DESCRIPTIVE TIMED MEMBRANE PETRI NETS FOR PERFORMANCE MODELLING OF PARALLEL AND DISTRIBUTED COMPUTING

Emilian GUȚULEAC[1], Ion BALMUŞ[2], Iurie ȚURCANU[3], Emilia GUȚULEAC[4]
Computer Science Department, Technical University of Moldova
Bul. Stefan cel Mare nr. 168, MD-2004 Chisinau, Republic of Moldova
[1]egutuleac@mail.utm.md

Abstract. *In order to capture the compartmentalization and the behaviour of membrane systems for performance modelling of parallel and distributed computing, we introduce the Descriptive Timed Membrane Petri Nets (DM-nets) that can modify, in run-time, their own structure by rewriting some of their descriptive expression components. Furthermore, this descriptive approach facilitates the understanding of complex models and their component-based construction as well as the application of modern computer engineering concepts.*
Keywords: *membrane systems, parallel computing, performance modeling, Petri nets.*

Introduction

Recent technological achievements require advances beyond the existing computational models in order to be used effectively. Pragmatic aspects of current and future computer systems will be modelled so that realistic estimations of efficiency can be given for algorithms in these new settings.
Petri nets (PN) are very popular formalism for the analysis and representation of parallel and distributed computing in concurrent systems, that has drawn much attention to modeling and verification of this type of systems [1, 3].
P-systems, also referred to as membrane systems, are a class of parallel and distributed computing models inspired from the structure and the functioning of living cells [5].
The interest of relating P systems with the PN model of computation leads to several important results on simulation and decidability issues. Some efforts have been made to simulate P systems with Petri nets [4, 6] in order to verify the many useful behavioral properties such as *reachability*, *boundedness*, *liveness*, *terminating*, *deadlock-free*, etc.
In this paper we propose a new approach to define the components of continuous-time P systems [5] through components of Descriptive Petri Nets (DPN) using descriptive expressions (DE) [2]. The DE formalism is used for analytical representation and compositional construction of PN models.
In order to model specific rules of P Systems within the framework of the *Descriptive Rewriting Timed* PN (RTN) [3], we introduce a new extension of RTN - the *Descriptive Timed Membrane* PN, called DM-nets. It can modify dynamically its own structure by rewriting rules that determine structural dynamic changes of its components within P systems.

Labeled Extended Petri nets

In this section, we present a variant of PN called labeled extended PN. Let L be a set of labels $L = L_P \cup L_T$, $L_P \cap L_T = \varnothing$. Each place p_i labeled $l(p_i) \in L_P$ has a local state and the transition t_j being an action is labeled $l(t_j) \in L_T$.
Labeled PN is a structure $\Gamma = < P, T, Pre, Post, Test, Inh, G, Pri, K_p, l >$, where: P is the finite set of *places* and T is a finite set of *transitions* so that $P \cap T = \varnothing$. In the graphical representation, the place is drawn as a circle and the transition is drawn as a black bar; the *Pre*, *Test* and $Inh: P \times T \times IN_+^{|P|} \to IN_+$ is respectively a *forward flow, test* and *inhibition* functions and $Post: T \times P \times IN_+^{|P|} \to IN_+$ is a *backward flow* function in the multi-sets of P, which define the

set of arcs A and describes the marking-dependent cardinality of arcs that connect transitions and places. The *directed* arcs, *Pre* and *Post,* are drawn as single arrows. The *inhibitory* arcs, *Inh,* are drawn with a small circle at the end, and the *test* arcs, *Test,* are drawn as dotted single arrows. It does not consume the content of the source place. By default, the cardinality value of arcs is 1; $G : T \times IN_{+}^{|P|} \to \{True,\ False\}$ is the *guard function* of transitions. For $t \in T$, a guard function $g(t, M)$ is evaluated in each marking and if it is evaluated to *True*, the transition t is enabled, otherwise t is disabled (the default value is *True*); $Pri: T \times IN_{+}^{|P|} \to IN_{+}$ defines the *priority functions* for the firing of each transition and maps transitions into natural numbers IN_{+} representing their priority level. The enabling of a transition with higher priority disables all the lower priority transitions; $K_p : P \to IN_{+}$ is the capacity of places, being by default of infinite value; the $l : T \cup P \to L$, is a labeling function that assigns a label to transitions and places. In this way, transition names are mapped into action names so that $l(t_j) = l(t_k) = \alpha$ where $t_j \neq t_k$ and $l(p_i) = l(p_n) = \beta$ where $p_i \neq p_n$.

A marked labeled extended PN net is a pair $N = <\Gamma, M_0>$, where Γ is a labeled PN structure and M_0 is the initial marking of the net. $M : P \to IN_{+}$ is the current marking of the net which is described by a vector-column $M = (m_i p_i\ ,\ m_i \geq 0, \forall\ p_i \in P)$, where $m_i p_i$ is the number of tokens m_i in place p_i. M is the state of net that assigns to each place tokens represented by black dots.

Details concerning the enabling and firing rules, and the evolution of $N = <\Gamma, M_0>$ can be found in [2] as they require a great deal of space.

