U$, $v$ is a multiset over $U \times D$ and $\delta \in \{\text{dissolve, not dissolve}\}$.

Over this data structure for representing evolution rules, we can define several operations.

1.2.1 Definitions:

Let $r = (u, v, \delta)$, $r_1 = (u_1, v_1, \delta_1)$ and $r_2 = (u_2, v_2, \delta_2)$ be evolution rules with labels in $L$ and objects in $U$. Let $a \in U$ and $n$ be a natural number. Then: Addition of Evolution rules: $r_1 + r_2 = (u_1 \cup u_2, v_1 \cup v_2, d_1 \cup d_2)$. Product of an evolution rule by a natural number: $n \cdot r = \sum_{i=1}^{n} r$. Inclusion of evolution rules: $r_1 \subseteq r_2$ if and only if $u_1 \subseteq u_2$.

Input of an evolution rule: $Input r = u$. Dissolve: $Dissolve r = \delta$.

Also, it is necessary to define some functions over evolution rules related to the rule consequent. These functions will provide important information over what happens when the rule is applied, and they will be used for defining evolution in regions of a P system.

Labels of membranes where the rule is sending objects:

$$Ins r = \{j \in L \mid (\bullet in_j) \in Supp v\}.$$  

Outputs of an evolution rule:

$$OutputToOut r a = v(a, \text{out})$$

$$OutputToHere r a = v(a, \text{here})$$

$$((OutputToIn j)r)a = v(a, in \ j)$$

Functions OutputToOut, OutputToHere, OutputToIn, return the multiset that the rule is sending to its “father region”, (ToOut), to the curent region where the rule is applied, (ToHere), and to a determined region (ToIn · · ·). These functions will be very useful in order to define the changed produced during the evolution of a transition P system.

1.3 Regions

A region is the inter-membranes space; they form vesicles for containing objects and a set of evolution rules able to make evolve the region. In order to define more precisely a region, let $L$ be a label set and $U$ be an object set. A region with the label in $L$ and objects in $U$ is a triple $(l, \omega, (R, \rho))$ where $l \in L$, $\omega$ is a multiset over $U$, $R$ is a set of evolution rules with labels in $L$ and objects in $U$ and $\rho$ is a partial order relation over $R$.

1.4 Transition P Systems

The tree structure of membrane structure in a transition P system suggests to represent the membrane structure as a tree. We define this more formally: A tree over an arbitrary set $S$ is a pair where the first component is an element in $S$ and the second one is a set of trees over $S$. In this definition, the second element of the pair can be the empty set, 

21
a case when the element is a leaf. Over trees it can be defined the multiplicity function which provides the number of occurrences of a given element \( a \in U \) in the tree.

**Multiplicity:** Let \( T = (u, S) \) be a tree, where \( u \in U \) and \( S \) is a set of trees over \( U \), then:

\[
(Mult \ T)a = \begin{cases} 
1 + \sum_{s \in S} (Mult \ s)a & \text{if } a = u \\
\sum_{s \in S} (Mult \ s)a & \text{if } a \neq u
\end{cases}
\]  

(4)

The membrane structure defines a tree whose root is the skin and elementary membranes are the leaves.

Finally, a transition P system can be defined as a tree of regions uniquely labelled. In a formal way, let \( L \) be a set of labels, let \( U \) be a set of objects. A transition P system with labels in \( L \) and objects in \( U \) is \( \Pi \), where \( \Pi \) is a tree of regions with labels in \( L \) and objects in \( U \) provided that:

\[
\forall l \in L, (Mult \ II)(l, a) < 2.
\]

(5)

An alternative definition can be given: A transition P system with labels in \( L \) and objects in \( U \) is a pair whose first element is a region with labels in \( L \) and objects in \( U \) and the second element is a set of transition P systems with labels in \( L \) and objects in \( U \), whose regions are uniquely labelled, that is,

\[
\Pi = ((l, \omega, (R, \rho)), \Pi) \text{ and } \forall l \in L, (Mult \ II)(l, a) < 2,
\]

(6)

where \( \omega \) is a multiset over \( U \), \( R \) is a set of evolution rules over \( L \) and \( U \), \( \rho \) is a partial order relation over \( R \), and \( \Pi \) is a set of transition P systems with labels in \( L \) and objects in \( U \). This algebraic representation defines precisely the static structure of transition P systems. We have called it a static structure because this representation do not take into account the evolution in transition P systems, it only represents transition P systems in a static manner.

