MEMBRANE DIFFERENTIAL PETRI NETS FOR PERFORMANCE MODELLING OF HYBRID P-SYSTEMS

Emilian Guțuleac¹, Mihai L. Mocanu², Iurie Țurcanu¹

¹ Computer Science Department, Technical University of Moldova, Bvd. Ștefan cel Mare nr. 168, Chișinău, Republic of Moldova, E-mail: egutuleac@mail.utm.md
² Department of Software Engineering, Faculty of Automation, Computers and Electronics, Bvd. Decebal nr. 107, University of Craiova, ROMANIA, E-mail: mocanu@software.ucv.ro

Abstract: P-systems, also referred to as membrane systems, are a class of computing models inspired from the structure and functioning of living cells. Efforts have been made to simulate P-systems with Petri nets, most oriented to verify their behavioral properties. In this paper, we introduce a new methodology based on dynamic rewriting Generalized Differential Petri Nets (GDPN) for performance modeling of P-systems. We define the concept of hybrid P-systems with active membranes and give its formalization based on the new approach - the Timed Membrane GDPN (also called DMH-net). An example is presented to illustrate our approach.

Keywords: P-systems, reconfigurable and mobile hybrid systems, Petri Nets, performance modelling.

1. INTRODUCTION

Hybrid systems are dynamic systems exhibiting both continuous and discrete dynamic behavior. In general, a hybrid system can be described by a few pieces of information. The state of the system consists of vector signals, which can change according to dynamic laws in the system data that includes: a flow equation describing the continuous dynamics, a flow set in which flow is permitted, a jump equation describing the discrete dynamics, and a jump set in which discrete state evolution is permitted. Initially, the term "hybrid system" denoted systems combining electrical and mechanical drivelines, and employing neural nets and fuzzy logic for modeling. But the introduction of this new category in modeling quickly brought up the benefit of having a larger class of systems within a dynamic structure that allows for far more flexibility. Thus, their area of application has been extended to many other fields. Recent observations on living organisms led to the development of a class of hybrid systems called P-systems, also referred as membrane systems.
Generally, discrete-continuous modeling and simulation is concerned with the description, analysis and optimization of the dynamic behavior encompass of hybrid systems. Among the formalisms that are used for modeling, the hybrid Petri nets (HPN) (Ala and David, 1998) and Differential Petri Nets (DPN) (Demongodin and Koussoulas, 1998) are very popular. For the simulation of P-systems, Petri nets (Qi and Mao, 2003) have been employed mostly to aid in verifying the many useful behavioral properties such as reachability, boundedness, liveness, etc.

We propose in this paper a new approach to express the components of continuous variant of P-systems through components of dynamic rewriting GDPN described in (Gutuleac, et al., 2006), using the descriptive expressions (DE) introduced in (Gutuleac and Mocanu, 2005). Generally, in rewriting P-systems Chmsky rules (Martin-Vide and Păun, 2000) or contextual rules (Păun, 1997) have been employed for processing string objects. Our approach is different and based on the original concepts mentioned above. To make design issues and analysis procedures more transparent, we tried to devise a little as possible from the concepts and the nets of timed HPN and DPN. Thus, we created our extension of DPN, which we call Generalized DPN (GDPN), and that is able to represent the behavior of hybrid systems in a common model. The features of GDPN accept the negative-continuous place capacity, negative real values for continuous place marking and negative token-dependent arc cardinalities of continuous part that permit to generalize the concept of timed HPN and DPN (Gutuleac, et al., 2006). Also, in order to capture the localities and the behavior of reconfigurable and mobile hybrid systems, we introduce hybrid P-systems with active membranes and present a formalization of this concept, the Timed Descriptive Membrane GDPN, based on the new approach. Also called DMH-nets, these are nets that can modify in run-time their own structure by rewriting some rules of their descriptive expressions components.

2. P-SYSTEMS AND MEMBRANE COMPUTING

The theory of P-systems, initiated by Păun about a decade ago and described in (Păun, 2002) is developing “on top” of a class of parallel and distributed computing models, able to reproduce the structure and functioning of living cells. Membrane computing, the field that covers the study of P-systems, is based on observations on processes taking place in the complex structure of a living cell – their evolution can be regarded as a computation driven by chemical reaction. Hybrid models on P-systems are allowed by the switch-like character of control at the molecular level (similar to genetic regulatory networks) for the development of living organisms.

The basic components of P-system are a membrane structure consisting of several cell-like membranes placed inside a main (outer most) membrane called skin. Graphically, a membrane structure is represented by a Venn diagram without intersection and with a unique superset. If a membrane does not contain any other membrane, it is called elementary. The membranes delimit regions (spaces between a membrane and all directly inner membranes - if any inner membrane exists) where objects are placed. Object symbols can be rewritten in a non-deterministic or maximally parallel manner, parallel operations can be performed on all available objects in a membrane system, and the objects can also be communicated from one region to another. In this manner, the configuration of the system can change, starting from an initial configuration and using the evolution rules; thus, we get a computation.