Descriptive expressions of Petri nets

Due to the space restrictions, we will give only a brief overview to this topic and refer the reader to [2, 3] and the references therein. In the following, because of abuse of notation, labels and names of nodes of the PN are the same.

We use the concept of a basic descriptive element (*bDE*) for a basic PN (*bPN*) introduced in [2] as the following:

$$bDE = |_{t_j}^{\alpha_j}\ m_i^0 p_i\ [W_i^+, W_i^-]\ |_{t_k}^{\alpha_k} . \qquad (1)$$

The translation of this *bPN* described by (1) is shown in figure 1a, where respectively $t_j = {}^{\bullet}p_i$ is input transition (action type α_j) and $t_k = p_i^{\bullet}$ is the output transition (action type α_k) of place $p_i \in P$ with the m_i^0 initial marking, and respectively the flow type relation functions $W_i^+ = \Pr e(t_j, p_i)$ and $W_i^- = Post(p_i, t_k)$, which return the multiplicity of input and output arcs of the discrete place $p_i \in P$. The derivative elements of *bDE* are for $p_i^{\bullet} = \varnothing,\ W_i^- = 0$ is $|_{t_j}^{\alpha_j}\ m_i^0 p_i\ [W_i]$ with final place p_i of t_j and for ${}^{\bullet}p_i = \varnothing,\ W_i^+ = 0$ is $m_i^0 p_i\ W_i\ |_{t_k}^{\alpha_k}$ with entry place p_i of t_k (see figure 1b). If the initial marking m_i^0 of place p_i is zero tokens we can omit $m_i^0 = 0$ in *bDE*. By default, if the type of action α is not mentioned this one matches the name of transition t. From a *bDE* we can build more complex *DE* of PN components by using composition operations.

Also, by default, if $W_i^+ = W_i^- = 1$, we present *bDE* and it derivatives as following:

$$|_{t_j}^{\alpha_j}\ m_i^0 p_i\ |_{t_k}^{\alpha_k}\ ,\ \ |_{t_j}^{\alpha_j}\ m_i^0 p_i \ \text{ or } \ m_i^0 p_i\ |_{t_k}^{\alpha_k} \qquad (2)$$

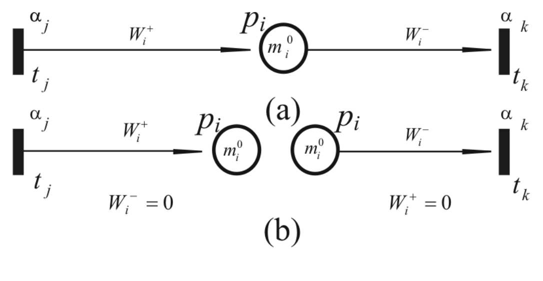


Figure 1. Translation in *bHN* (a) of *bDE* and (b) its derivatives.

A descriptive expression (*DE*) of a labeled PN is either *bDE* or a composition of *DE* a *N*:

$$DE ::= bDE \mid DE * DE \mid \circ DE\ ,$$

where $*$ is any binary composition operation and $\circ$ is any unary descriptive operation.

Descriptive Compositional Operations. In the following, by default the labels of *N* are encoded in the name of the transitions and places. The composition operations are reflected at the level of the *DE* components of *N* models by fusion of places, fusion of transitions with same type and same name (label) or sharing of subnets.

Place-Sequential Operation. This binary operation, denoted by the " | " *sequential operator*, determines the logic of an interaction between two local states p_i (pre-condition) and p_k (post-condition) by t_j action, which are in precedence and succeeding (causality-consequence) relation relative to this action. Thus, the expression:

$$DE1 = m_i^0 p_i\ [W_i] \,|_{t_j}^{\alpha_j}\, m_k^0 p_k\ [W_k]$$
$$\neq m_k^0 p_k\ [W_k] \,|_{t_j}^{\alpha_j}\, m_i^0 p_i\ [W_i] \qquad (3)$$

means the fact that the specified conditions (local state) associated with place-symbol p_i are always satisfied by means of the action t_j before the occurrence of the conditions associated with place-symbol p_k. Also, the PN modeling of the *iteration* operation is obtained by the fusion of head (entry) place with the tail (final) place that have the same name (*closing* operation) in *DE* which describes this net. The self-loop of *N*2 net described by:

$$DE2 = m_i^0 p_i\ [W_i] \,|_{t_j}^{\alpha_j}\, p_i\ [W_i] = m_i^0 \widetilde{p}_i\ [W_i] \,|_{t_j}^{\alpha_j} \qquad (4)$$

and represents the *test operator* "~", i.e. *test arc.*

Inhibition Operation. This unary operation is represented by *inhibitory operator* " ¯ " (represented by place-symbol with overbar). $DE3 = m_i^0 \overline{p}_i\ [W_i] \,|_{t_j}^{\alpha_j}$ describes the inhibitor arc in *N* models with a weight function $W_i = Inh(p_i, t_j)$.

Synchronization Operation. This binary operation, represented by "•" or "$\wedge$" *join* operator, describes the rendez-vous synchronization (by transition t_j) of two or more conditions represented respectively by symbol-place $p_i \in {}^\bullet t_j$, $i = 1,..n$, i.e. It indicates that all preceding conditions of occurrence actions must have been completed.

