Contents lists available at ScienceDirect

Nonlinear Analysis: Hybrid Systems

journal homepage: www.elsevier.com/locate/nahs

Hybrid and Hybrid Adaptive Petri Nets: On the computation of a Reachability Graph

Estíbaliz Fraca [*], Jorge Júlvez, Manuel Silva

Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, María de Luna 3, E-50018 Zaragoza, Spain

ARTICLE INFO

Article history:
Received 1 April 2014
Accepted 26 November 2014

Keywords:
Petri Nets
Hybrid Petri Nets
Hybrid Adaptive Petri Nets
Reachability
Reachability Graph
Discrete event systems modelling

ABSTRACT

Petri Nets (PNs) constitute a well known family of formalisms for the modelling and analysis of Discrete Event Dynamic Systems (DEDS). As general formalisms for DEDS, PNs suffer from the state explosion problem. A way to alleviate this difficulty is to relax the original discrete model and deal with a *fully* or *partially* continuous model. In Hybrid Petri Nets (HPNs), transitions can be either *discrete* or *continuous*, but not both. In Hybrid Adaptive Petri Nets (HAPNs), each transition commutes between discrete and continuous behaviour depending on a threshold: if its load is higher than its threshold, it behaves as continuous; otherwise, it behaves as discrete. This way, transitions *adapt* their behaviour dynamically to their load. This paper proposes a method to compute the Reachability Graph (RG) of HPNs and HAPNs.

1. Introduction

The *state explosion problem* is a crucial drawback in the analysis of Discrete Event Dynamic Systems (DEDS). An interesting technique to alleviate this difficulty is to relax the original discrete model and deal with a continuous approximation. Such a relaxation aims at computationally more efficient analysis methods, but at the price of losing some fidelity. Unfortunately, the fluidization of the model may not always preserve important properties of the original discrete model. For instance, in the context of Petri Nets (PNs), the relaxation from discrete to continuous [1,2] does not preserve, in general, deadlock-freeness, liveness or reversibility [3,4].

This paper focuses on untimed and bounded Hybrid (HPNs) and Hybrid Adaptive Petri Nets (HAPNs). In HPNs, a *static* partial fluidization over the discrete PN formalism is done: some of the transitions of the PN system are *continuous*, while some others remain *discrete*. This fact makes HPNs a useful formalism for the modelling and analysis of real systems (see, for example, [1,5–8]).

In HAPNs, the partition between continuous and discrete transitions is not *static* but *dynamic*: each transition of a HAPN can behave as discrete or as continuous depending on its load, i.e., on its enabling degree. An *adaptive* transition has two different modes: *continuous* and *discrete*, continuous mode for high transition load (in this case, enabling degree higher than a given threshold) and discrete in other case. The HAPN formalism provides a common framework that includes three well known formalisms: Discrete (DPN), Continuous (CPN) and Hybrid PN (HPN). Thus, the modelling power of HAPN subsumes the modelling power of the other formalisms, and any analysis technique applicable to HAPN can also be applied to the others. It captures in a compact way the fact that the behaviour of a highly loaded system can be approximated by continuous dynamics while the behaviour of a system with low loads usually requires explicit discrete dynamics. The HAPN formalism

[*] Corresponding author.
E-mail addresses: efraca@unizar.es (E. Fraca), julvez@unizar.es (J. Júlvez), silva@unizar.es (M. Silva).

http://dx.doi.org/10.1016/j.nahs.2014.11.005
1751-570X/© 2014 Elsevier Ltd. All rights reserved.

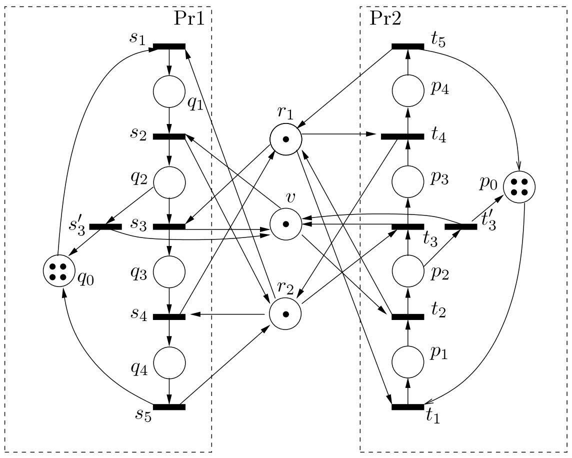


Fig. 1. A S^{3}PR system [13] that is deadlock-free as discrete, but it deadlocks as continuous ($\boldsymbol{m}_d = (3, 0, 0, 1, 0,\ 3, 0, 0, 1, 0,\ 0, 0, 1)$). It is deadlock-free as HAPN with $\boldsymbol{\mu} = \mathbf{1}$.

is intended to solve the existing trade-off between the high complexity and accuracy of DPN and the low-complexity but potential loss of accuracy associated to CPN. Moreover, it allows to preserve some properties of the discrete PN systems that are difficult to be preserved by CPN.

In this work, algorithms to compute the reachability graph of HPNs and HAPNs respectively are proposed. The algorithm for HPN can be obtained as a particular case of the one for HAPN, however they are presented in this order for easiness of the presentation.

In contrast to [9], in this work HAPNs will be generalized to the *untimed* framework, allowing full non determinism. The introduction of time in a given system would constrain the possible behaviours, producing some system trajectories that are also achievable in the untimed one. Thus, some results for properties as deadlock-freeness in the untimed framework can be almost straightforwardly applied on timed systems. In the following, it is assumed that the reader is familiar with Petri nets (see [10,11] for a gentle introduction).

Let us motivate the HAPN formalism with Example 1, in which the PN in Fig. 1 is considered, which is a System of Simple Sequential Processes (S^{3}PR) [12].

Example 1. The PN in Fig. 1 models a system in which two processes Pr_1 and Pr_2 share resources $r_1,\ r_2$ and r_3. The PN system is deadlock-free as discrete. However, when the PN system is fluidified, i.e., transitions can be fired in non negative real amounts, the system can reach a deadlock: from the initial marking $\boldsymbol{m}_0$, where $m_0[r_1] = m_0[r_2] = m_0[v] = 1$, $m_0[q_0] = m_0[p_0] = 4$ and the other places are empty, the firing sequence $\sigma = \frac{1}{2}s_1\frac{1}{2}t_1\frac{1}{2}s_2\frac{1}{2}t_2\frac{1}{2}s_3\frac{1}{2}t_3\frac{1}{2}s_1\frac{1}{2}t_1\frac{1}{2}s_2\frac{1}{2}t_2\frac{1}{2}s_3\frac{1}{2}t_3$ can be fired, reaching the deadlock marking $\boldsymbol{m}_d$, where $m_d[v] = m_d[q_3] = m_d[p_3] = 1$, $m_d[q_0] = m_d[p_0] = 3$ and the other places are empty. Therefore, the continuous PN system does not preserve the deadlock-freeness of the discrete PN system. However, this property will be preserved by the HAPN system with appropriate thresholds.

Related work

Some works in the literature deal with timed elements of very different orders of magnitude, for example in the study of manufacturing systems, chemical reactions, biological systems, etc. In this kind of systems, there are very fast and very slow reactions, leading to *stiffness* problems [14]. Moreover, in these systems (e.g. a genetic network) there are some species with very *small* populations (e.g. a gene), or with very *large* ones (e.g. proteins). In those cases, small populations should be modelled as discrete stochastic processes [15], while large populations can be well approximated with ordinary differential equations [16]. In the field of PNs, small populations could be modelled with *discrete (Markovian) PNs* [17], while large populations could be modelled with *(timed) continuous PNs* [1,4].

An important issue in this context is the partition of the populations into *large* ones and *small* ones (and hence, the transitions in continuous and discrete). With *static* partitions, each element of the system is modelled either as continuous or as discrete, but not both, dealing to *hybrid* PN [1]. With *dynamic* partitions, the elements are split during the evolution of the system. Inspired in fuzzy logic theory, the *fuzzy multimodel* is proposed in [18], where the partition is determined by a fuzzy function, i.e., a *dynamic* partition which allows some uncertainty. Another alternative is to *adapt* dynamically the partitions with a *crisp* function which uses a *threshold* associated to the transitions. HAPNs, proposed in [9] in a timed context and studied in [19] as untimed, deal with this modelling feature. Dynamic *adaptation* to the workload has been also considered for hybrid simulation in [20,21].