Since the P-systems were firstly introduced, a number of classes of P-systems have been proposed and investigated. Efforts have been made in the last few years in many directions such as the description, analysis and optimization of the dynamic behavior or the formal verification of the dynamical properties of these systems. These efforts rely on the use of discrete abstractions and models checking, and have been designed to cope with aspects one often encounters when analyzing these systems, like large uncertainties on the parameter values.

Some similarities between the Petri net theory and P-systems which concern the approaches to processes generated by concurrent systems may provide a form of “transfer” of results from Petri net theory into P-systems theory. The next sections of the paper will review some of our previous results on Petri nets previously applied to other classes of systems, with a potential to facilitate an algebraic characterization of some P systems. Descriptive composition operations and
expressions will allow for the creation of reconfigurable GDPN models for P-systems.

3. LABELED GDPN

Let \( L \) be a set of labels \( L = L_p \cup L_T, \)
\( L_p \cap L_T = \emptyset \). Each place \( p_i \), labeled \( l(p_i) \in L_p \)
has a local state and every transition \( t_j \) is action
labeled as \( l(t_j) \in L_T \).

Definition 1: A labeled GDPN is a 11-tuple
\[ H^e = \langle P, T, Pre, Post, Test, Inh, K_p, \ K_b, \ G, \ Pri, l, \rangle \]
where:
- \( P \) is the finite set of places partitioned into a
set of discrete places \( P_D \), and a set of continuous
places \( P_C \), \( P = P_D \cup P_C \), \( P_D \cap P_C = \emptyset \). The
discrete places may contain a natural number of
tokens, while the marking of a continuous
place is a real number (fluid level). In the
 graphical representation, a discrete place is
drawn as a single circle while a continuous place
is drawn with two concentric circles;
- \( T \) is a finite set of transitions, that can be
partitioned into a set \( T_D \) of discrete transitions
and a set \( T_C \) of continuous transitions, \( T
= T_D \cup T_C \), \( T_D \cap T_C = \emptyset \). A discrete transition
\( t_j \in T_D \) is drawn as a black bar; a continuous
transition \( t_j \in T_C \) is drawn as a rectangle;
- \( Pre, Test \) and \( Inh: P \times T \rightarrow Bag(P) \)
respectively, are forward flow, test and
inhibition functions. \( Bag(P) \) is a discrete or
real-valued multisets functions over \( P \). The backward
flow function in the multisets of \( P \)
is \( Post: T \times P \rightarrow Bag(P) \). These functions define
the set of arcs \( A \) and it describes the marking-
dependent cardinality of arcs connecting
transitions with places and vice-versa. Also, the
\( A \) set is partitioned into subsets: \( A_d, A_s, A_h, \)
\( A_c \) and \( A_t \). The subset \( A_d \) and \( A_s \) contains
respectively the discrete normal and
continuous normal arcs which can be seen as a
function:
\[
A_d: ((P_D \times T_D) \cup (T_D \times P_D)) \times \mathbb{N}^{+} 
\rightarrow \mathbb{N}_{\geq 0},
\]
\[
A_s: ((P_C \times T_D) \cup (T_D \times P_C)) \times \mathbb{R}^{+} 
\rightarrow \mathbb{R}_{\geq 0}.
\]
The arcs of \( A_d \) and \( A_s \), are drawn as single
arrows. The subset of discrete inhibitory arcs
is
\[
A_h: (P_D \times T_D) \times \mathbb{N}_{\geq 0} \rightarrow \mathbb{N}_{\geq 0},
\]
or that of
continuous inhibitory arcs
\[
A_t: (P_C \times T_D) \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}.
\]
These arcs are drawn with a small circle at the end. The subset \( A_c \)
defines the continuous flow arcs \( A_c: ((P_D \times T_D) \cup (T_D \times P_C)) \times \mathbb{R}^{+} \rightarrow \mathbb{R} \), and these arcs are
drawn as double arrows to suggest a pipe.

A test input arc \( A_t \) is directed from a place of
any kind to a transition of any kind, that
\( A_t: (P_D \times T_D) \times \mathbb{N}_{\geq 0} \rightarrow \mathbb{N}_{\geq 0} \) or
\( A_t: (P_C \times T_D) \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R} \) and is drawn as dotted
single arrows. It does not consume the content
of the source place. The arc of a net is drawn if
the cardinality is not identically zero and it is
labeled next to the arc, with a default value
being 1. The \( \mathbb{N}_{\geq 0} \) and \( \mathbb{R}_{\geq 0} \) are the sets of discrete
natural and real numbers, respectively.