## 2 Calculus of Complete Multisets of Evolution Rules in Regions

We pass now to characterize the parallel application of evolution rules. We will describe the dynamic evolution in a given region of a P system in a determined step of evolution. The obtained result will be to specify a set of characteristic evolution rules able to make evolve the region by the application of one of them [1]. A region has been defined as a triple:

\[
reg = (label, multiset, (Rules, Partial \ Order \ Relation)) = (l, \omega, (R, \rho))
\]

(7)

If we want to define the rules able to make evolve the region, we will need the following definitions:

**Maximal Set:** Let \( (U, <) \) be a partial order over \( U \); then:

\[
((Maximal \ U) <) = \{u \in U \mid \not\exists v \in U, u < v\} = \{u | (u \in U) \land (\neg (u < \_))\}
\]

(8)

**Useful Rules:** An evolution rule in a set of rules is useful over a given set of labels \( L \) if and only if \( Insr \) is included in \( L \). In a more formal way, let \( R \) be a set of evolution rules with labels in \( L \) and objects in \( U \). Let \( L' \subset L \).

\[
(Useful \ R)L' = \{r \in R \mid Insr \subseteq L' \} = \{r | (r \in R) \land (Insr \subseteq L')\}
\]

(9)
By the definition of transition P systems, objects communicated by rules can pass through only one membrane.

**Applicable Rules:** A rule is applicable over a multiset $\omega$ if and only if its antecedent is included in $\omega$. Let $R$ be a set of evolution rules with labels in $L$ and objects in $U$. Let $\omega$ be a multiset over $U$.

$$\text{(Applicable } R\text{)}\omega = \{r \in R| Input \ r \subset \omega\} = \{r|\ (r \in R) \land (Input \ r \subset \omega)\} \quad (10)$$

Therefore, a rule is applicable in a region if and only if the rule antecedent is included in the multiset that the region contains. In addition, in a determined evolution step of a transition P system, the applicable rules are the only ones that can be used.

**Active Rules:** A rule $r \in R$ is active with respect to a partial order relation $\rho$ defined over $R$ if and only if there is not any rule in $R$ higher than $r$ according to $\rho$. Therefore, active rules are characterized as follows. Let $(R, \rho)$ be a partial order relation defined over a set of evolution rules $R$ with labels in $L$ and objects in $U$. Then:

$$\text{(Active } R\text{)}\rho = \text{(Maximal } R\text{)}\rho \quad (11)$$

**Adjacent regions to a region:** Let $reg$ be a region of a transition P system $\Pi$ with labels in $L$ and object in $U$. Let $(reg, \Pi)$ be the sub-tree of $\Pi$ with the root is $reg$. We define the regions adjacent to $reg$ in $\Pi$ as follows.

$$\text{ADJACENT}_{\Pi} reg = \{r| (r, \Pi') \in \Pi\Pi\} = \{r| (r, \omega) \in \Pi\Pi\} \quad (12)$$

**Rules able to make evolve the region:** With these definitions, it is possible to determine the set of evolution rules that can be applied in order to make evolve the region $reg$. By definition, they are those rules satisfying the following equation:

$$L' = \{l \in L| (t, \omega) \in \text{ADJACENT}_{\Pi} reg\} \quad (13)$$

where $L' = \text{Active}(\text{Applicable}(\text{Useful } R)L')\omega\rho$

The dynamic evolution of a transition P system is described inside a region as follows: every rule satisfying (13) can be applied in parallel to all occurrences of rule antecedent multiset, for all regions at the same time [3]. Therefore, we must search for a linear combination of rules satisfying (13) and complete. The application of two rules $r_1, r_2 \in R$ at the same evolution step in a region, in parallel, is equivalent to applying the rule $r_1 + r_2$. In the same sense, we can say that to apply $n$ times the same rule in parallel has the same effect as applying the rule defined by $nr$. Therefore, parallel application of evolution rules inside a region can be substituted by the application of a linear combination of evolution rules. What is necessary is that such a linear combination is maximal in covering the multiset of objects of the region, i.e., it is a complete linear combination of evolution rules in the region. Finally, the region evolves in one step by the application of a rule formed by a complete linear combination of evolution rules in $R$ satisfying (13) [1]. Moreover, any linear combination of evolution rules can be represented as a multiset over $R$, the set of evolution rules of the region as follows. Let $R = \{r_1, \ldots, r_n\}$ be the set of evolution rules of region $reg$. Any linear combination of evolution rules in $R$ and scalars in $\mathbb{N}$ can be represented by a multiset over $R$ as follows:

$$\sum_{r_i \in R} n_i r_i \Rightarrow MR : R \rightarrow \mathbb{N}, \quad (14)$$

$$r_i \rightarrow n_i \rho.$$  

Once it is known how it is possible to make evolve a region in one step, we need to characterize the multisets of complete evolution rules.
2.0.1 Definitions:

Let $R$ be a set of evolution rules with labels in $L$ and objects in $U$. Let $MR, MR_1$ and $MR_2$ be multisets over $R$ and let $\omega$ be a multiset over $R$.

**MultiAdd:** This operation performs the transformation of a multiset of evolution rules into a linear combination of evolution rules. From now on, we will keep in mind this equivalence between linear combinations of evolution rules and multiset of evolution rules.

\[
\oplus MR = \sum_{r \in R} (MR_r)r
\]

**MultiInclusion:** This binary relation defines a partial order relation over multiset of evolution rules over $R$. It will be very useful in order to determine which multiset over $R$ will be complete:

\[
MR_1 \subseteq_u MR_2 \equiv (\oplus MR_1) \subseteq_u (\oplus MR_2)
\]

**MultiApplicable:** This function defines the set of multisets over $R$ (evolution rules set) that can be applied over $\omega$.

\[
(MultiApplicable\ R)_\omega = \{MR \mid (MR : R \to N) \wedge ((Input (\oplus MR)) \subseteq \omega)\}
\]

**MultiComplete:**

\[
(Multi\ complete\ R)_\omega = (Maximal((MultiApplicable\ R)_\omega)) \subseteq_u
\]

The above expression describes the set of complete multisets over $(R_u \subseteq_u$) and $\omega$.

Finally, let $\Pi$ be a transition $P$ system with labels in $L$ and objects in $U$, let $\text{reg} = (l, \omega, (R, \rho))$ be one region of $\Pi$ and let $(\text{reg}, \Pi\Pi)$ be the sub-tree of $\Pi$ whose root is $\text{reg}$. The set of complete multisets of evolution rules is defined by the following equations:

\[
LABELS = \{l \in L \mid (l, \omega, \bot) \in \Pi\Pi\}
\]

\[
ACTIVES = \text{Active(Applicable((Useful\ R)LABELS)_\omega)}\rho
\]

\[
COMPLETE\ complete\ ACTIVES = (Multi\ complete\ ACTIVES)_\omega
\]

The $LABELS$ set defines the set of labels of regions adjacent to region $\text{reg}$. The $ACTIVES$ set defines the set of evolution rules that can be applied in order to make the region evolve. The $COMPLETE$ set defines the set of complete multisets over $R$ (the set of evolution rules of the region) capable of making evolve the region in one step. Therefore, there are as many different possibilities for the region evolution as many elements $COMPLETE$ has.

3 Characterization of Region Evolution

By now, it has been determined, for each region, the set of multiset of evolution rules able to make them evolve. Let us fix a determined region $\text{reg}$ in a transition $P$ system $\Pi = (skin, \Pi\Pi)$. The region $\text{reg}$ is the element of a node from the regions tree defined by $\Pi$, and that node is the root of a sub-tree of $\Pi$. In order to study the evolution of $\text{reg}$, it is necessary to consider: the $\text{reg}$'s father region, denoted by $\text{reg}f$, and the sub-tree whose root is $\text{reg}$, denoted by $(\text{reg}, \Pi\Pi)$. Another way to represent $\Pi\Pi$ is given by the following equation:

\[
\{(\text{reg}, \Pi\Pi_i) \mid i \in LABELS, \text{reg}\}
\]

24
What has been stated in (19) is that evolution in each region of $\Pi$ is obtained by the application of only one rule. Therefore for every $MR \in COMPLETES_{reg}$, MultiAddMR determines one possible evolution for $reg$. If we want to pass from the actual configuration of $\Pi$ to the next one, it is necessary to define the contents of every region in $\Pi$ for the next configuration, and to determine those regions that the system keeps. Hence, some regions can disappear after the application of their corresponding complete multisets of evolution rules because the rules can make disappear the external membrane that defines the region.