The rest of the paper is organized as follows: In Section 2, HAPNs are formally defined, and discrete, continuous and hybrid PNs are derived as a particular case of HAPN. Section 3 presents an algorithm to compute the reachability graph of

HPNs, and it is compared to other method in the literature. It is used to check some properties of the net system, and it is applied to three examples. Section 4 extends the algorithm to compute the reachability graph of HAPN, how to exploit the reachability graph and how to preserve some system properties with HAPN is also discussed. The algorithm is applied to two examples. Conclusions and future work are presented in Section 5.

2. Formal definitions

HAPNs are a wide formalism which includes other known formalisms: discrete, continuous and hybrid PNs. This point of view allows us to have an integrated view about them. Furthermore, methods and properties on HAPN may also be suitable for the particular formalisms.

First, HAPNs are defined in Section 2.1. Based on this general definition, the other mentioned PN formalisms are defined in the following subsection, as particular cases.

2.1. Hybrid Adaptive Petri Nets: definitions

The formalism of HAPNs is formally introduced in this section. The structure of a HAPN has the same elements of the discrete one and a threshold vector, which associates a *threshold* μ_j to each transition $t_j \in T$.

Definition 2. A HAPN is a tuple $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{Post}, \boldsymbol{\mu}\rangle$ where:

- $P = \{p_1, p_2, \ldots, p_n\}$ and $T = \{t_1, t_2, \ldots, t_m\}$ are disjoint and finite sets of places and transitions
- **Pre** and **Post** are $|P| \times |T|$ sized, natural valued, incidence matrices
- $\boldsymbol{\mu} \in \{\mathbb{R}_{\geq 0} \cup \infty\}^{|T|}$ is the threshold vector.

Given a place (or transition) $v \in P$ (or T), its *preset*, ${}^\bullet v$, is defined as the set of its input transitions (or places), and its *postset* $v^\bullet$ as the set of its output transitions (or places). We assume all transitions have at least one input place: $\forall t \in T, |{}^\bullet t| \geq 1$.

A *marking* $\boldsymbol{m}$ of a HAPN is defined as a $|P|$ sized, non negative, real valued vector: $\boldsymbol{m} \in \mathbb{R}^{|P|}_{\geq 0}$. A HAPN system is defined as follows:

Definition 3. A HAPN system [19] is a tuple $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_A$, where:

- $\mathcal{N}$ is a HAPN
- $\boldsymbol{m}_0 \in \mathbb{R}^{|P|}_{\geq 0}$ is the initial marking.

The enabling degree of a transition t_j at a marking $\boldsymbol{m}$ is defined as:

$$enab(t_j, \boldsymbol{m}) = \min_{p \in {}^\bullet t_j} \left\{ \frac{m[p]}{Pre[p, t_j]} \right\}. \tag{1}$$

The threshold μ_j of a transition t_j determines the values of the enabling degree for which the transition behaves in *continuous* or in *discrete* mode:

$$mode(t_j, \boldsymbol{m}) = \begin{cases} continuous & \text{if } enab(t_j, \boldsymbol{m}) > \mu_j \\ discrete & \text{otherwise.} \end{cases} \tag{2}$$

If a transition t_j is in *continuous* mode, i.e., $enab(t_j, \boldsymbol{m}) > \mu_j$, then t_j is enabled as continuous and it can fire. On the other hand, if t_j is in *discrete* mode, i.e., $enab(t_j, \boldsymbol{m}) \leq \mu_j$, then it is enabled iff $\lfloor enab(t_j, \boldsymbol{m})\rfloor \geq 1$.

A transition t_j that is enabled can fire. The admissible firing amounts depend on its mode:

- If $mode(t_j, \boldsymbol{m})$ is *continuous*, t_j can fire in any non negative real amount $\alpha \in \mathbb{R}_{>0}$ that does not make the enabling degree cross the threshold μ_j, i.e., $0 < \alpha \leq enab(t_j, \boldsymbol{m}) - \mu_j$.
- If $mode(t_j, \boldsymbol{m})$ is *discrete*, t_j can fire as a usual discrete transition in any natural amount $\alpha \in \mathbb{N}$ such that $0 < \alpha \leq enab(t_j, \boldsymbol{m})$.

The maximal value which can reach the enabling degree of a transition t is denoted enabling bound, which can be computed as: $eb(t) = \max\{k | \boldsymbol{m} \geq k \cdot Pre[p, t], \boldsymbol{m} = \boldsymbol{m}_0 + C \cdot \boldsymbol{\sigma}, \boldsymbol{m}, \boldsymbol{\sigma} \geq \mathbf{0}\}$.

The firing of t_j from marking $\boldsymbol{m}$ in a certain amount α leads to a new marking $\boldsymbol{m}'$, and it is denoted as $\boldsymbol{m} \xrightarrow{\alpha t_j} \boldsymbol{m}'$. It holds $\boldsymbol{m}' = \boldsymbol{m} + \alpha \cdot \boldsymbol{C}[P, t_j]$, where $\boldsymbol{C} = \boldsymbol{Post} - \boldsymbol{Pre}$ is the token flow matrix (incidence matrix if $\mathcal{N}$ is self-loop free). Hence, as in discrete systems, $\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}$, the state (or fundamental) equation summarizes the way the marking evolves, where $\boldsymbol{\sigma}$ is the firing count vector of the fired sequence. Right and left natural annullers of the token flow matrix are called T- and P-semiflows, respectively. When $\exists \mathbf{y} > \mathbf{0}$ s.t. $\mathbf{y} \cdot \boldsymbol{C} = \mathbf{0}$ the net is said to be *conservative*, and when $\exists \mathbf{x} > \mathbf{0}$ s.t. $\boldsymbol{C} \cdot \mathbf{x} = \mathbf{0}$ the net is said to be *consistent*. The *support* of a vector $\mathbf{v} \geq \mathbf{0}$ is $\|\mathbf{v}\| = \{v_i | v_i > 0\}$, the set of positive elements of $\mathbf{v}$. The *reverse* net of $\mathcal{N}$ is defined as $\mathcal{N}^{-1} = \langle P, T, \boldsymbol{Post}, \boldsymbol{Pre}, \boldsymbol{\mu}\rangle$, in which places, transitions and thresholds coincide, and arcs are inverted.

A PN system $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_A$ is bounded if $\exists b \in \mathbb{R}^+$ such that $\forall \boldsymbol{m} \in \mathrm{RS}_A(\mathcal{N}, \boldsymbol{m}_0),\ \forall p \in P, \boldsymbol{m}[p] \leq b$. The systems considered in this work are assumed to be bounded. Notice that boundedness is a requirement in most real systems. Moreover, structural boundedness, which is a sufficient condition for boundedness, can be checked in polynomial time (as it is checked for discrete PN [22]).

2.2. Discrete, continuous and hybrid PN as HAPN

In general, an *adaptive* transition has both discrete and continuous behaviours, depending on its threshold and its workload. However, if very high or very low thresholds are chosen, the transition can exhibit only discrete or only continuous behaviours.

Specifically, a transition t_j whose threshold is equal to ∞ (or equal to its enabling bound), behaves always in *discrete* mode (because its enabling degree will be always lower than ∞). It will be enabled at a marking $\boldsymbol{m}$ iff $enab(t_j, \boldsymbol{m}) \geq 1$, and it can fire a natural amount α smaller or equal to its enabling degree: $0 < \alpha \leq enab(t_j, \boldsymbol{m})$.

On the other hand, the transition t_j whose associated threshold is equal to 0, behaves always in *continuous* mode. It will be enabled at a marking $\boldsymbol{m}$ when $enab(t_j, \boldsymbol{m}) > 0$, and it can fire in any real amount α smaller than or equal to its enabling degree: $0 < \alpha \leq enab(t_j, \boldsymbol{m})$.

Using these properties, the formalisms of discrete, continuous and hybrid PNs can be obtained within the formalism of HAPNs.