- \( K_p: P_D \rightarrow \mathbb{N}_{\geq 0} \) is the capacity-function
of discrete places and for each \( p_i \in P_D \) this is
represented by the minimum \( K_{p_i}^{\text{min}} \) and
maximum \( K_{p_i}^{\text{max}} \) capacity which can contain an
natural number of tokens, \( 0 \leq K_{p_i}^{\text{min}} \leq K_{p_i}^{\text{max}} < +\infty \);
- \( K_b: P_C \rightarrow \mathbb{R}_{\geq 0} \) is the capacity-function
of continuous places and for each \( p_i \in P_C \) it
describes the fluid lower bounds \( x_i^{\text{min}} \) and
upper bounds \( x_i^{\text{max}} \) of the fluid, so that
\(-\infty < x_i^{\text{min}} < x_i^{\text{max}} < +\infty \). By default,
the \( x_i^{\text{min}} = 0 \), \( x_i^{\text{max}} = +\infty \);
- \( G: T \times Bag(P) \rightarrow \{ \text{true}, \text{false} \} \) is the guard
function defined for each transition. For \( t \in T \)
a guard function \( g(t, M) \) will be evaluated in each
marking \( M \), and if it evaluates to \( \text{true} \), the
transition may be enabled, otherwise \( t \) is
disabled (by default is \( \text{true} \));
- \( Pri: T_D \rightarrow \mathbb{N}_{\geq 0} \) defines the priority functions
for the firing of each transition. By default it is 0.
The enabling of a transition with higher priority
disables all the lower priority transitions;
- \( l: T \cup L \rightarrow \mathbb{N}_{\geq 0} \) is a labeling function that
assigns a label to nodes (transitions and places)
of a net in such a way that it maps the node
name of the net into an action name or a
condition name.
The structure of a GDPN is static. The dynamics of a net structure is specified by defining its initial marking and its marking evolution rule.

**Definition 2:** A timed marked labeled GDPN is a pair $NH = < N, M_0 >$, where $N = < H, \theta, W, V >$ is a labeled GDPN structure (see Definition 1) with the respective attributes of timed transitions and $M_0$ is the initial marking of the net so that:

- The current marking (state) value of a net depends on the kind of place, and it is described by a pair of vector-columns $M = (m, x)$, where $m: P_D \rightarrow \mathbb{N}$ and $x: P_C \rightarrow \mathbb{R}$ are the marking functions of respective type of places. $m_i \geq 0, \forall p_i \in P_D$ with $m_i$ describe the number of tokens in discrete place $p_i$, and it is represented by black dots. $x_k = x(b_k)$ in continuous place $b_k$ and it is a real number, also allowed to take negative real value. The initial marking of net is $M_0 = (m_0, x_0)$. Vector-columns $m_0$ and $x_0$ give the initial marking of discrete places and of continuous places, respectively;

- The set of discrete transitions $T_D$ is partitioned into $T_D = T_0 \cup T_\theta, T_0 \cap T_\theta = \emptyset$ so that: $T_\theta$ is a set of timed discrete transitions and $T_0$ is a set of immediate discrete transitions, so that $\text{Pri}(T_0) > \text{Pri}(T_\theta)$. A timed discrete transition $t \in T_\theta$ has a firing delay associated to it: $\theta: T_\theta \times \text{Bag}(P) \rightarrow \mathbb{R}$, which can be marking dependent. The $\mathbb{R}$ is the set of nonnegative real numbers. Timed discrete $t$ is drawn as a black rectangle.

Let $T(M)$ denote the set of enabled transitions in current marking $M = (m, x)$. Thus, a timed transition $t \in T_\theta(M)$ is enabled in current tangible marking $M$ and it fires after a delay $\theta(t, M)$;

- $W: T_0 \times \text{Bag}(P) \rightarrow \mathbb{R}$ is the weight function of immediate discrete transitions $t_j \in T_0$, and this type of transitions is drawn with a black thin bar and has a zero constant firing time. Several enabled immediate transitions are scheduled to fire at the same time in vanishing marking $M_t$;

- $V: T_C \times \text{Bag}(P) \rightarrow \mathbb{R}$ is the marking dependent fluid rate function of timed continuous transitions $T_C$. If $t_i \in T_C$ is enabled in tangible marking $M$ it fires with rate $V(M)$, so that it continuously changes the fluid level of place $b_i \in P_C$.

Figure 1 summarizes the all possible ways of placing arcs in a GDPN net for discrete transition and continuous transition with the discrete places and continuous places, respectively.

**Fig. 1.** All the possible ways of placing arcs in a timed GDPN net.

Enabling and firing of transitions is described in (Gutułeac, et al., 2006).