Split Operation. This binary operation, represented by the "$\Diamond$" or "($\Diamond$)" *split* or *fork* operators, describes, determines the causal relations between activity t_j and its post-conditions: after completion of the preceding action of t_j concomitantly, several other post-condition can occur in parallel.

Competing Parallelism Operation. This compositional binary operation is represented by the *"$\vee$ "* competing parallelism *operator*, and it can be applied over N_A with $DE_A = A$ and N_B with $DE_B = B$ or internally into resulting N_R with $DE_R = R$, between the places of a single N_R where the symbol-places with the same name are respectively fused. We can represent the resulting DE_R = $R = A \vee B$ as a set of ordered pairs of places with the same name to be fused, with the first element belonging to *A* and the second to *B*. The fused places will inherit the arcs of the place in *A* and *B*.

Precedence Relations between the Operations. We introduce the following *precedence relation* between the compositional operations in the *DE*: a) the evaluation of operations in *DE* are applied left-to-right; b) an unary operation binds stronger than a binary one; c) the "•"operation is superior to "/" and "$\Diamond$" is superior to "$\vee$".

Further details on definitions, enabling and firing rules, and evolution for of *N* can be found in [3] as they require a great deal of space.

Dynamic Rewriting Petri Nets

In this section we introduce the model of *descriptive dynamic net rewriting* PN system. Let $X \rho Y$ be a binary relation. The *domain* of ρ is the *Dom*(ρ) $= \rho Y$ and the *codomain* of ρ is the *Cod*(ρ) $= X\rho$. Let *A*= < *Pre*, *Post*, *Test*, *Inh* > be a set of arcs that belong to net Γ.

Definition 1. A *descriptive dynamic rewriting* PN system is a structure:

$RN = < \Gamma,\ R,\ \phi, G_{tr}\ G_r,\ M >$, where Γ = < *P*, *T*, *Pre*, *Post*, *Test*, *Inh*, *G*, *Pri*, K_p, *l* >; $R = \{r_1, ..., r_k\}$ is a finite set of rewriting rules about the runtime structural modification of net so that $P \cap T \cap R = \varnothing$.

In the graphical representation, the rewriting rule is drawn as two embedded empty rectangles. We let $E = T \cup R$ denote the set of *events* of the net; $\phi : E \rightarrow \{T, R\}$ is the function that indicates for every rewriting rule the type of event can occur; $G_{tr} : R \times IN_{+}^{|P|} \rightarrow \{True, False\}$ is the *transition rule guard function* associated with $r \in R$, and $G_{r} : R \times IN_{+}^{|P|} \rightarrow \{True,\ False\}$ is the *rewriting rule guard function* defined for each rule of $r \in R$, respectively. For $\forall r \in R$, the $g_{tr}(M) \in G_{tr}$ and $g_{r}(M) \in G_{r}$ will be evaluated in each marking and if they are evaluated to *True*, the rewriting rule *r* is *enabled*, otherwise it is *disabled*. Default value of $g_{tr}(M) \in G_{tr}$ is *True* and for $g_{r}(M) \in G_{r}$ is *False*). Let $RN = < R\Gamma,\ M >$ and $R\Gamma = < \Gamma,\ R, \phi, G_{tr}\ G_{r} >$ be represented by the descriptive expression $DE_{R\Gamma}$ and DE_{RN}, respectively. A dynamic rewriting structure modifying the rule $r \in R$ of *RN* is a map $r : DE_{L} \triangleright DE_{W}$, where the *codomain* of the rewriting operator, $\triangleright$, is a fixed descriptive expression DE_{L} of a subnet RN_{L} of current net *RN*, where $RN_{L} \subseteq RN$ with $P_{L} \subseteq P$, $E_{L} \subseteq E$ and set of arcs $A_{L} \subseteq A$ and the *domain* of $\triangleright$ is a descriptive expression DE_{W} of a new RN_{w} subnet with $P_{w} \subseteq P$, $E_{w} \subseteq E$ and set of arcs A_{W}. The rewriting operator, $\triangleright$, represents a binary operation which produces a *structure change* in DE_{RN} of the *RN* net by replacing (rewriting) the fixed current DE_{L} of the subnet RN_{L} (DE_{L} and RN_{L} are dissolved) by the new DE_{W} of subnet RN_{W} that now belongs to the new modified resulting $DE_{RN'}$ of the net $RN' = (RN \setminus RN_{L}) \cup RN_{w}$ with $P' = (P \setminus P_{L}) \cup P_{W}$ and $E' = (E \setminus E_{L}) \cup E_{W}$, $A' = (A - A_{L}) + A_{W}$ where the meaning of \ ($\cup$) is operation of removing (adding) RN_{L} from (RN_{W} to) net *RN*. In this new net RN', obtained by the execution of the enabled rewriting rule $r \in R$, the places and events with the same attributes which belong to RN' are fused. By default, the rewriting rules $r : DE_{L} \triangleright \varnothing$ or $r : \varnothing \triangleright DE_{W}$ describe the rewriting rule which maintains $RN' = (RN \setminus RN_{L})$ or $RN' = (RN \cup RN_{W})$. A state of a *RN* net is the pair ($R\Gamma, M$), where $R\Gamma$ is the configuration of the net together with a current marking *M*. Also, the pair ($R\Gamma_{0}, M_{0}$) with $P_{0} \subseteq P$, $E_{0} \subseteq E$ and marking M_{0} is called the initial state of the net. ■