- *Discrete Petri Net* (DPN) systems [10,11] (denoted as $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_D$ in this work) are HAPN systems in which all the thresholds are equal to "∞", i.e., all transitions fire in discrete amounts. In the case of bounded PNs, it will be enough to have a threshold equal to $eb(t)$ for every transition t.
- *Continuous Petri Net* (CPN) systems [1,4] (denoted as $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_C$ in this work), which represent the full fluidization of DPNs, are a particular case of HAPNs in which all the thresholds are equal to 0.
- *Hybrid Petri Net* (HPN) systems [1] (denoted as $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_H$ in this work) are partially fluidified Petri nets in which the set of transitions T is partitioned into two sets, T^c and T^d, such that if $t_j \in T^d$ then t_j always behaves as discrete (as HAPN, its threshold μ_j is equal to ∞), and if $t_j \in T^c$ then t_j always behaves as continuous (as HAPN, its threshold μ_j is equal to 0).

In conclusion, $\boldsymbol{\mu}$ defines a DPN if $\boldsymbol{\mu} = \infty$; $\boldsymbol{\mu}$ defines a CPN if $\boldsymbol{\mu} = \boldsymbol{0}$; and $\boldsymbol{\mu}$ defines a HPN if $\boldsymbol{\mu} \in \{0, \infty\}^{|T|}$.

3. On the computation of a Reachability Graph for HPNs

In this section, a method for the computation of a Reachability Graph (RG) for HPNs is introduced. First, some reachability concepts on discrete, continuous and hybrid PNs are recalled. Then, the algorithm to compute the RG is presented. It is explained with an example and it is compared with another technique from the literature. The stopping condition of the algorithm is motivated with another example. It is used to obtain the RG of a HPN which models a production system, and finally the RG is exploited to check some system properties.

3.1. Basic reachability concepts on discrete, continuous and hybrid PN

The Reachability Set (RS) of a PN system is the union of all the markings which are reachable by the system from the initial one. The RS of a DPN system is a set of *disjoint* points in the $\mathbb{N}^{|P|}$ space. By contrast, the RS of a CPN system is a convex set in $\mathbb{R}^{|P|}_{\geq 0}$ [3]. Because HPN combines discrete and continuous transitions, its RS is a union of certain *disjoint* sets (due to the combination of discrete and continuous firings). These sets are defined as follows (where t_{γ_i} denotes the ith transition of the sequence σ):

Definition 4.
- $\mathrm{RS}_D(\mathcal{N}, \boldsymbol{m}_0) = \{\boldsymbol{m} | \exists \sigma, \sigma = t_{\gamma_1} \cdots t_{\gamma_k}, \text{s.t. } \boldsymbol{m}_{i-1} \xrightarrow{t_{\gamma_i}} \boldsymbol{m}_i \ \forall i \in \{1..k\} \text{ and } \boldsymbol{m}_k = \boldsymbol{m}\}$
- $\mathrm{RS}_C(\mathcal{N}, \boldsymbol{m}_0) = \{\boldsymbol{m} | \exists \sigma, \sigma = \alpha_1 t_{\gamma_1} \cdots \alpha_k t_{\gamma_k}, \text{s.t. } \boldsymbol{m}_{i-1} \xrightarrow{\alpha_i t_{\gamma_i}} \boldsymbol{m}_i, \alpha_i \in \mathbb{R}_{>0} \ \forall i \in \{1..k\}, \text{ and } \boldsymbol{m}_k = \boldsymbol{m}\}$
- $\mathrm{RS}_H(\mathcal{N}, \boldsymbol{m}_0) = \{\boldsymbol{m} | \exists \sigma, \sigma = \alpha_1 t_{\gamma_1} \cdots \alpha_k t_{\gamma_k}, \text{s.t. } \boldsymbol{m}_{i-1} \xrightarrow{\alpha_i t_{\gamma_i}} \boldsymbol{m}_i, \text{ where } \alpha_i \in \mathbb{R}_{>0} \text{ if } \mu_{\gamma_i} = 0, \text{ while } \alpha_i = 1 \text{ if } \mu_{\gamma_i} = \infty, \forall i \in \{1..k\}, \text{ and } \boldsymbol{m}_k = \boldsymbol{m}\}$.

An interesting concept which is used in the characterization of the set of markings due to continuous firings is denoted as *Fireable Set* (FS), and it is a set composed of the sets of transitions for which a continuous firing sequence exists. The FS is defined for a CPN system $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_C$ [23], and it can be defined and computed similarly for a HPN system $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_H$ if only continuous transitions are considered. $\mathrm{FS}_C(\mathcal{N}, \boldsymbol{m}_0) \subseteq 2^T$ and $\mathrm{FS}_H(\mathcal{N}, \boldsymbol{m}_0) \subseteq 2^T$, where 2^T denote the set of all possible subsets of T, are defined as follows:

Definition 5.
- $\mathrm{FS}_C(\mathcal{N}, \boldsymbol{m}_0) = \{\boldsymbol{\theta} | \exists \sigma \text{ fireable from } \boldsymbol{m}_0, \text{ such that } \boldsymbol{\theta} = \|\boldsymbol{\sigma}\|\}$
- $\mathrm{FS}_H(\mathcal{N}, \boldsymbol{m}_0) = \{\boldsymbol{\theta} | \exists \sigma \text{ fireable from } \boldsymbol{m}_0, \text{ such that } \boldsymbol{\theta} = \|\boldsymbol{\sigma}\|, \text{ and } \forall t_i \in \|\boldsymbol{\sigma}\|, \mu_i = 0\}$.

Consider the PN example in Fig. 2(a). It is a HPN in which t_1, t_2 are discrete and t_3, t_4 are continuous ($\boldsymbol{\mu} = \{\infty, \infty, 0, 0\}$). Its fireable set is $\mathrm{FS}_H(\mathcal{N}, \boldsymbol{m}_0) = \{\emptyset, \{t_4\}, \{t_4, t_3\}\}$.

A general characterization of the set of reachable markings in CPNs is presented in [23,24]:

Theorem 6 *([24]). Given $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_C$, $\boldsymbol{m} \in RS_C(\mathcal{N}, \boldsymbol{m}_0)$ iff there exists a vector $\boldsymbol{\sigma}$ s.t.*

- $\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \ \boldsymbol{m} \geq \boldsymbol{0}, \ \boldsymbol{\sigma} \geq \boldsymbol{0}$
- $\|\boldsymbol{\sigma}\| \in FS_C(\mathcal{N}, \boldsymbol{m}_0) \cap FS_C(\mathcal{N}^{-1}, \boldsymbol{m})$.

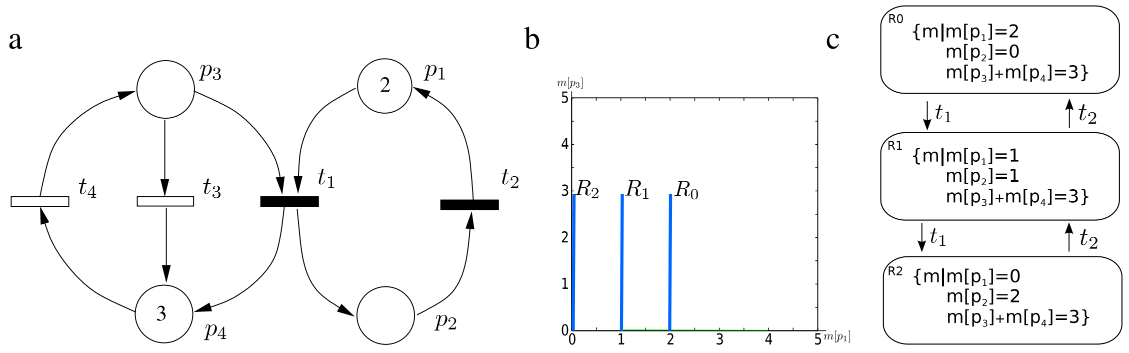

a
b
c
p_3
p_1
2
t_4
t_3
t_1
t_2
3
p_4
p_2
$m[p_3]$
R_2
R_1
R_0
$m[p_1]$
0
1
2
3
4
5
R0 {m|m[p₁]=2 m[p₂]=0 m[p₃]+m[p₄]=3}
R1 {m|m[p₁]=1 m[p₂]=1 m[p₃]+m[p₄]=3}
R2 {m|m[p₁]=0 m[p₂]=2 m[p₃]+m[p₄]=3}

Fig. 2. (a) HPN system [1]. (b) Its Reachability Set. (c) Its Reachability Graph.