### 4. DESCRIPTIVE GDPN

In this section we present the model of descriptive NH net using the approach considered in (Gutuleac and Mocanu, 2005), which consists in incorporating the compositional features into GDPN. Due to the space restrictions we will only give a brief overview to this topic and refer the reader to (Gutuleac and Mocanu, 2005) and the references therein.

In following for abuse of notation, labels and name of transitions/places are the same.

A basic descriptive expression (DE) element (bDE) for a basic net (bNH) is:

$$bDE = \sum_{i_j}^{\alpha_j} \left[ m_i^0 p_i [W_i, W_i] \right]_v^{\alpha_j}.$$
The translation of this bNH is shown in figure 2a where $e_i$ is the input event (transition $t_j$ or rewriting rule $r_j$) with the action $\alpha_j$ and $e_j$ is the output event $e_i$ with the action $\alpha_i$ of the place $p_i$ with initial marking $m_i^0 = M(p_i)$. The flow relation functions $W_i^+ = \text{Pre}(t_j, p_i)$ and $W_i^- = \text{Post}(t_j, p_i)$, return the multiplicities of the input and output arcs of $p_i$. The derivative elements of $bDE$ are for $p'_i = \emptyset$, $W_i^- = 0$ is $i_{t_j}^{\alpha_j} p_i [W_i]$ with final place $p_i$ of $t_j$ and for $\ast p_i = \emptyset$, $W_i^- = 0$ is $m_i^0 [W_i]^{\alpha_k}$ with entry place $p_i$ of $t_k$ (see figure 2b).

![Fig. 2. Translation in bPN (a) of bDE and its derivatives (b).](image)

If the initial marking $m_i^0$ of place $p_i$ is zero tokens (or fluid level) we can omit $m_i^0 = 0$ in $bDE$. By default, if the type of action $\alpha$ is not mentioned this means it matches the name of a transition $t$. From a $bDE$ we can build more complex $DE$ of NH components by using composition operations.

The $DE$ of $NH$ is either a $bDE$ or a composition of $DE$, i.e. $DE := bDE \mid DE \star DE \mid \circ DE$, where $\star$ represents any binary and $\circ$ any unary operation.

Some of them are the following. The unary inhibition “$\lnot$” ($\overline{p_i}$) or test “$\sim$” ($\overline{\overline{p_i}}$) operation describes the inhibitory or test arc, respectively. The $DE1 = m_i^0 \overline{p_i} [W_i]^{\alpha_j}_{t_j}$ describe the inhibitor arc in $NH$ with a weight function: $W_i = \text{Inh}(p_i, t_j)$.

The binary sequential operation “$|$” determines the logics of an interaction between two local states: pre-condition and post-condition, by the action (event). The specified conditions are always fulfilled.

The sequential operation is an associative, reflexive and transitive property, but is not commutative. The descriptive expression

$$DE \ 2 = m_i^0 p_i [W_i]^{\alpha_j}_{t_j} m_k^0 p_k [W_k]$$

means that the specified conditions (local state) associated with place-symbol $p_i$ are always fulfilled and happen before the occurrence of the conditions associated with place-symbol $p_k$ by means of the action $t_j$.

The binary synchronization operation is represented by the “$*$” or “$\wedge$” operator which describes the rendez-vous synchronization (by the transition $t_j$) of two or more conditions represented respectively by symbol-place $p_i \in \ast t_j$, $i = 1, n$, i.e. it indicates that all preceeding conditions of occurrence actions must have been completed.

The binary split operation represented by the “$\circ$” split operator describes and 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.

Thus the $NH$ subnet shown in figure 1a and figure 1b is described by the $DE$’s $A$ and $B$, respectively:

$A = A1 | (p_i \circ b_3) ; B = B1 | (6.35b_2 \circ b_3)$ with $A1 = (2.4b_1 \cdot 1p_i \cdot 2p_2 \cdot 3\overline{p_3} \cdot 1.5\overline{b_2} \cdot 5\overline{b_4} \cdot 4.5\overline{b_5})_{t_i}$

and $B1 = (2.4b_2 \cdot 1\overline{p_1} \cdot \overline{p_2} \cdot 3.6b_3 \cdot \overline{b_4} \cdot 4.5\overline{b_5})_{t_i}$

The compositional binary competing parallelism operation “$\lor$” means that it can be applied over two $NH$ nets: $NH_A$ with $DE_A = A$, and $NH_B$ with $DE_B = B$. The resulting net $NH_R$ with $DE_R = R$ can be represented by $DE_R = R = A \lor B$. The fused nodes, with the same names, will inherit the incidence arcs of the marking-controlled nodes $A$ and $B$.

The precedence relations between the operations in the $DE$ is following: 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 “$\circ$”, in turn, these are superior to “$\lor$” operation. Further details on enabling and firing rules, and evolution for discrete part of $NH$ can be found in (Gütlec
and Mocanu, 2005) as they require a great deal of space.