The process of computing a complete multiset of evolution rules for every region in $\Pi$ and to applying it in every region can be make in parallel. However, for getting the next configuration for the transition P system $\Pi - \Pi_{NEXT}$ – all regions must have finished their own changing process. That implies that all regions in $\Pi$ must have sent and received objects to/from their adjacent regions. Moreover, if some of the external membranes disappear by application of the complete multisets of evolution rules in regions, such regions must send their contents to their father regions. When this process finished in every region of $\Pi$, then the system gets its next configuration, and the evolutionary process can continue.

Let us now describe how a region $reg \in \Pi$ gets its new configuration. Region $reg$ determines the sub-tree ($reg, \Pi_{reg}$). In order to determine the new configuration of the region, the following elements are necessary:

- the father region $reg_f$ and $MR_f \in COMPLETES_{reg_f}$
- $reg$ and $MR_{reg} \in COMPLETES_{reg}$
- $\forall l' \in LABELS_{reg}, reg$ and $MR_l \in COMPLETES_{reg_l}$

The configuration of the region $reg$ in the next evolution step will be:

$$reg_{NEXT} = (l, \omega_{NEXT}, (R, \rho))$$

(The node of the super-cell that defines $\Pi$, if it exists, is defined by:

$$(reg_{NEXT}, \Pi_{NEXT})$$

Therefore, what must be computed are $\omega_{NEXT}$ and $\Pi_{NEXT}$ because $l$ and $(R, \rho)$ are fixed and they are not changed from one step to other one. If the node does not exist in the new configuration, then $\omega_{NEXT}$ will be added to $\omega_{NEXT}$ - the multiset of its father region- and $\Pi_{NEXT}$ will be added to $\Pi_{NEXT}$ - the set of transition P system in its father region. Hence, these two elements define exactly the necessary dataset to establish the new configuration of the P system $\Pi$ for all regions in $\Pi$. The computation of $\omega_{NEXT}$ can be performed in three phases:

1. The first phase can be performed in parallel in every region of $\Pi$. It is related to objects consumed by the complete multiset of evolution rules and objects sent to the region by the applied rule. The expression for calculating this first state of $\omega_{NEXT}$ is given by:

$$\omega = (Input(MultiAdd MR)) \cup (OutputToHere(MultiAdd MR))$$

25
2. The second phase is where communication among regions is produced. Regions send
and receive objects from their respective father regions and from their respective
adjacent regions. With the introduced notation, it is possible to calculate the objects
multisets that a determined region is receiving from its environment and to transform
them into one objects multiset:

\[
((\text{OutputToInl})(\text{MultiAdd MR}_l)) \\
\cup \forall \nu \in \text{LABELS}_\text{reg} \\
(\text{OutputToOut}(\text{MultiAdd MR}_r))
\]

3. The third phase is devoted to add to \( \omega \) the contents of its adjacent regions that the
applied complete multiset of evolution rules have dissolved their external membranes.
These contents must to be added to \( \omega \) too. The following expression determines the
resultant objects multiset from them:

\[
\cup \forall \nu \in \text{LABELS}_\text{reg} \cap \text{Dissolve}(\text{MultiAdd MR}_r) \omega'_{\nu\text{NEXT}}
\]  

(26)

If the first phase can be done in parallel in every region of \( \Pi \), the second and the third
ones cannot. They involve some synchronization among membranes. Moreover, the third
one cannot be started until the second one has finished in every adjacent membrane to
\( \text{reg} \). The resulting expression for \( \omega_{\text{NEXT}} \) is given by the following equation:

\[
\omega_{\text{NEXT}} = \omega - (\text{Input}(\text{MultiAdd MR})) \\
\cup (\text{OutputToHere}(\text{MultiAdd MR})) \\
\cup ((\text{OutputToInl})(\text{MultiAdd MR}_l)) \\
\cup \forall \nu \in \text{LABELS}_\text{reg} (\text{OutputToOut}(\text{MultiAdd MR}_r)) \\
\cup \forall \nu \in \text{LABELS}_\text{reg} \cap \text{Dissolve}(\text{MultiAdd MR}_r) \omega'_{\nu\text{NEXT}}
\]  