Condition $\|\boldsymbol{\sigma}\| \in \mathrm{FS}_C(\mathcal{N}, \boldsymbol{m}_0)$ is needed to ensure that every transition is fireable from $\boldsymbol{m}_0$, and condition $\|\boldsymbol{\sigma}\| \in \mathrm{FS}_C(\mathcal{N}^{-1}, \boldsymbol{m})$ is needed to ensure that $\boldsymbol{m}$ is reachable with a finite sequence [24].

Finally, a RG for a HPN is defined. It is a directed graph which consists of nodes, and arcs connecting the nodes. The nodes correspond with the sets of markings obtained by the continuous firings of transitions, while the arcs represent the firing of a discrete transition in an amount equal to 1. Given a HPN system $\langle \mathcal{N}, \boldsymbol{m}_0 \rangle_H$, its $\mathrm{RG}_H(\mathcal{N}, \boldsymbol{m}_0)$ is defined as:

Definition 7. $\mathrm{RG}_H(\mathcal{N}, \boldsymbol{m}_0) = \langle \mathit{setofNodes}, \mathit{setofArcs} \rangle$ is a directed graph of nodes $\mathit{setofNodes} = \{R_1, R_2, \ldots, R_i, \ldots\}$ and directed arcs $\mathit{setofArcs} = \{A_1, A_2, \ldots, A_k, \ldots\}$:

- Each $R_i \in \mathit{setofNodes}$ is a set which belongs to the RS, i.e., $R_i \subseteq \mathrm{RS}_H(\mathcal{N}, \boldsymbol{m}_0)$. It holds that $\cup_{i=1}^{|\mathit{setofNodes}|} R_i = \mathrm{RS}_H(\mathcal{N}, \boldsymbol{m}_0)$.
- Each $A_k \in \mathit{setofArcs}$ is defined as a tuple $A_k = \langle R_i, R_j, t \rangle$:
  - $R_i, R_j \in \mathit{setofNodes}$ are the source and target nodes of the arc.
  - $t \in T$ is the discrete transition which is fired to move from R_i to R_j.
  - It holds that $\forall \boldsymbol{m}_j \in R_j, \exists \boldsymbol{m}_i \in R_i$ and a continuous sequence σ such that $\boldsymbol{m}_i \xrightarrow{1t\cdot\sigma} \boldsymbol{m}_j$, and $\forall t_z \in \|\boldsymbol{\sigma}\|, \mu_z = 0$.

$R_0 \in \mathit{setofNodes}$ is the initial node of the graph, where $\boldsymbol{m}_0 \in R_0$.

The set of markings R_i can be described with linear inequalities over $\mathbb{R}^{|P|}$, as it can be seen along the examples. In contrast with the RG of discrete PN systems, in which every firing of a transition is explicitly denoted, the RG of HPN compacts the firings of the continuous transitions inside the nodes. It can be identified as two levels of description, in which nodes are not states, but sets of markings inside which transitions can be fired. The firing of the transitions can be directly seen in the RG, but the evolution of the continuous part of the marking takes place inside the nodes and it is not explicitly seen in the RG. It has the advantage of compacting the continuous behaviour inside the nodes.

3.2. Algorithm to compute the Reachability Graph

In this section, an algorithm to calculate the RG defined in the previous section is presented. The general idea of the algorithm is to build the RG of the HPN system recursively in two steps: given a reachable marking or set of markings (initially, $\boldsymbol{m}_0$); (1) calculate the set of markings that are reachable considering only continuous firings, which constitute a node of the RG; and (2) consider the discrete firing of each discrete transition, which generates an arc of the RG from that node to a new one, which is explored in the same way.

Algorithm 1 initializes the set of nodes and the set of arcs, and calls the recursive algorithm $\mathit{explore}_H$ (Algorithm 2) with the initial marking, which is the first set to be "explored".

Algorithm 2 takes a set of markings, R, and calculates the RG from it (unless R was already considered, or a stopping condition holds, as explained after Example 9). First, the markings which are reachable considering only continuous firings (in any possible fireable amount) are calculated, which constitute a set. This is done with the function $\mathit{continuousM}_H$ (see Theorem 6):

$$\mathit{continuousM}_H(\mathcal{N}, R) = \{\boldsymbol{m} | \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \boldsymbol{m}_0 \in R, \boldsymbol{m} \geq \boldsymbol{0}, \boldsymbol{\sigma} \geq \boldsymbol{0}, \|\boldsymbol{\sigma}\| \in \mathrm{FS}_H(\mathcal{N}, \boldsymbol{m}_0) \cap \mathrm{FS}_H(\mathcal{N}^{-1}, \boldsymbol{m})\}. \quad (3)$$

Then, for every possible discrete transition t_j, i.e., $\mu_j = \infty$, the markings due to its discrete firing in an amount equal to 1 are calculated with the function $\mathit{discreteM}_H$:

$$\mathit{discreteM}_H(\mathcal{N}, R, t_j) = \{\boldsymbol{m} | \boldsymbol{m} = \boldsymbol{m}_0 + C[P, t_j], \boldsymbol{m}_0 \in R, \mathit{enab}(t_j, \boldsymbol{m}_0) \geq 1\}. \quad (4)$$

Once the markings due to the discrete firing, d, are calculated, an arc of the RG is created. This arc goes from the current node which is being explored to a new node which will be created when $\mathit{explore}_H$ will be recursively called with the new set of markings d.

Reachability set computation. The RS of a HPN system, $\mathrm{RS}_H(\mathcal{N}, \boldsymbol{m}_0)$, is the union of all the homogeneous regions included in $\mathrm{RG}_H(\mathcal{N}, \boldsymbol{m}_0)$ which have been computed with Algorithms 1 and 2.

Algorithm 1 *calculateRG$_H$*

```
Input PN ($\mathcal{N}$), initialMarking ($\boldsymbol{m}_0 \in \mathbb{R}^{|P|}_{\geq 0}$)
Output Reachability Graph ($RG$)
1: $setofNodes = \emptyset$
2: $setofArcs = \emptyset$
3: $R_0$ = explore$_H(\mathcal{N}, \{\boldsymbol{m}_0\})$
4: $RG = \langle setofNodes, setofArcs \rangle$
5: return $RG$
```

Algorithm 2 *explore$_H$*

```
Input PN ($\mathcal{N}$), setOfMarkings ($R \subseteq \mathbb{R}^{|P|}_{\geq 0}$)
Output setOfMarkings ($n$)
1: if $R \notin setofNodes \wedge$ not $stoppingCond(R, setofNodes)$ then
2:     $n = continuousM_H(\mathcal{N}, R)$
3:     $setofNodes = setofNodes \cup n$
4:     for all $t_j \in T$ s.t. $\mu_j = \infty$ do
5:         $d = discreteM_H(\mathcal{N}, n, t_j)$
6:         if $d \neq \emptyset$ then
7:             $setofArcs = setofArcs \cup \langle n, explore_H(\mathcal{N}, d), t_j \rangle$
8:         end if
9:     end for
10: end if
11: return $n$
```

Example 8. Let us first apply the algorithm to an example, and then compare the algorithm proposed here with other technique proposed in the literature for the computation of the RG of HPN. The HPN depicted in Fig. 2(a) is taken from [1], in which transitions t_1 and t_2 are *discrete* and t_3 and t_4 are *continuous* (i.e., $\boldsymbol{\mu} = (\infty, \infty, 0, 0)$) and the initial marking is $\boldsymbol{m}_0 = (2, 0, 0, 3)$.

First, *calculateRG$_H$*$(\mathcal{N}, \boldsymbol{m}_0)$ is called. Then, in the algorithm, *explore$_H$* is called with this initial marking: *explore$_H$*$(\mathcal{N}, \boldsymbol{m}_0)$. The RG obtained by the algorithm is depicted in Fig. 2(c).

The union of all the sets of the RG gives the RS (see Fig. 2(b)). Its RS is represented in the axes $m[p_1]$, $m[p_3]$, because the marking of the other places can be calculated from them: $m[p_2] = 2 - m[p_1]$ and $m[p_4] = 3 - m[p_3]$.

The RG obtained with Algorithm 1 (Fig. 2(c)) has three nodes, each of them containing a set defined in $\mathbb{R}^{|P|}_{\geq 0}$. The firings of the discrete transitions make the marking move from one set to another. Notice that, as specified by *continuousM$_H$*$(\mathcal{N}, \boldsymbol{m}_0)$, an arc labelled with discrete transition t_1 can be taken from those markings at which t_1 is enabled. This corresponds with the guard $[m[p_3] = 1]$ on the arc. Moreover, the firings of the continuous transitions are codified inside the reachable sets.