5. DYNAMIC REWRITING GDPN

In this section we introduce the model of descriptive dynamic net rewriting systems. Let $X \rho Y$ be a binary relation. The domain of $\rho$ is the Dom($\rho$) = $\rho Y$ and the codomain of $\rho$ is the Cod($\rho$) = $X \rho$. Also, let $A = \{\text{Pre, Post, Test, Inh}\}$ be a set of arcs belonging to net $H = < P, T, \text{Pre, Post, Test, Inh, Kp, Kb, G, Pri, l}>$ (see Definition 1).

Definition 3: A dynamic rewriting GDPN is a system $RH = < N, R, \phi, G_r, G_s, M >$, where:

- $N = < H\Gamma$, $\theta, W, V >$ and $R = \{r_1, ..., r_l\}$ is a finite set of discrete rewriting rules (DR) about the run-time structural modification of a net, so that $P \cap T \cap R = \emptyset$. In the graphical representation, the DR rule is drawn as two embedded empty rectangles;
- $\phi: E \rightarrow \{T_D, R\}$ is a function which indicates for every rewriting rule the type of event that can occur and $E = T_D \cup R$ denote the set of events of the net;
- $G_r: R \times Bag(P) \rightarrow \{\text{true, false}\}$ is the transition rule guard function associated with $r \in R$, and $G_s: R \times Bag(P) \rightarrow \{\text{true, false}\}$ is the rewriting rule guard function defined for each rule of $r \in R$, respectively. For $\forall r \in R$, the function $g_r(M) \in G_r$ and $g_s(M) \in G_s$ will be evaluated in each marking and if they are evaluated to true, the rewriting rule $r$ may be enabled, otherwise it is disabled.

The default value of $g_r(M) \in G_r$ is true and for $g_s(M) \in G_s$ is false, in current marking $M$.

Let $RN = < N, R, \phi, G_r, G_s, M >$ and the $RN = < N, R, \phi, G_r, G_s, M >$ be represented by the descriptive expression $DE_{RN}$ and $DE_{RN}$, respectively. A dynamic rewriting structure modifying rule $r \in R$ of $RN$ is a map $r: DE_L \triangleright DE_W$, where the codomain of the $\triangleright$ rewriting operator is a fixed descriptive expression $DE_L$ of a subnet $RN_L$ of current net $RN$, where the domain of the $\triangleright$ is a descriptive expression $DE_W$ of a new $RN_W$ subnet. The rewriting operator $\triangleright$ represents the binary operation which produces a structure change in the $DE_{RN}$ and the net $RN$ by replacing (rewriting) the fixed current $DE_L$ of the subnet $RN_L$ ($DE_L$ and $RN_L$ are dissolved) with the new $DE_W$ of the subnet $RN_W$, now belonging to the new modified resulting $DE_{RN'}$ of the net $RN' = (RN \setminus RN_L) \cup RN_W$ where the meaning of $\setminus$ and $\cup$ is operation of removing (adding) $RN_L$ from $(RN, M)$ to the net $RN$. In this new net $RN'$, obtained by firing of enabled rewriting rule $r \in R$, the places and events with the same attributes which belong to $RN'$ are fused.

A state configuration of a net $RN$ is a pair $\Gamma(s)$, where $\Gamma$ is the current structure of net together with a current state $s = (M, \beta(M))$.

The (DR, s) is called the initial state configuration of a net $RN$.

Enabling and Firing of Events. The enabling of events depends on the marking of all places. We say that a transition $t_j \in T_D$ of the event $e_j$ is enabled in current marking $M$ if the enabling condition $ec_d(e_j, M)$ and is verified. This is described in (Guțuleac, et al., 2006)

The discrete rewriting rule $r_j \in R$, that changes the structure of $RN$, is enabled in current marking $M$ if the $ec_a(e_j, M)$ and the $g_v(r_j, M)$ are verified.

Let $T_D(M)$ and $R(M), T_D(M) \cap R(M) = \emptyset$, be the sets of enabled discrete transitions and enabled rewriting rule in current marking $M$, respectively. We denote the set of enabled events in a current marking $M$ by the $E(M) = T_D(M) \cup R(M)$.

The event $e_j \in E(M)$ fires if no other event $e_i \in E(M)$ with higher priority is enabled.

Hence, for each event $e_j$ if

if ($\phi = t_j \land (\phi = r_j \land (g_v(r_j, M) = \text{False}))$) then

the firing of transition $t_j \in T_D(M)$ or rewriting rule $r_j \in R(M)$ changes only the current marking:

(R$\Gamma$, s) $\xrightarrow{\epsilon_i}$ (R$\Gamma$, s') $\iff$ (R$\Gamma$ = R$\Gamma$ and in R$\Gamma$ the M$[e_j > M'$). Also, for event $e_j$

if ($\phi = r_j \land (g_v(r_j, M) = \text{True}))$ then

the event $e_j$ occurs to firing of rewriting rule $r_j$ and it changes the configuration and marking of the current net in the following way:
\( (RΓ, s) \xrightarrow{e_j} (RΓ', s'), \ M[r_j > M'] \).