(27)

It is also necessary to obtain an expression for \( \Pi_{\text{NEXT}} \); this determines the set of descen-
dent nodes that the node containing the region \( \text{reg} \) has in \( \Pi \), or if the external membrane
of \( \text{reg} \) is dissolved by \( \text{MR} \) the set of nodes that the node containing region \( \text{reg} \) provides
to the transition \( \text{P} \) system \( \Pi \). The following equation determines such an expression:

\[
\Pi_{\text{NEXT}} = \{ (\text{reg}', \Pi_{\text{NEXT}}), \forall \nu \in \text{LABELS}_\text{reg} \cap \text{NOTDissolve}(\text{MultiAdd MR}_r) \} \Pi_{\text{NEXT}}
\]  

(28)

Equations (27 & 28) express algebraically the way to calculate evolution in every region of
a transition \( \text{P} \) system. Finally, the contribution made by the region \( \text{reg} \) to the transition \( \text{P} \)
system can be expressed in term of its contents and the complete sub-tree starting in it – if the external membrane is not dissolved by the applied complete multiset of evolution rules.

- If \( \text{NOTDissolve}(\text{MultiAdd MR}) \) then \( \text{reg} \) is kept in \( \Pi \) and the sub-tree with root
in \( \text{reg} \) is defined by:

\[
(\text{reg}_{\text{NEXT}}, \Pi_{\text{NEXT}}) = ((l, \omega_{\text{NEXT}}, (R, \rho)), \Pi_{\text{NEXT}})
\]  

(29)
• If Dissolve(MultiAdd MR) then $\text{reg}$ is kept in $\Pi$ anymore and the sub-trees from adjacent regions to $\text{reg}$ must be added to the father region of $\text{reg}$, $\text{reg}_f$. In this case, evolution of $\text{reg}$ provides a set of sub-trees not one sub-tree as in the previous case.

$$
\{\Pi \Pi' | l' \in \text{LABELS}_{\text{reg} \text{NEXT}} \}
$$

In order to unify the evolution of $\text{reg}$, when the region is not kept in the system it can be said that evolution of $\text{reg}$ produces an unitary set of transition P systems; in the other case, the evolution of $\text{reg}$ produces a set of transition P system that could be even the empty set. The above reasoning is applicable to every region of $\Pi$ excepting the skin region, because the external membrane of $\Pi$ cannot be dissolved.

4 A Recursive Algorithm for Implementing the Evolution of Transition P Systems

Transition P systems are computational devices that have not been implemented. One possibility for implementing them is by simulation on digital computers, and this is the way we have chosen to follow. Simulations have advantages and disadvantages over other ways for implementing models of Molecular Computing. The main disadvantage is that in most cases the parallel nature of such computational paradigms has to be lost. However, the simulation offers in many cases the possibility of testing the power of the simulated systems before they have been developed and often it serves as a tool for showing the capabilities of the simulated models. Having in mind these characteristics of software simulations, here we present a recursive algorithm for simulating the evolution in transition P systems.

The simulation of transition P systems have to be launched by a global function applied to the whole system. Let us to call it NEXT. The NEXT function will receive two parameters: a multiset of objects and a transition P system $\Pi$. As usual, the transition P systems $\Pi$ will not receive input from the outside, so the multiset of objects will be the empty multiset; however, NEXT will be applied to the set of transition P systems which form the second component of the tuple that defines the system and then they will receive the object multiset that their father region is sending to them. The NEXT function will return a set of tuples whose first element will be a set of transition P systems produced by the evolution of the input transition P system $\Pi$ and the second one is the objects multiset that the external region of the transition P system sends to the external region. The phases two and three described above for the evolution of regions are joined in the functional algorithm. The algorithm for NEXT function will provide every possible next configuration for the transition P system.