Using the technique proposed by David and Alla [1], the RG of this HPN (Fig. 4.21, p. 140) consists of 9 nodes together with the interactions of discrete and continuous transitions among those nodes. An arc labelled with a discrete transition in the graph represents that it has been fired (an amount of 1). However, the firing of continuous transitions is not measured explicitly in the arcs of the RG. In our method, the firing of discrete transitions is also explicit, but the firing of continuous transitions is compacted inside the marking sets.

The RG of bounded DPN has a finite number of nodes. An interesting question is to consider if it is also true in the RG of HPN. Example 9 shows a bounded HPN whose RG has an infinite number of nodes. In the cases in which the RG has an infinite number of steps, the algorithm has to detect this situation and stop the computation of those nodes. This stopping condition is motivated with Example 9 and discussed after the example.

Example 9 (*A Bounded HPN with Infinite Reachability Graph*)**.** Consider the HPN in Fig. 3, in which t_1, t_2 are discrete and t_3, t_4 are continuous (i.e., $\boldsymbol{\mu} = (\infty, \infty, 0, 0)$).

Let us consider the computation of its RG from $\boldsymbol{m}_0 = (1, 0, 2, 0)$. At $\boldsymbol{m}_0$, transitions t_1 and t_3 are enabled. The first node, R_0, is obtained considering every possible firing of continuous transitions (in this case, any firing of t_3) from $\boldsymbol{m}_0$. It corresponds to lines 2–3 in Algorithm 2, and it results the segment between $(1, 0, 2, 0)$ and $(1, 0, 0, 1)$, see node R_0 in Fig. 4.

From R_0, the firings of discrete transitions are considered to calculate its adjacent nodes. Consider the firing of t_1, which is the only enabled discrete transition. First, the set of markings obtained by the firing of t_1 from the markings in R_0 is calculated, d in line 5 in Algorithm 2. If t_1 is fired, $m[p_3]$ and $m[p_4]$ do not change, $m[p_1] = 0$ and $m[p_2] = 1$, hence d is a segment between $(0, 1, 2, 0)$ and $(0, 1, 0, 1)$. Once calculated d, it is recursively *explored*, and the arc from R_0 to R_1 is obtained (line 7 in Algorithm 2). In the recursive call to *explore$_H$*$(\mathcal{N}, d)$, the process is repeated: first the firings of the enabled continuous transition t_4 are considered (and the triangle in R_1 is obtained), and then the possible arcs are calculated.

The marking of p_3 and p_4 (consider $m[p_3] + m[p_4]$) is decreasing when t_3, t_4 are successively fired. Notice that transitions t_1 and t_2 have to be fired every time that the marking of p_3, p_4 is divided by 2 (for example, to move from $(1, 0, 0, 1)$ to

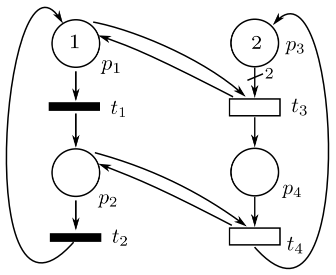


Fig. 3. HPN whose RG has an infinite number of nodes.

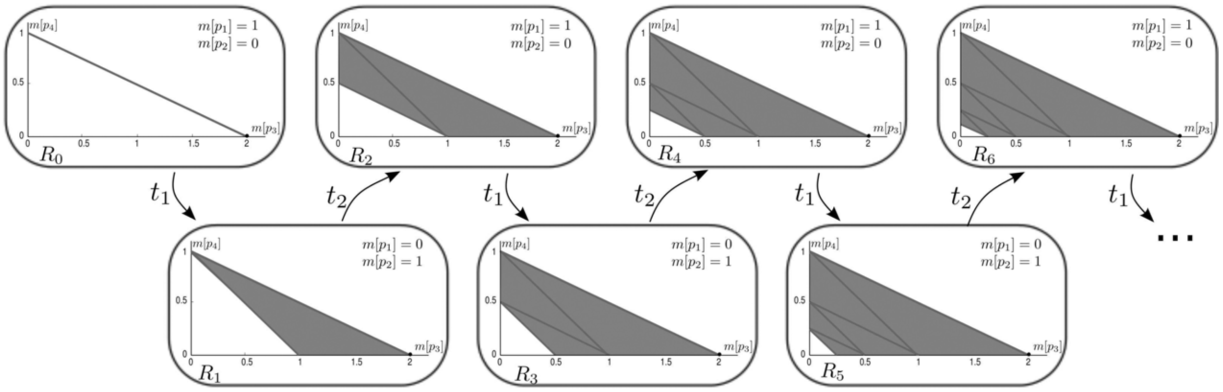


Fig. 4. RG of the PN in Fig. 3. It has an infinite number of nodes.

$(1, 0, 0, 0.5)$). It means that after firing successively t_1 and t_2 the marking would approach marking $(1, 0, 0, 0)$, but it would require an infinite number of firings of the discrete transitions, which creates an infinite number of nodes in the RG.

Stopping condition. As discussed in Example 9, the RG of a HPN system may have an infinite number of nodes. This can be due to two kinds of situations: (a) the marking of a place tends to infinity (so, the net system is unbounded), which is not possible because only bounded nets are considered in this work; or (b) the difference among the new node and a previous one tends asymptotically to 0, which would generate infinite nodes, even in bounded net systems. The occurrence of this situation is checked by the *stopping condition* when a new region R is going to be explored (line 1 in Algorithm 2). If it holds, R is not explored further to avoid an infinite execution of the procedure. In this situation, R contains the same markings than a previously created node R' except a small difference (an increment, decrement of markings, or both) which tends to 0. Sets R and R' are very similar (hence, $R \cap R' \neq \emptyset$), and the difference between them (which is $(R \cup R') \setminus (R \cap R')$) is as small as desired, measured as its n-dimensional volume. The n-dimensional volume of a set S, denoted as $\lambda(S)$, is a generalization of the concept of *area* in $\mathbb{R}^2$ or *volume* in $\mathbb{R}^3$ to a n-dimensional $S \subseteq \mathbb{R}^n$ (here, $n = |P|$). Hence, the condition to be checked is: $\exists R' \in \mathit{setofNodes}$ s.t. $R \cap R' \neq \emptyset$ and $\lambda((R \cup R') \setminus (R \cap R')) < \varepsilon$, where ε is a very small value, e.g., $\varepsilon = 10^{-3}$. This stopping condition, included in Algorithm 2 (line 1), is expressed as follows:

$$\mathit{stoppingCond}(R, \mathit{setofNodes}) = \exists R' \in \mathit{setofNodes} \text{ s.t. } (R \cap R' \neq \emptyset \wedge \lambda((R \cup R') \setminus (R \cap R')) < \varepsilon).$$

The above stopping condition guarantees the termination of the algorithm for bounded HPN systems. The computation of the RG of a HPN system which models a production system is considered in Example 10.

Example 10 (*A Production System*). The HPN in Fig. 5 is a simplified model of an example in [1], in which p_1, p_2, p_3 model one production line, and p_4, p_5, p_6 model another one. Both lines share a resource, modelled by p_8 (marked when the resource is idle), p_7 and p_9.

From the initial marking $\boldsymbol{m}_0 = (500, 0, 0, 700, 0, 0, 0, 1, 0)$, continuous transitions t_1 and t_4 are enabled; considering their possible firings, the set of markings represented in node R_0 is obtained (see Fig. 6). From R_0, discrete transitions t_7 or t_8 can be fired when their guards are satisfied. If the firing of t_7 is considered, the arc from R_0 to R_1 is created. R_1 contains the set of markings reachable after firing t_7, considering the possible firing of continuous transitions. The algorithm is recursively called from the new created nodes, and the complete RG is obtained, which is depicted in Fig. 6. In the figure, m_i represents $m[p_i]$, markings are non negative values, and markings which equal to 0 are not shown. Every arc is labelled with the discrete transition which is fired.