The accessible state graph of a \( RN = \langle RΓ, M >\) net is the labeled directed graph whose nodes are the states and whose arcs which are labeled with events or rewriting rules of \( RN \):

a) firing of an enabled event \( e_j \in E(M) \) determines an arc from the state \( (RΓ, s) \) to the state \( (RΓ, s') \) which is labeled with event \( e_j \) when this event can fire in the net configuration \( RΓ \) at marking \( M \) and leads to a new state:

\[
\begin{align*}
  s': (RΓ, s) & \xrightarrow{e_j} (RΓ', s') \iff \\
  (RΓ = RΓ' \land M[e_j > M' \in RΓ); \\
  b) change configuration: arcs from state \( (RΓ, s) \) to state \( (RΓ', s') \) labelled with the rewriting rule \( r_j \in R \), so that \( r_j : (RΓ,M_L)\triangleright (RΓ',M_R) \) which represent the change configuration of current \( RN \) net: \( (RΓ, s) \xrightarrow{r_j} (RΓ', s') \) with \( M[r_j > M' \in RΓ \).
\end{align*}
\]

As an example, let we consider the discrete part \( RN1 \) net given by the following descriptive expression:

\[
\begin{align*}
  DE_{RΓ1} &= p_1 \mid, p_3 \vee \bar{p}_1 \mid, b_1[2.75]\vee DE_{RΓ1}' \vee DE_{RΓ1}^s, \\
  DE_{RΓ1}' &= (p_2 \cdot p_2 \cdot b_1) \mid, p_3 \mid, p_4 \mid, (p_0 \cdot \bar{p}_2 \cdot p_2), \\
  DE_{RΓ1}^s &= (p_2 \cdot b_1[1.8]) \mid, b_2 \vee (\bar{p}_4 \cdot b_1) \mid, \\
  M_0 &= (5p_1, 1p_3, 125x_1), r_1:DE_{RΓ1} \triangleright DE_{RΓ2}, \\
  \beta(M_0) &= \text{diag}(0.95, 0), g(r_1, M) = (m_q = 3) & (m_r = 0).
\end{align*}
\]

The translation of the \( DE_{RΓ1} \) in \( RN1 \) is shown in Fig. 4a, and the \( DE_{RΓ2} \) in \( RN2 \) is shown in Fig. 4b.

6. TIMED MEMBRANE GDPN

Also, for rewriting rule \( r_j \) is required to identify if \( RNf \) net belongs to \( RΓ \). Upon firing, the enabled events or rewriting rule modify the current marking.

Here we present the DMH-nets for encoding of hybrid P-systems mentioned above into descriptive dynamic rewriting \( RN \). The basis for DMH-nets is a membrane structured \( RN \), that the DE of \( RN \) net structure comprises: 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 hybrid P-system \( IP \). In order to represent the compartmentisation of membranes in \( IP \) with the \( RN \) we now use the notion of located places and located transitions introduced in (Gutuleac, 2005) and locally maximally concurrent executions of co-located transitions. The mapping of \( IP \) into DMH-net is constituted of two separate steps.

First, for every membrane \([ h ]\), we associate:

(i) to each object \( \omega_{h,i} \in \omega_h \) one place \( [h \ y_{h,i}^0 \rho_{h,i}]_h \) with the initial marking \( y_{h,i}^0 \in \{ m_0(p_i), x_0(b_i) \} \) of place \( p_{h,i} \in P^h \) and to each continuous object rule \( \rho_{h,i}^c \) one continuous transition \( [h \ u_{h,i}]_h, u_{h,i} \in T^h_c \) co-located in membrane \( h \);

(ii) to each discrete object rule \( \rho_{h,i}^d \) or discrete rewriting rules \( \rho_{h,i}^{dm} \) one discrete event \( [h \ e_{h,i,j}]_h, e_{h,i,j} \in E^h \) that acts on this membrane.

Second, for every membrane \([ h ]\), we define the \( DE_h^0 \) of \( RN_h^0 \) that corresponds to the initial configuration of the P system \( IP \) as \([ h \ DE_h^0 ]_h \).