4.1 Functional Algorithm for Transition P system Evolution

$$
\text{NEXT}(o, \Pi) = (\text{NEXT}_\delta(o, \Pi) \cup \text{NEXT}_{\omega}(o, \Pi))
$$

The set of possible transitions for the input configuration of $\Pi$ are produced by the application of complete multisets of evolution rules that can dissolve or not the external membrane of the system (note that there is only one skin membrane in $\Pi$, but NEXT is recursively defined and every node of the tree associated with $\Pi$ can be viewed as a transition P system, and it can dissolve its external membrane). The output produced by $\text{NEXT}_\delta(o_m, \Pi_m)$, where $\Pi_m$ is any given transition P system from $\Pi$ is given by:

$$
\text{NEXT}_\delta(o_m, \Pi_m) = \{(\Pi_{out}, o_{out}) | (\Pi_{in} = ((\omega_{in}, (R_{in}, \rho_{in})), \Pi_{\Pi_{in}}))\}
$$
The computation of the set is given by varying over the set of transition P systems enclosed by $\Pi_{\in}$. Such a calculus is developed in four steps.

1. Computation of complete multisets of evolution rules for the region:
\[
\text{reg} = (l_{\in}, \omega_{\in}, (R_{\in}, \rho_{\in}))
\]
\[
\text{LABELS} = \{l|((l, \cdot), \cdot) \in \Pi_{\Pi_{\in}} \} \tag{33}
\]
\[
\text{ACTIVES} = \text{Active(Applicable((Useful \ R_{\in}) \text{LABELS})\omega_{\in})}\rho_{\in} \tag{34}
\]
\[
\text{COMPLETE} = (\text{Multicomplete \ ACTIVES})\omega_{\in} \tag{35}
\]

2. For all complete multisets of evolution rules one computes the multiset of objects which remain in $\text{reg}_{\Pi_{\in}}$ and the multiset of objects that are sent out of the region:
\[
\forall MR \in \text{COMPLETE} \land (\text{Dissolve}(\oplus MR))
\]
\[
(\omega_{\text{intra}} = (\omega_{\in} - (\text{Input}(\oplus MR)) \cup (\text{OutputToHere}(\oplus MR))) \tag{36}
\]
\[
(o_{\text{prevout}} = (\text{OutputToOut}(\oplus MR))) \tag{37}
\]

3. Once $MR$ is fixed, it sends to each element of $\Pi_{\Pi_{\in}}$ one multiset of objects for making evolve them by the application of the $\text{NEXT}$ function. After the application of $\text{NEXT}$, outputs will be collected.
\[
\forall \Pi \in \Pi_{\Pi_{\in}} \text{SELECTION} = \tag{38}
\]
\[
\{\text{NEXT}((\text{OutputToIn} \ l) \oplus MR, \Pi)|\Pi = ((l, \cdot), \cdot)\}
\]
\[
(\Pi_{\Pi_{\text{intra}}} = \cup \{\Pi_{\Pi_{\text{i}}}|\Pi_{\text{i}} \in \text{SELECTION} \Pi\Pi \}) \tag{39}
\]
\[
(o_{\text{intra}} = o_{\in} \cup (\cup \{o\}|o \in \text{SELECTION} \ o) \tag{40}
\]

4. Finally, the application of the complete multiset of evolution rules produces an output for $\Pi$ in terms of a tuple with two elements (in the case when the external membrane of $\Pi$ is dissolved): the first one is the set of evolved transition P systems sons of $\Pi_{\in}$, and the second one is the multiset of objects that $\Pi_{\in}$ sends to its father.
\[
\Pi_{\Pi_{\text{out}}} = \Pi_{\Pi_{\text{intra}}} \tag{41}
\]
\[
o_{\text{out}} = o_{\text{prevout}} \cup \omega_{\text{intra}} \cup o_{\text{intra}} \tag{42}
\]

In a very similar way is defined $\text{NEXT}_{-\delta}(o_{\in}, \Pi_{\in})$, where $\Pi_{\in}$ is any given transition P system from $\Pi$:
\[
\text{NEXT}_{-\delta}(o_{\in}, \Pi_{\in}) = \{(\Pi_{\Pi_{\text{out}}}, o_{\text{out}})|(\Pi_{\Pi_{\in}} = ((l_{\in}, \omega_{\in}, (R_{\in}, \rho_{\in})), \Pi_{\Pi_{\text{in}}})) \} \tag{43}
\]

The computation of the set is given by varying over the set of transition P systems enclosed by $\Pi_{\in}$. Such a calculus is developed in four steps.