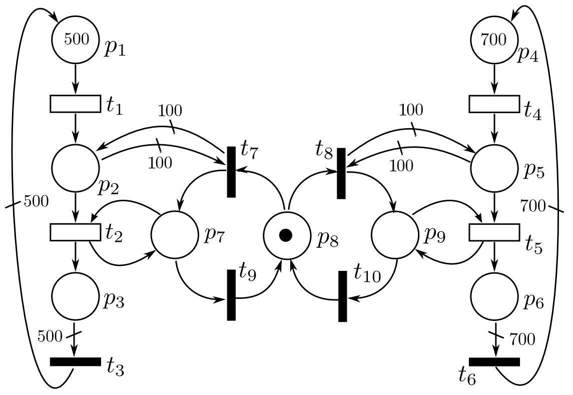
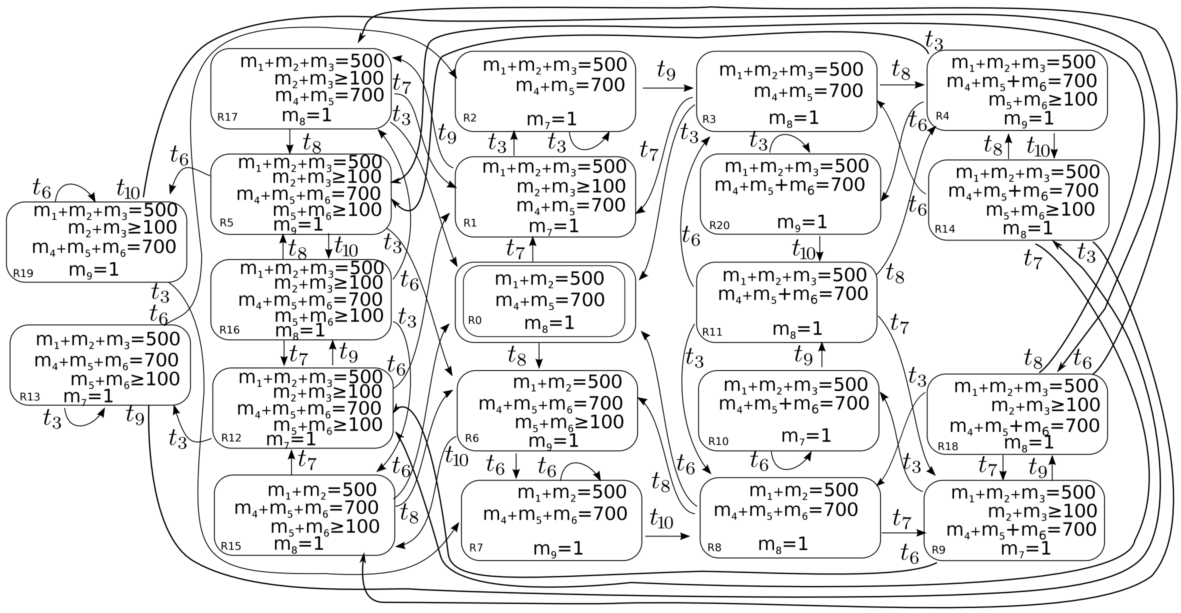


Fig. 5. HPN which models a production system in which two production lines share a resource.



R0 m₁+m₂=500 m₄+m₅=700 m₈=1
R1 m₁+m₂+m₃=500 m₂+m₃≥100 m₄+m₅=700 m₇=1
R2 m₁+m₂+m₃=500 m₄+m₅=700 m₇=1
R3 m₁+m₂+m₃=500 m₄+m₅=700 m₈=1
R4 m₁+m₂+m₃=500 m₄+m₅+m₆=700 m₅+m₆≥100 m₉=1
R5 m₁+m₂+m₃=500 m₂+m₃≥100 m₄+m₅+m₆=700 m₅+m₆≥100 m₉=1
R6 m₁+m₂=500 m₄+m₅+m₆=700 m₅+m₆≥100 m₉=1
R7 m₁+m₂=500 m₄+m₅+m₆=700 m₉=1
R8 m₁+m₂=500 m₄+m₅+m₆=700 m₈=1
R9 m₁+m₂+m₃=500 m₂+m₃≥100 m₄+m₅+m₆=700 m₇=1
R10 m₁+m₂+m₃=500 m₄+m₅+m₆=700 m₇=1
R11 m₁+m₂+m₃=500 m₄+m₅+m₆=700 m₈=1
R12 m₁+m₂+m₃=500 m₂+m₃≥100 m₄+m₅+m₆=700 m₅+m₆≥100 m₇=1
R13 m₁+m₂+m₃=500 m₄+m₅+m₆=700 m₅+m₆≥100 m₇=1
R14 m₁+m₂+m₃=500 m₄+m₅+m₆=700 m₅+m₆≥100 m₈=1
R15 m₁+m₂=500 m₄+m₅+m₆=700 m₅+m₆≥100 m₈=1
R16 m₁+m₂+m₃=500 m₂+m₃≥100 m₄+m₅+m₆=700 m₅+m₆≥100 m₈=1
R17 m₁+m₂+m₃=500 m₂+m₃≥100 m₄+m₅=700 m₈=1
R18 m₁+m₂+m₃=500 m₂+m₃≥100 m₄+m₅+m₆=700 m₈=1
R19 m₁+m₂+m₃=500 m₂+m₃≥100 m₄+m₅+m₆=700 m₉=1
R20 m₁+m₂+m₃=500 m₄+m₅+m₆=700 m₉=1
t_3 t_6 t_7 t_8 t_9 t_{10}

Fig. 6. RG of the HPN system in Fig. 5; m_i represents $m[p_i]$, zero markings are not shown.

3.3. *Exploiting the Reachability Graph*

The main goal of the algorithm is to compute the set of reachable markings. As in DPN, the set of reachable markings can be exploited to check some properties of the system. Some general considerations can be done on the obtained RG. First, it can be seen that the nodes computed by the algorithm are not *disjoint* (in contrast with the RG of DPN, in which each node corresponds to a marking). For instance, $R_0 \cap R_3 \neq \emptyset$ in Example 10 (Fig. 6). Moreover, the nodes of the RG of the HPN do not directly correspond to nodes of a subjacent discrete RG.

Because the representation of the continuous behaviour is contained inside the nodes but not explicitly seen in the arcs, the RGs from two markings $\boldsymbol{m}$ and $\boldsymbol{m}'$ mutually reachable can be different, although they describe the same *reachability sets*. For instance, the RG in Fig. 6 is obtained from $\boldsymbol{m}_0 = (500, 0, 0, 700, 0, 0, 0, 1, 0)$, but the RG obtained from $\boldsymbol{m}'_0 = (0, 0, 500, 0, 0, 700, 0, 1, 0)$ would start from an initial node $\{m_3 = 500, m_6 = 700, m_8 = 1\}$, which is not a node itself in Fig. 6 but it is included in node R_{11}. Nevertheless, both RGs describe the same reachable markings. As an unfortunate consequence, the reachability of a marking $\boldsymbol{m}'$ from a marking $\boldsymbol{m} \in \text{RS}_H(\mathcal{N}, \boldsymbol{m}_0)$ cannot be "directly" checked over $\text{RG}_H(\mathcal{N}, \boldsymbol{m}_0)$, but it should be checked over $\text{RG}_H(\mathcal{N}, \boldsymbol{m})$.

Fortunately, the computed RG can be exploited to check "directly" certain safety (nothing bad may happen) properties of $\langle \mathcal{N}, \boldsymbol{m}_0 \rangle_H$, by checking every obtained node R_i (if node R_j is a subset of R_i, checking R_j is not needed):

- The bound of a place, defined as $B(p) = \max\{m[p] | \boldsymbol{m} \in \text{RS}_H(\mathcal{N}, \boldsymbol{m}_0)\}$ can be easily computed by calculating the following linear programming problem (LPP) in each node R_i : $z_i = \max\ m[p]$ s.t. $\boldsymbol{m} \in R_i$, where $B(p) = \max\{z_i\}$. For instance, $B(p_2) = 500$ in the example.