Definition 4: The DMH-nets of degree \( n \geq 0 \), is a construct \( DMH = \bigvee_{h=0}^{n} (h \ DE_h^0 )_h \), where:

- The evolving object rule \( \rho_{u,j} : [h \ [u \rightarrow v]_v ]_h \) with multisets of objects \( u, v \) which will be kept in \([h \ ]_h \) is encoded as: \([h \ [p_{h,u} \mid e_{h,j}]_h, p_{h,v} \ ]_h \);
- The antiport rule with multisets of objects \( u, v \) and \( u', v' \), that realize a synchronization with object \( c \) and the exchange of the objects

![Fig. 4. Translation of (a) \( DE_{RΓ1} \) in \( RN1 \), (b) \( DE_{RΓ2} \) in \( RN2 \).](image-url)
\[ \rho_{k',j} : \{a_{[k', v', v]_k} \to \{a_{v'}[k', u'] \}_{k'} \}, \]

is encoded as:
\[ [h_k \ (p_{h', u} \cdot \bar{p}_{h', e})]_{i_{h', k}} \cdot (p_{h', u} \delta p_{h', v}) \}_{k'} \];

- The synport rule, that moves objects from inside to outside a membrane, or vice-versa
\[ \rho_{k',k} : \{a_{[k', v']_k} \to \{a_{v'}[k', u'] \}_{k'} \}; \]

because the configuration means both a membrane structure and the associated multisets of objects, we need the discrete rewriting rules for processing of membranes’ evolution 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 discrete rewriting events in DE’s) are defined as:

- **mr0**: Change rewriting rule, that in runtime the current structure and the multisets of objects to membrane \( h \) encoded by descriptive expression \( DE_{h} \) and it marking \( M_{h} \) is changed in a new structure \( DE_{h}^* \) with new marking \( M_{h}^* \):

\[ [a_{[k', v']_k} \to \{a_{v'}[k', u'] \}_{k'}]_{i_{h'}} ; \]

- **mr1**: Dissolve rewriting rule, that the objects as \( M_{h} \) and sub-membranes of membrane \( h \) are split into two membranes, with labels \( h' \) and \( h'' \), and the new marking is \( M_{h} = M_{h'} + M_{h''} \);

- **mr2**: Move rewriting rule where a membrane \( h' \) can be moved out or moved into a membrane \( h'' \) as a whole is done by a:

\[ [a_{[k', v']_k} \to \{a_{v'}[k', u'] \}_{k'}]_{i_{h'}} \to [a_{[k', v']_k} \to \{a_{v'}[k', u'] \}_{k'}]_{i_{h''}} \]

with their markings, respectively.

Thus, using the DMH-nets facilitates a compact and flexible specification verification and performance evaluation of parallel and distributed computing models of hybrid systems. In order to describe the details of this approach, we present a simple but illustrative example of encoding \( \Gamma_1 \) into DMH-net.

Consider the P system \( \Gamma_1 \) of degree 3 with the dissolving rule \( \delta : \mu = \{b, [a, [2, 1], 1]_0, O_d = \{a, b, c, d\}, O_e = \{a'_0, a'_1, a'_2\}_0, \omega_0 = \{b, 15.8a_2\}, \omega_2 = \emptyset \}

\[ \omega_1 = \{a, 10.5a_1\}, \rho_{0,2} = \{\rho_{0,2}^c : c \rightarrow dd_{w_2}\}, \rho_{0,2}^d = \{\rho_{0,2}^d \cdot a \rightarrow a_{w_2}^1, \rho_{0,2}^e = \{\rho_{0,2}^e \cdot a b \rightarrow a_{w_2}^1, \rho_{0,2}^f : b a \rightarrow b_{w_2}^1, \rho_{0,2}^g : b \rightarrow b_{w_2}^1, \rho_{0,2}^h : b \rightarrow b_{w_2}^1, \rho_{0,2}^i : b \rightarrow b_{w_2}^1, \rho_{0,2}^j : b \rightarrow b_{w_2}^1, \rho_{0,2}^k : b \rightarrow b_{w_2}^1, \rho_{0,2}^l : b \rightarrow b_{w_2}^1, \rho_{0,2}^m : b \rightarrow b_{w_2}^1, \rho_{0,2}^n : b \rightarrow b_{w_2}^1, \}

\]

The encoding solution for the initial configuration of \( \Gamma_1 \) is given by the DMI-net, where every object can be represented as a place labeled as the name of objects:

\[ l(p_{0,1}) = l(p_{1,1}) = a, l(p_{0,2}) = l(p_{1,2}) = b, \]

\[ l(p_{0,3}) = c, l(p_{1,3}) = l(p_{0,4}) = d, \]

and 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 \( \rho_{0,1} : c \rightarrow dd_{w_2} \) 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 \( \Gamma_1 \) are encoded in DMIH1-net as following:

\[ \text{DMH1} = \{0 \ DE_0 \ [1 \ DE_1 \ [2 \ DE_2] \ [1]_0\}, \]

\[ DE_0 = p_{0,1} \lor p_{0,2} \lor p_{1,2} \lor p_{2,1} \lor p_{3,2}, \]