1. Computation of complete multisets of evolution rules for the region:
\[
\text{reg} = (l_{\in}, \omega_{\in}, (R_{\in}, \rho_{\in}))
\]
\[
\text{LABELS} = \{l|((l, \cdot), \cdot) \in \Pi_{\Pi_{\in}} \} \tag{44}
\]
\[
\text{ACTIVES} = \text{Active(Applicable((Useful \ R_{\in}) \text{LABELS})\omega_{\in})}\rho_{\in} \tag{45}
\]
\[
\text{COMPLETE} = (\text{Multicomplete \ ACTIVES})\omega_{\in} \tag{46}
\]

28
2. For all complete multisets of evolution rules one computes the multiset of objects which remain in \( \text{reg}_i \) and the multiset of objects that are sent out of the region:

\[ \forall MR \in \text{COMPLETES} \land \neg \text{Dissolve}(\oplus MR) \]

\[ (\omega_{\text{intra}} = (\omega_{\text{in}} - (\text{Input}(\oplus MR)))) \cup (\text{OutputToHere}(\oplus MR)) \]  \hspace{1cm} (47)

\[ (o_{\text{preout}} = (\text{OutputToOut}(\oplus MR))) \]  \hspace{1cm} (48)

3. Once \( MR \) is fixed, it sends to each element of \( \Pi \) one object multiset for making them evolve by application of \textbf{NEXT} function. After the application of \textbf{NEXT}, outputs will be collected.

\[ \forall \Pi \in \Pi_{i_{in}} \textbf{SELECTION} = \]

\[ \{ \text{NEXT}((\text{OutputToIn} l) \oplus MR, \Pi)|\Pi = ((l, \_ , \_ )) \]  \hspace{1cm} (49)

\[ (\Pi_{\text{intra}} = \cup_{(\omega_{\text{in}}) \in \text{SELECTION} \Pi}\Pi) \]  \hspace{1cm} (50)

\[ (o_{\text{intra}} = o_{\text{in}} \cup (\cup_{(\omega_{\text{in}}) \in \text{SELECTION} o}) \]  \hspace{1cm} (51)

4. Finally, the application of the complete multiset of evolution rules produces an output for \( \Pi \) in terms of a tuple with two elements (in the case when the external membrane of \( \Pi \) is not dissolved): the first one is the unitary set of evolved transition P system \( \Pi_{in} \), and the second case is the multiset of objects that \( \Pi_{in} \) sends to its father.

\[ \Pi_{\text{out}} = \{ ((l_{in}, \omega_{\text{intra}} \cup o_{\text{intra}}, (R_{in}, \rho_{in})), \Pi_{\text{intra}}) \]  \hspace{1cm} (52)

\[ o_{\text{out}} = o_{\text{preout}} \]  \hspace{1cm} (53)

The described algorithm is a recursive one in step three; it is necessary to wait for all descendent nodes in the transition P system tree to finish the application of \textbf{NEXT} function in order to continue. On the other hand, the function can be propagated in parallel to all descendent nodes as usual in concurrent programming. The algorithm is also based on the concept of transition P systems as trees of regions, which is equivalent to a tuple constituted by a region and a set of transition P systems; this permits to design a recursive function in order to simulate the evolution in such systems.

5 Conclusions

Based on algebraic consideration about the main components of transition P systems, we have developed a recursive algorithm that describes all possible transitions from a given configuration to any other possible configuration. The proposed algorithm to cover all the possibilities of evolution in transition P systems. However, transition P systems are non-deterministic systems, and this implies that there are several possibilities for evolution. The simulation of all evolutive paths in a P system could imply a combinatorial explosion. A real simulation must be based on some selection function in order to built only one possibility in a non-deterministic way. As one can easily see, some important concepts from the P systems design point of view, such as output membrane, skin membrane, and successful computation, have not been taken into account. However, we believe that from the simulation point of view these features are not so important.
References