Enabling and Firing of Events. The enabling of events depends on the marking of all places. We say that a transition t_{j} of the event e_{j} is enabled in current marking *M* if the following enabling condition $ec(t_{j}, M)$ is verified:

$$ec(t_{j},M) = (\underset{\forall p_{i} \in {}^{\bullet}t_{j}}{\wedge}(m_{i} \geq Pre(p_{i},t_{j}))\ \&\ \underset{\forall p_{k} \in {}^{\circ}t_{j}}{\wedge}(m_{k} < Inh(p_{k},t_{j}))\ \&\ \underset{\forall p_{l} \in {}^{*}t_{j}}{\wedge}(m_{l} \geq Test(p_{l},t_{j}))\ \&\ \underset{\forall p_{n} \in t_{j}^{\bullet}}{\wedge}((K_{p_{n}} - m_{n}) \geq Post(p_{n},t_{j}))\ \&\ g(t_{j},M)) \quad (5)$$

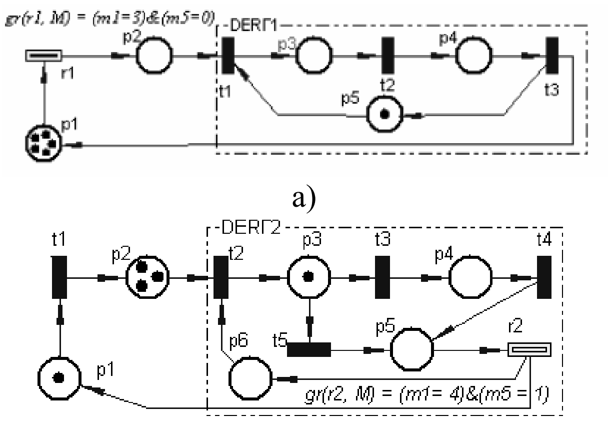
The rewriting rule $r_{j} \in R$ is enabled in current marking *M* if the following enabling condition $ec_{tr}(r_{j}, M)$ is verified:

$$ec_{tr}(r_{j},M) = (\underset{\forall p_{i} \in {}^{\bullet}r_{j}}{\wedge}(m_{i} \geq Pre(p_{i},r_{j}))\ \&\ \underset{\forall p_{k} \in {}^{\circ}r_{j}}{\wedge}(m_{k} < Inh(p_{k},r_{j}))\ \&\ \underset{\forall p_{l} \in {}^{*}r_{j}}{\wedge}(m_{l} \geq Test(p_{l},r_{j}))\ \&\ \underset{\forall p_{n} \in r_{j}^{\bullet}}{\wedge}((K_{p_{n}} - m_{n}) \geq Post(p_{n},r_{j}))\ \&\ g_{tr}(r_{j},M)) \quad (6)$$

Let the *T*(*M*) and *R*(*M*), $T(M) \cap R(M) = \varnothing$, be the set of enabled transitions and rewriting rules in current marking *M*, respectively. Let the $E(M) = T(M) \cup R(M)$, be the set of enabled events in a current marking *M*. The event $e_{j} \in E(M)$ fires if no other event $e_{k} \in E(M)$ with higher priority has been enabled. Hence, for e_{j} event *if* $((\phi_{j} = t_{j}) \vee (\phi_{j} = r_{j}) \wedge (g_{tr}(r_{j},M) = False))$ *then* the firing of the transition $t_{j} \in T(M)$ or of the rewriting rule $r_{j} \in R(M)$ changes only the current marking:
$(R\Gamma,\ M) \xrightarrow{e_{j}} (R\Gamma,\ M') \Leftrightarrow (R\Gamma = R\Gamma$ and the $M[e_{j} > M'$ in $R\Gamma$)). Also, for the event e_{j}, *if* $((\phi_{j} = r_{j}) \wedge (g_{r}(r_{j},M) = True))$ *then* the event e_{j} occurs at firing of the rewriting rule r_{j} and it changes the configuration and marking of the

current net, so that: $(R\Gamma, M)\xrightarrow{r_j}(R\Gamma', M')$, $M[r_j > M')$.

The accessible state graph of a net $RN = < R\Gamma, M >$ is the labeled directed graph whose nodes are states and whose arcs, which are labeled with events or rewriting rules of RN, are of two kinds: a) *firing* of an enabled event $e_j \in E(M)$: arcs from state $(R\Gamma, M)$ to state $(R\Gamma, M')$ labeled with event e_j, so that this event can fire in the configuration $R\Gamma$ at marking M and leads to a new marking $M' : (R\Gamma, M)\xrightarrow{e_j}(R\Gamma', M') \Leftrightarrow (R\Gamma = R\Gamma'$ and $[M[e_j > M'$ in $R\Gamma$); b) *change configuration*: arcs from state $(R\Gamma, M)$ to state$(R\Gamma', RM')$ labeled with the rewriting rule $r_j \in R$, $r_j : (R\Gamma_L, M_L) \triangleright (R\Gamma_W, M_W)$ which represent the change configuration of current RN net: $(R\Gamma, M)\xrightarrow{r_j}(R\Gamma', M')$ with $M[r_j > M'$.



a)

b)

Figure 2: Translation: a) $DE_{R\Gamma 1}$ in *RN*1 and b) $DE_{R\Gamma 2}$ in *RN*2.