\[ DE_1 = p_{0,1} \lor p_{0,2} \lor p_{1,2} \lor p_{2,1} \lor p_{3,2}, \]

\[ DE_2 = p_{0,1} \lor p_{0,2} \lor p_{1,2} \lor p_{2,1} \lor p_{3,2}. \]
The dissolving rule \( \delta_{r_1} = \delta \) is represented in \( DMH1\text{-net} \) by a following Dissolve rule:

\[
DE_1 = p_{1.1} |_{r_1.1} (p_{0.1} \land p_{2.1} \land p_{1.2} |_{r_1.2} p_{0.1} \lor DE')_r,
\]

\[
DE_1' = p_{1.2} |_{r_1.2} b_{1.2}[2.5] |_{w_1.2} b_{1.3},
\]

\[
DE_2 = p_{2.1} |_{r_1.1} p_{1.2}, \text{ Pr } i(t_{0.1}) > \text{ Pr } i(t_{0.2}),
\]

\[
M = (p_{0.2}, 1 p_{1.1}, 15.8 b_{0.1}, 10.5 b_{1.1}),
\]

\[
DE1 = DE_0 \lor DE_2 \lor DE_2'.
\]

The dissolving rule \( \delta_{r_1} = \delta \) is represented in \( DMH1\text{-net} \) by a following Dissolve rule:

\[
DE_1 = p_{1.1} |_{r_1.1} (p_{0.1} \land p_{2.1} \land p_{1.2} |_{r_1.2} p_{0.1} \lor DE')_r,
\]

\[
DE_1' = p_{1.2} |_{r_1.2} b_{1.2}[2.5] |_{w_1.2} b_{1.3},
\]

\[
DE_2 = p_{2.1} |_{r_1.1} p_{1.2}, \text{ Pr } i(t_{0.1}) > \text{ Pr } i(t_{0.2}),
\]

\[
M = (p_{0.2}, 1 p_{1.1}, 15.8 b_{0.1}, 10.5 b_{1.1}),
\]

\[
DE1 = DE_0 \lor DE_2 \lor DE_2'.
\]

The translation of \( \Gamma P \) into \( DMH1\text{-net} \) in figure 4a and \( DMH1\text{-net} \) is shown in figure 4b.

**Fig. 4**: Translation of \( DE1 \) into \( DMH1\text{-net} \) (a) and \( DE1' \) into \( DMH1\text{-net} \) for \( \Gamma P \) (b).

The reachability graph of \( DM1\text{-net} \) in listing form is:

\[
M_{DE1} = (1 p_{0.2}, 1 p_{1.1}) [U_i > M_{DE1}^1],
\]

\[
M_{DE1}^1 = (1 p_{0.2}, 2 p_{1.2}, 1 p_{1.3}) [U_i > M_{DE1}^2],
\]

\[
M_{DE1}^2 = (1 p_{0.2}, 2 p_{1.2}, 2 p_{1.4}) [U_i > M_{DE1}^3],
\]

\[
M_{DE1}^3 = (1 p_{0.2}, 3 p_{1.2}, 1 p_{1.4}) [U_i > M_{DE1}^4];
\]

\[
M_{DE1}^4 = (1 p_{0.2}, 4 p_{1.4}) [U_i = \{t_{0.2}, t_{1.1}\}],
\]

\[
U_2 = \{t_{0.1}, t_{1.2}\}; U_3 = U_4 = \{t_{0.3}\}.
\]

### 7. CONCLUSIONS AND FUTURE WORK

In this paper we have proposed a new approach for modeling performances of a P-system based on dynamic rewriting GDPN. In our method, components of continuous variant of P-systems are expressed through components of dynamic rewriting GDPN using descriptive expressions. For the creation of reconfigurable GDPN models we have also introduced a set of descriptive composition operations. In order to capture the localities and the behavior of reconfigurable and mobile hybrid systems, we defined the hybrid P-systems with active membranes and its formalization based on a new approach, the \( DMH\text{-nets} \) which allow for performance modeling.

Further work will be oriented towards the exploration of alternative modeling techniques based on Petri nets that have been suggested by their successful implementation in other scientific domains. Some features capturing the computational completeness of P-systems with maximal parallelism, and, in addition to the approaches proposed, stochastic techniques will be employed through Petri nets. Behavioral properties in P-systems such as terminating, liveness, and boundedness will be further included in our studies, based on these formalizations. We intend to gather our developments in the form of a high-level framework that includes proper modeling of the dynamic features of P systems (such as dissolve, divide, or move). The envisaged results may not be limited to P systems. We will also try to validate our modeling tool for both generic P-systems and dynamic GDPN, as a graphical as well as an algebraic tool.

We are currently developing a software visual simulator with a friendly interface for the verification and performance evaluation of descriptive rewriting GDPN and \( DMH\text{-nets} \).

### REFERENCES