Let we consider the *RN*1 net, given by the following descriptive expression:

$$DE_{R\Gamma 1} = p_1 |_{r_1} p_2 \vee DE'_{R\Gamma 1}$$

$$DE'_{R\Gamma 1} = (p_2 \cdot p_5)|_{t_1} p_3 |_{t_2} p_4 |_{t3} (p_1 \Diamond p_5) \quad (7)$$

$$M_0 = (5p_1, 1p_5),\ r_1 : DE_{R\Gamma 1} \triangleright DE_{R\Gamma 2}$$

$$g_r(r_1, M) = (m_1 = 3) \& (m_5 = 0)$$

Also, for the rewriting rule r_j is required to identify if RN_L net belongs to the $R\Gamma$. The firing of enabled events or rewriting rules modify the current marking and/or modify the structure and the current marking of *RN*1 in *RN*2 given by:

$$DE_{R\Gamma 2} = p_1 |_{t_1} p_2 \vee DE'_{R\Gamma 2},$$

$$DE'_{R\Gamma 2} = (p_2 \cdot p_6)|_{t_2} p_3 (|_{t_3} p_4 |_{t_4} p_5 \vee |_{t_5} p_5 |_{r_2} (p_1 \Diamond p_6)),$$

$$r_2 = r_1^{-1} : DE_{R\Gamma 2} \triangleright DE_{R\Gamma 1}, M = (1p_1, 3p_2, 1p_3),$$

$$g_r(r_2, M) = (m_1 = 4) \& (m_5 = 1) \quad (8)$$

The translation of expression (7) $DE_{R\Gamma 1}$ in *RN*1 and expression (8) $DE_{R\Gamma 2}$ in *RN*2 is shown in figure 2a and figure 2b, respectively.

Dynamic Rewriting Timed Petri Nets

Systems are described in timed PN (TPN) as interactions of components that can perform a set of activities associated with events. An event $e = (\alpha, \theta)$, where $\alpha \in E$ is the type of an activity (action name), and θ is the firing delay.

Definition 2. A descriptive dynamic rewriting TPN is a *RTN* = < *RN*, θ >, where:

- $RN = < \Gamma, R, \phi, G_{tr}\ G_r, M >$, $\Gamma = < P, T,$ *Pre*, *Post*, *Test*, *Inh*, *G*, *Pri*, K_p, l > with set of events $E = E_0 \cup E_\tau$ which can be partitioned into a set E_0 of *immediate* events and a set E_τ of *timed* events, so that $E_0 \cap E_\tau = \varnothing$, *Pri*($E_0$) > *Pri*($E_\tau$). The immediate event is drawn as a thin bar and the timed event is drawn as a black rectangle for transitions and two embedded empty rectangles for rewriting rules;
- $\theta : E \times IN_+^{|P|} \rightarrow IR_+$ is the weight function that maps events onto real numbers IR_+ (delays or weight speeds). It can be dependent of the marking. The delays $\theta(e_k, M) = d_k(M)$ defines the duration of timed events.

If several timed events $e_j \in E(M)$ are concurrently enabled in current marking for the $e_j \in {}^\bullet p_i = \{\forall e_j \in E : \mathrm{Pr}\,e(p_i, e_j) > 0\}$, either in competition or independently, we assume that a *race competition* condition exists between them. The evolution of the model will determine whether the other timed events have been aborted or simply interrupted by the resulting state change. The $\theta(e_j, M) = w_j(M)$ is the weight speed of immediate events $e_j \in E_0$. If several enabled immediate events are scheduled to fire at the same time in *vanishing* marking M with the weight speeds, then the probability of the enabled immediate event e_j to *fire* is:

$$q_j(M) = w(e_j, M) / \sum_{e_l \in (E(M) \& {}^{\bullet}p_i)} w(e_l, M) \qquad (9)$$

where $E(M)$ is the set of enabled events in M. An immediate event has a zero firing time. ■

P Systems and Timed Membrane Petri Nets

Here we give a brief review of P systems and its encoding with DM-nets. A full guide for P systems can be referred to [4]. The main components of P systems are membrane structures consisting of hierarchically embedded membranes in the outermost skin membrane. Each membrane encloses a region containing a multiset of objects and possibly other membranes.

In general, a basic evolution-communication P system with active membranes (of degree $n \geq 0$) is $\mathit{\Gamma P} = (O, H, \mu, \Omega, (\rho, \pi))$, where: O is the alphabet of objects; H is a finite set of labels for membranes; μ is a membrane structure consisting of n membranes labeled with elements h = 0,1,..., n-1 in H; Ω is the configuration, that is mapping from membrane h of $\mathit{\Gamma P}$ (nodes in μ) to multisets of objects $\omega_h \in \Omega$, $\omega_h = \{\omega_{h,i}\}$, i=0,1,...,$|O|$ present in the corresponding region of membrane h, then the system is created; ρ and π is respectively the set of rules $\rho_h = \{\rho_{h,j}\}$ and its priorities $\pi_h = \{\pi_{h,j}\}$, j=0,1,..., k. The active membranes can be of two forms of rules $\rho_{h,j}$: a) the *object rules* (OR), i.e., evolving and communication rules concerning the objects; b) the *membranes rules* (MR), i.e., the rewriting rules about the structural modification of membranes.

The structure of a membrane is: $[_h\, \omega, [_i\ ]_i, ..., [_l\ ]_l\ ]_h$, and each object rule $\rho_{h,j} \in \rho$ has a form: $\rho_{h,j} : \omega \to \omega_{here}\omega_{out}\omega_{in_l} \times \{\delta, \neg\delta\}$, here l belongs to the label set of its sub-membranes, and $\omega_{here}, \omega_{out}$, and ω_{in_l} denote the multiset of objects which will be kept in this membrane, send out of this membrane, and send into its sub-membrane labeled by l, respectively. The δ denotes the dissolving rule.

Here we present the *DM-nets* for encoding of P systems mentioned above into descriptive dynamic rewriting TPN as a *RTN*. The basis for *DM-nets* is a membrane *RTN* that is a DE net structure comprising: places and its capacity, transitions and its priority, and guard function, weighed directed arcs from places to transitions and vice-versa, weighed inhibitory and test arcs. Consider the P system $\mathit{\Gamma P}$. The encoding of $\mathit{\Gamma P}$ into DM-net is decomposed into two separate steps. First, for every membrane $[_h\ ]_h$ we associate: to each object $\omega_{h,i} \in \omega_h$ one place $[_h\ m_i^0 p_{h,i}\ ]_h$ with the initial marking $m_{h,i}^0$, and to each rule $\rho_{h,j} \in \rho$ one event $[_h\ e_{h,j}\ ]_h$, that acts on the this membrane. Second, for every membrane $[_h\ ]_h$ we define the DE_h of RTN_h that corresponds to the initial configuration of the P system $\mathit{\Gamma P}$ as $[_h\ DE_h\ ]_h$.

Let u, v and u', v' be a multiset of objects.

Definition 3. The *DM-nets* of degree $n \geq 0$, is a construct $DM = \vee_{h=0}^{n-1} [_h\ DE_h^0\ ]_h$, where:

- The $\rho_{h',j} : [_h\ [_{h'}\ u \to v\ ]_{h'}\ ]_h$ *evolving* object rule with multiset of objects u, v which will be kept in $[_{h'}\ ]_{h'}$ is encoded as:

$$[_h\ [_{h'}\ p_{h,u}\ |_{th,j}\ p_{h,v}\ ]_{h'}\ ]_h\ ;$$

- The *antiport* rule, that realizes a synchronization of object c with the objects of the type $\rho_{h',i} : [_h u\, [_{h'}\ v\ ]_{h'}\ ]_h \to [_h v'\, [_{h'}\ u'\ ]_{h'}\ ]_h$, is encoded as:

$$[_h\ [_{h'}\ (p_{h',u} \cdot p_{h',v} \cdot \widetilde{p}_{h',c})\ |_{th',i}\ (p_{h',u'} \Diamond p_{h',v'})\ ]_{h'}\ ]_h\ .$$

Also, the $\rho_{h',k} : [_h u\ [_{h'}\ ]_{h'}\ ]_h \to [_h\ [_{h'}\ u'\ ]_{h'}\ ]_h$ *symport* rule, that moves objects from inside to outside a membrane, or vice-versa is encoded as:

$$[_h\ [_{h'}\ (p_{h',u} \cdot \widetilde{p}_{h',c})\ |_{th',k}\ p_{h',u'}\ ]_{h'}\ ]_h\ .$$

Because the configuration means both a membrane structure and the associated multisets of objects, we need the rewriting rules for

processing membranes and multisets of objects as: $MR = \{mr_0, mr_1, mr_2, mr_3, mr_4, mr_5, mr_6\}$.
The above membrane rewriting rules (realized by the rewriting events in *DE's*) are defined as:

- mr_0: *Change* rewriting rule, that changes, in run-time, the current structure and the multisets of objects of membrane *h*, encoded by descriptive expression $DE_{h'}$ and its marking $M_{h'}$ in a new structure $DE'_{h'}$ with new marking $M'_{h'}$: $[_h\, [_{h'}\, DE_{h'}\,]_{h'}\,]_h \triangleright [_h\, [_{h'}\, DE'_{h'})\,]_{h'}\,]_h$;
- mr_1: *Dissolve* rewriting rule says that the membrane h' is *dissolved* and the objects as $M_{h'}$ and sub-membranes of membrane h' belong now to its parent membrane *h*. The skin membrane cannot be *dissolved*:
$[_h DE_h\, [_{h'}\, DE_{h'}\,]_{h'}\,]_h \triangleright [_h\, DE'_h\,]_h$, $M'_h = M_h + M_{h'}$;
- mr_2: *Create* rewriting rule, says that the new membrane h' with $DE''_{h'}$ and $M''_{h'}$ is created in membrane *h*, the rest remain in the parent membrane *h*:
$[_h\, DE_h\,]_h \triangleright [_h DE'_h\, [_{h'}\, DE''_{h'}\,]_{h'}\,]_h, M_h = M'_h + M''_{h'}$;
- mr_3: *Divide* rewriting rule says that the objects and sub-membranes are reproduced and added into membrane h' and into h'', respectively:
$[_h\, DE_h\,]_h \triangleright [_h\, [_{h'}\, DE_h\,]_{h'}\, [_{h''}\, DE_h\,]_{h''}]_h$;
- mr_4: *Merge* rewriting rule says that the objects of membrane h' and h'' are added to a new membrane h is:
$[_h [_{h'}\, DE'_{h'}\,]_{h'}\, [_{h''}\, DE''_{h''}\,]_{h''}]_h \triangleright [_h\, DE'_{h'} \vee DE''_{h''}\,]_h$
with new marking $M'_{h'} + M''_{h''} = M_h$;
- mr_5: *Separate* rewriting rule is the counterpart of the *Merge* rewriting rule and is done by a:
$[_h\, DE'_{h'} \vee DE''_{h''}\,]_h \triangleright [_h [_{h'}\, DE'_{h'}]_{h'}\, [_{h''} DE''_{h''}\,]_{h''}]_h$
with meaning that the content of membrane h is split into two membranes, with labels h' and h'', and the new marking is $M_h = M'_h + M''_{h'}$;
- mr_6: *Move* rewriting rule where a membrane h'' can be moved out or moved into a membrane h' as a whole is done by a:

$$[_h\, [_{h'} DE_{h'}\, [_{h''} DE_{h''}\,]_{h''}]_{h'}]_h \triangleright [_h\, [_{h'} DE_{h'}\,]_{h'}\, [_{h''}\, DE_{h''}\,]_{h''}]_h \quad \text{or}$$
$$[_h\, [_{h'}\, DE_{h'}\,]_{h'}\, [_{h''} DE_{h''}\,]_{h''}]_h \triangleright [_h [_{h'}\, DE_{h'}\, [_{h''}\, DE_{h''}\,]_{h''}]_{h'}]_h$$

with their marking, respectively. ■
Thus, using the *DM-nets* facilitates a compact and flexible specification and verification of parallel computing models.
In order to describe the details of this approach, we present a simple but illustrative example of encoding $\mathit{\Gamma P}$ into *DM-net*.
Consider the following P system $\mathit{\Gamma P}1$ of degree 3:

$$\mu = [_0\ b,\ [_1\ a, [_2\,]_2,]_1,]_0,\ O = \{a,b,c,d\},$$
$$\omega_0 = \{b\}, \omega_1 = \{a\}, \rho_0 = \{\rho_{0,1} : c \rightarrow dd_{in2},$$
$$\rho_{0,2} : b \rightarrow b_{in1}\},\ \pi_0 = \{\pi_{0,1} > \pi_{0,2}\}, \tag{10}$$
$$\rho_1 = \{\rho_{1,1} : a \rightarrow b_{here} c_{out} d_{in2},\ \rho_{1,2} : b \rightarrow a_{out}\delta\},$$
$$\rho_{1,2} : b \rightarrow a_{out}\delta\},\ \pi_1 = \varnothing, \omega_2 = \varnothing,\ \pi_2 = \varnothing.$$

The encoding solution for the initial configuration of $\mathit{\Gamma P}1$ described by (10) is given by the *DM1-net*, where every object can be represented as a place labeled as the name of objects: $l(p_{0,1}) = l(p_{1,1}) = a$, $l(p_{0,2}) = l(p_{1,2}) = b$,

$$l(p_{0,3}) = c,\ l(p_{2,1}) = l(p_{0,4}) = d.$$

The number of tokens in this place denotes the number of occurrences of this object.
Every object- rule can be represented by an event type transition. For example, in membrane 0, the rule $\rho_{0,1} : c \rightarrow dd_{in2}$, can be described by a transition $t_{0,1}$. Because two copies of object *d* are send to membrane 2, the weight of the arc $(t_{0,1}, p_{2,1})$ is 2, which denotes that whenever the rule $\rho_{0,1}$ is performed, one copy of object *c* will be removed in membrane 0 and two copies of object *d* will be sent to membrane 2.
Up to now, all objects and rules of $\mathit{\Gamma P}1$ are encoded in *DM1-net* as following:

$$DM1 = [_0\ DE_0\ [_1\ DE_1\ [_2\, DE_2\,]_2\ ]_1\ ]_0$$
$$DE_0 = p_{0,1} \vee 1p_{0,2}\, |_{t_{0,2}}\ p_{2,1} \vee p_{0,3}\, |_{t_{0,1}}\ p_{2,1}[2]$$
$$DE_1 = 1p_{1,1}\, |_{t_{1,1}}\ (p_{0,3} \Diamond p_{1,2}\, |_{r_{1,1}}\ p_{0,1} \Diamond p_{2,1}) \tag{11}$$
$$DE_2 = p_{2,1}\, |_{t_{2,1}}\ p_{1,2},\ \Pr i(t_{0,1})\,) > \Pr i(t_{0,2})$$
$$DE1 = DE_0 \vee DE_1 \vee DE_2$$

The dissolving rule $mr_1 = \delta$ is represented in *DM1-net* by the following mr_1 rule:

$$r_{1,1} :\ DM\ 1 \triangleright DM'1, DM'1 = [_0\, DE'_0\,]_0 \tag{12}$$
$$DE'_0 = 1p_{0,1} \vee p_{0,3}\, |_{t_{0,1}}\ 2p_{0,4}[2]\, |_{t_{0,2}}\ 2p_{0,2},$$

$$gr(r_{1,1}, M^{DE1}) = (M^{DE1} = (p_{0,3}, 2p_{1,2}, 1p_{2,1}))$$

where M^{DE1} is the current marking of $DM1$.
The translation of P system $\Gamma P1$ into *DM1-net* described by (11) and *DM'1-net* described by (12) is shown in figure 3 and figure 4, respectively.

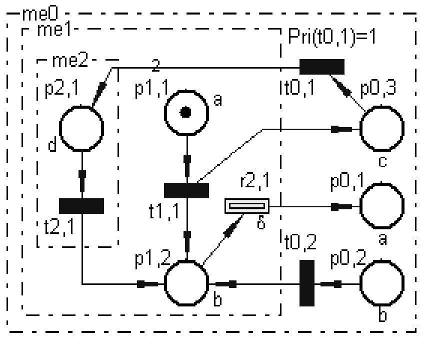


Figure 3. Translation of $DE1$ into *DM1-net* for $\Gamma P1$.

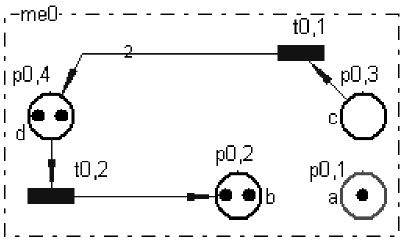


Figure 4: Translation of $DE'1$ into $DM'1$*-net* for $\Gamma P1$.

The reachability graph of *DM1-net* in the listing form is:

$M_0^{DE1} = (1p_{0,2}, 1p_{1,1})\, [U_1 > M_1^{DE1}$;

$M_1^{DE1} = (1p_{0,3}, 2p_{1,2}, 1p_{2,1})\, [U_2 > M_2^{DE'1}$;

$M_2^{DE'1} = (1p_{0,1}, 2p_{0,2}, 2p_{0,4})\, [U_3 > M_3^{DE'1}$;

$M_3^{DE'1} = (1p_{0,1}, 3p_{0,2}, 1p_{0,4})\, [U_4 > M_4^{DE'1}$;

$M_4^{DE'1} = (1p_{0,1}, 4p_{0,2})\, [$,

where $U_1 = \{t_{0,2}, t_{1,1}\}$; $U_2 = \{t_{0,1}, r_{1,1}, t_{2,1}\}$; $U_3 = U_4 = \{t_{0,3}\}$.

Conclusions

In this paper we have proposed an approach to the performance modeling of the behaviour of P systems through a class of Petri nets, called Descriptive Membrane Timed PN (DM-nets).

Based upon the introduction of a set of descriptive composition operation and rewriting rules attached to transitions for the creation of dynamic rewriting TPN, the membrane structure can be successfully encoded as a timed membrane Petri nets models which permits the description of the state based process, in run-time structure change of P systems, and verification of its behavioral properties.

An important feature characterizing the proposed basic DM-net model is its robustness, in the sense of being easily extendable to handle salient features of more sophisticated active membrane systems.

We are currently developing a visual simulator software with a friendly interface for verifying and performance evaluation of descriptive rewriting TPN models and DM-nets.

References

[1] Ajmone-Marsan, M., Balbo G., Conte G., Donatelli S., and Francheschinis G. (1995) *Modeling with Generalized Stochastic Petri Nets*, In Parallel Computing, New York: Wiley.

[2] Guţuleac, E. (2004) *Descriptive Compositional Construction of GSPN Models for Performance Evaluation of Computer Systems*, In Proc. of the 8-th International Symposium SACCS04, 22-23 October, Iasi, România, CD.

[3] Guţuleac, E., Mocanu M. (2005) *Descriptive Dynamic Rewriting GSPN-based Performance Modeling of Computer Systems*, In Proc. of the 15th Intern. Conf. CSCS15, 25-27 May 2005, Bucureşti, România, pp. 656-661.

[4] Kleijn J., Koutny M., Rozenberg G. (2005) *Towards a Petri Net Semantics for Membrane Systems.* In Proceedings of the WMC6 2005, July 18-21, Wien, Austria, pp. 439-459.

[5] Păun, Gh. (2002) *Membrane Computing. An Introduction*, Natural computing Series. ed. G. Rozenberg, Th. Back, A.E. EibenJ.N. Kok, H.P. Spaink, Leiden Center for Natural Computing, Springer–Verlag, Berlin, p. 420.

[6] Qi Z., You J., and Mao H. (2003) *P Systems and Petri Nets*, Proceedings WMC 2003, Lecture Notes in Computer Science, vol. 2933, Springer-Verlag, Berlin, pp. 387-403.