- Mutual exclusion property can also be checked by examining each node of the RG. In the example, places p_7, p_8 and p_9 are in mutual exclusion.
- In order to check deadlock-freeness, the deadlock condition (no transition is enabled) [4] has to be checked in each node. In the example, the marking $\boldsymbol{m}_d = (0, 50, 450, 0, 50, 650, 0, 1, 0)$ is a deadlock (no transition is enabled at $\boldsymbol{m}_d$) and it is reachable in node R_{16} (even if this node has some descendent ones: R_5 (by t_8), R_{12} (by t_7), R_{15} (by t_3) and R_{17} (by t_6)).
- The deviation bound [25] of two transitions t and t', $DB(t, t')$, denotes the maximal amount that t can be fired without firing t'; and it can be checked in the RG if $t' \in T^d$. In case $t \in T^d$, it can be checked by looking at the arcs of the RG, as in discrete RGs. In case $t \in T^c$, $DB(t, t')$ can be calculated with the following LPP over each node R_z which is reachable from R_0 without considering the arcs labelled by $t' : d_z = \max\{\boldsymbol{\sigma}(t) \text{ s.t. } \boldsymbol{m} + \boldsymbol{C} \cdot \boldsymbol{\sigma} \geq 0, \boldsymbol{m} \in R_z\}$. Once d_z is computed for each node R_z, the deviation bound is obtained as $\sum d_z$. Considering the deviation bounds between two discrete transitions, B-fairness [25] can be checked.

The RG obtained with the technique proposed in [1] can also be used to check deadlock-freeness, boundedness, mutual exclusion, and deviation bound properties in an analogous way, by checking every node of the RG. However, the basic reachability property (given a marking $\boldsymbol{m}$, is it reachable in $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_H$?) requires more computation.

4. On the computation of a Reachability Graph for HAPNs

The method proposed to compute the RG which is presented in the previous section for HPNs is extended in this section for HAPNs. First, some reachability concepts from Section 3.1 are redefined in the context of HAPNs. Then, the algorithm to compute the RG is presented and illustrated with an example. Some considerations about exploiting the RG and about how to preserve deadlock-freeness property of the original discrete PN system are discussed. Finally, the technique is applied to an example.

4.1. Some reachability concepts

Let us denote $mode(t_j, \boldsymbol{m}) = D$ the discrete mode and $mode(t_j, \boldsymbol{m}) = C$ the continuous mode.

The set of reachable markings of a HAPN system $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_A$ is denoted as $\text{RS}_A(\mathcal{N}, \boldsymbol{m}_0)$:

Definition 11. $\text{RS}_A(\mathcal{N}, \boldsymbol{m}_0) = \{\boldsymbol{m} | \exists\sigma, \sigma = \alpha_1 t_{\gamma_1} \cdots \alpha_k t_{\gamma_k}, \text{ s.t. } \boldsymbol{m}_{i-1} \xrightarrow{\alpha_i t_{\gamma_i}} \boldsymbol{m}_i,\ \alpha_i \in \mathbb{R}_{>0}$ if $mode(t_{\gamma_i}, \boldsymbol{m}_{i-1}) = C$; and $\alpha_i = 1$ if $mode(t_{\gamma_i}, \boldsymbol{m}_{i-1}) = D, \forall i \in \{1..k\}$, and $\boldsymbol{m}_k = \boldsymbol{m}\}$.

The Fireable Set FS_A of HAPNs can be defined in a similar way to the FS_H of HPNs. For a HAPN, $\text{FS}_A(\mathcal{N}, \boldsymbol{m}_0)$ is the set of sets of transitions which are enabled as continuous in $\boldsymbol{m}_0$, and those which are exactly at its threshold at $\boldsymbol{m}_0$ but can become continuous by the firing of other transition in continuous mode.

Definition 12. $\text{FS}_A(\mathcal{N}, \boldsymbol{m}_0) = \{\boldsymbol{\theta} | \exists\sigma = \alpha_1 t_{\gamma_1} \cdots \alpha_k t_{\gamma_k} \text{ s.t. } \boldsymbol{m}_{i-1} \xrightarrow{\alpha_i t_{\gamma_i}} \boldsymbol{m}_i, \alpha_i \in \mathbb{R}_{>0}, \forall i \in \{1..k\}\ mode(t_{\gamma_i}, \boldsymbol{m}_{i-1}) = C$ and $enab(t_{\gamma_i}, \boldsymbol{m}_0) \geq \mu_{\gamma_i}$, and $\boldsymbol{\theta} = \|\boldsymbol{\sigma}\|\}$.

Intuitively, $\text{FS}_A(\mathcal{N}, \boldsymbol{m}_0)$ can be computed by firing transitions which are enabled as continuous in $\boldsymbol{m}_0$, or that are in the threshold and will be enabled as continuous by the continuous firing of other transitions. It can be computed with Algorithm 3.

Algorithm 3 Fireable Set$_A$

Input PN ($\mathcal{N}$), initial marking ($\boldsymbol{m}_0$)
Output Fireable Set (FS_A)

```
1: $V = \{t_j | t_j \in T, enab(t_j, \boldsymbol{m}_0) > \mu_j\}$ % transitions enabled as continuous at $\boldsymbol{m}_0$
2: $FS_A = \{v | v \subseteq V\}$ % all the subsets of V
3: while not every element of $FS_A$ has been taken do
4:     Take $f \in FS_A$ that has not been taken yet
5:     $V = \{t_j | enab(t_j, \boldsymbol{m}_0) = \mu_j \wedge (\forall p \in {}^\bullet t_j, \frac{m_0[p]}{Pre[p,t_j]} > \mu_j \vee {}^\bullet p \cap f \neq \emptyset)\}$
6:     $FS_A = FS_A \cup \{f \cup v | v \subseteq V\}$
7: end while
```

To illustrate how the set $\text{FS}_A(\mathcal{N}, \boldsymbol{m}_0)$ is computed, consider the HAPN system in Fig. 2, in which every transition is *adaptive* with threshold $\boldsymbol{\mu} = (1, 1, 1, 1)$ and initial marking $\boldsymbol{m}_0 = (2, 0, 1, 2)$. Its FS_A is, after step 2, $\text{FS}_A(\mathcal{N}, \boldsymbol{m}_0) = \{\emptyset, \{t_4\}\}$, because t_4 is the only transition enabled as continuous. At line 5 of Algorithm 3, with $f = \{t_4\}$ the set V results $\{t_1, t_3\}$, thus three new sets can be added to the set: $\text{FS}_A(\mathcal{N}, \boldsymbol{m}_0) = \{\emptyset, \{t_4\}, \{t_1, t_4\}, \{t_3, t_4\}, \{t_1, t_3, t_4\}\}$. No more transition is enabled as continuous, and the algorithm stops.

Let us now define a *homogeneous region* as a set of markings for which the enabling degree of each transition is either over (or equal to) the threshold for every marking, or below (or equal to) the threshold for every marking.

Definition 13. Let $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_A$ be a HAPN system. A set $R \subseteq \mathbb{R}^{|P|}_{\geq 0}$ is a *homogeneous region* of markings if:

- $R \subseteq \mathrm{RS}_A(\mathcal{N}, \boldsymbol{m}_0)$.
- For each $t_j \in T,\ (\forall \boldsymbol{m} \in R,\ enab(t_j, \boldsymbol{m}) \leq \mu_j) \vee (\forall \boldsymbol{m} \in R,\ enab(t_j, \boldsymbol{m}) \geq \mu_j)$.

The RG of a HAPN can be defined similarly to the RG of a HPN. The RG of HAPN systems consists of nodes, that correspond with homogeneous regions obtained by the continuous firings of transitions, and arcs connecting the nodes. In this case, the arcs are of two different types, corresponding either with the fact that a transition mode has changed (ε-arcs) or with the discrete firing of a transition (D-arcs). Given a HAPN system $\langle \mathcal{N}, \boldsymbol{m}_0\rangle_A$, we define $\mathrm{RG}_A(\mathcal{N}, \boldsymbol{m}_0)$ as a labelled directed graph with two types of arcs, whose nodes are homogeneous regions defined over $\mathrm{RS}_A(\mathcal{N}, \boldsymbol{m}_0)$:

Definition 14. $\mathrm{RG}_A(\mathcal{N}, \boldsymbol{m}_0) = \langle setofNodes, setofEarcs, setofDarcs\rangle$, where:

- Each $R_i \in setofNodes$ is a homogeneous region, $R_i \subseteq \mathrm{RS}_A(\mathcal{N}, \boldsymbol{m}_0)$.
  It holds that $\cup_{i=1}^{|setofNodes|} R_i = \mathrm{RS}_A(\mathcal{N}, \boldsymbol{m}_0)$.
- Each $A^{\varepsilon}_k \in setofEarcs$ is defined as a tuple $A^{\varepsilon}_k = \langle R_i, R_j, t_z\rangle$:
  - $R_i, R_j \in setofNodes$ are the source and target nodes of the arc.
  - $t_z \in T$ indicates the transition whose mode changes when the marking moves from R_i to R_j.
  - It holds that $\forall \boldsymbol{m}_j \in R_j, \exists \boldsymbol{m}_i \in R_i$ s.t. $enab(t_z, \boldsymbol{m}_i) = \mu_z$, a fireable sequence σ exists s.t. $\boldsymbol{m}_i \xrightarrow{\sigma} \boldsymbol{m}_j$, and σ contains only continuous firings.
- Each $A^{D}_k \in setofDarcs$ is defined as a tuple $A^{D}_k = \langle R_i, R_j, guard, t_z\rangle$:
  - $R_i, R_j \in setofNodes$ are the source and target nodes of the arc.
  - $guard \subseteq \mathbb{R}^{|P|}_{\geq 0}$ gives a set of conditions over R_i.
  - $t_z \in T$ is the transition which is fired as discrete to move from R_i to R_j.
  - It holds that $\forall \boldsymbol{m}_j \in R_j, \exists \boldsymbol{m}_i \in R_i \cap guard$, a fireable sequence σ exists s.t. $\boldsymbol{m}_i \xrightarrow{1t_z \cdot \sigma} \boldsymbol{m}_j$, and σ contains only continuous firings.

$R_0 \in setofNodes$ is the initial node of the graph, where $\boldsymbol{m}_0 \in R_0$.

4.2. Algorithm to compute the Reachability Graph

In this section, a procedure based on Algorithms 1 and 2 to compute the RG of a HAPN is proposed. Algorithm 4 initializes some global variables (*setofNodes*, *setofEarcs*, *setofDarcs*), and calls the recursive function $explore_A$ (Algorithm 5) with the initial marking of the system. As in the case of HPN, $explore_A$ is a recursive function that, given a marking or a set of markings, calculates the markings reachable from it due to continuous firings, and calculates its adjacent nodes (reached with ε-arcs or D-arcs). Function $explore_A$ will be recursively called until the full RG has been calculated.

Function $explore_A$ takes a homogeneous region R as input parameter. After checking if it has been considered before or the stopping condition holds (line 1), the markings which are reachable from R due to continuous firings (i.e., considering only firings of the transitions of the FS) are computed, with the condition that no transition traverses its threshold. This set of markings can be mathematically characterized with (5), obtained from Theorem 6 where $\mathrm{FS}_A(\mathcal{N}, \boldsymbol{m}_0)$ is calculated as specified in the previous subsection. This function is called in line 2 in Algorithm 5:

$$\begin{aligned} continuousM_A(\mathcal{N}, R) = \{&\boldsymbol{m} | \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C}\cdot\boldsymbol{\sigma},\ \boldsymbol{m}_0 \in R, \boldsymbol{m} \geq \boldsymbol{0},\ \boldsymbol{\sigma} \geq \boldsymbol{0}, \|\boldsymbol{\sigma}\| \in \mathrm{FS}_A(\mathcal{N}, \boldsymbol{m}_0) \cap \mathrm{FS}_A(\mathcal{N}^{-1}, \boldsymbol{m}), \\ &\forall t_i \in T,\ enab(t_i, \boldsymbol{m}) \geq \mu_i \Leftrightarrow enab(t_i, R) \geq \mu_i\} \end{aligned} \quad (5)$$

where $enab(t_i, R)$ is the enabling degree of t_i at any arbitrary marking $\boldsymbol{m}'$ in the homogeneous region R.

Then, the ε-arcs, those due to mode changes, are calculated (lines 4–9). This is done in three steps: First, the set of markings in which transition t_j changes its *mode* is characterized. This set is characterized with (6), which just adds an additional constraint to (5), and which is used in line 5:

$$\begin{aligned} frontierM_A(\mathcal{N}, R, t_j) = \{&\boldsymbol{m} | \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C}\cdot\boldsymbol{\sigma},\ \boldsymbol{m}_0 \in R, \boldsymbol{m} \geq \boldsymbol{0},\ \boldsymbol{\sigma} \geq \boldsymbol{0}, \|\boldsymbol{\sigma}\| \in \mathrm{FS}_A(\mathcal{N}, \boldsymbol{m}_0) \cap \mathrm{FS}_A(\mathcal{N}^{-1}, \boldsymbol{m}), \\ &enab(t_j, \boldsymbol{m}) = \mu_j, \forall t_i, enab(t_i, \boldsymbol{m}) \geq \mu_i \Leftrightarrow enab(t_i, R) \geq \mu_i\}. \end{aligned} \quad (6)$$

Algorithm 4 calculateRG$_A$

```
Input PN ($\mathcal{N}$), initial marking ($\boldsymbol{m}_0 \in \mathbb{R}^{|P|}_{\geq 0}$)
Output Reachability graph ($reachGraph$)
1: $setofNodes = \emptyset$
2: $setofDarcs = \emptyset$
3: $setofEarcs = \emptyset$
4: $R_0$ = explore$_A(\mathcal{N}, \{\boldsymbol{m}_0\})$
5: $RG_A = \langle setofNodes, setofDarcs, setofEarcs\rangle$
6: return $RG_A$
```

Second, once the *frontier* markings f are calculated, the recursive algorithm $explore_A$ is called with f. Third, an ε-arc is created from the current node to the node obtained by $explore_A$ (the second and third steps are done in line 7). This ε-arc is labelled with the transition t_j which changes its mode, which indicates that this ε-arc can only be taken from the markings of R which hold $enab(t_j) = \mu_j$.

Algorithm 5 explore$_A$

```
Input PN ($\mathcal{N}$), markingSet ($R \subseteq \mathbb{R}^{|P|}_{\geq 0}$)
Output set of markings ($n$)
1:  if $R \notin setofNodes \wedge$ not $stoppingCond(R, setofNodes)$ then
2:    $n = continuousM_A(\mathcal{N}, R)$
3:    $setofNodes = setofNodes \cup n$
4:    for all $t_j \in T$ do
5:      $f = frontierM_A(\mathcal{N}, R, t_j)$
6:      if $f \neq \emptyset$ then
7:        $setofEarcs = setofEarcs \cup \langle n, \text{explore}_A(\mathcal{N}, f), t_j \rangle$
8:      end if
9:    end for
10:   for all $t_j \in T$ do
11:     $neighbT = ({}^\bullet t_j)^\bullet \cup (t_j{}^\bullet)^\bullet$
12:     for each $T_{cn} \in 2^{neighbT}$ do
13:       $d = discreteM_A(\mathcal{N}, n, t_j, neighbT, T_{cn})$
14:       if $d \neq \emptyset$ then
15:         $setofDarcs = setofDarcs \cup \langle n, \text{explore}_A(\mathcal{N}, d), [\boldsymbol{m}_0 | \boldsymbol{m}_0 = \boldsymbol{m} - C[P, t_j], \boldsymbol{m} \in d], t_j \rangle$
16:       end if
17:     end for
18:   end for
19: end if
20: return $n$
```

Finally, the D-arcs from node R are created (lines 10–18), considering the possible discrete firing of each transitions from R. A transition t_j can only be fired as discrete if $1 \leq enab(t_j, \boldsymbol{m}_0) \leq \mu_j$. In principle, the set of markings reached by the discrete firing of t_j an amount equal to 1 is:

$$dFM = \{\boldsymbol{m} | \boldsymbol{m} = \boldsymbol{m}_0 + C[P, t_j], \boldsymbol{m}_0 \in R, 1 \leq enab(t_j, \boldsymbol{m}_0) \leq \mu_j\}.$$

The preliminary idea would be to consider dFM as the target node of the D-arc. Unfortunately, dFM might not be a homogeneous region, in fact the discrete firing of t_j can modify the mode of the transitions whose input places are in ${}^\bullet t_j$ (because its marking is decreased) or $t_j{}^\bullet$ (because its marking is increased). In the algorithm, these transitions are denoted as neighbour transitions: $neighbT = ({}^\bullet t_j)^\bullet \cup (t_j{}^\bullet)^\bullet$ (line 11).

The nodes of the RG have to be *homogeneous* regions. Hence, the algorithm has to *partitionate dFM* before adding new nodes to the RG. Notice that in each target homogeneous region R_j, for every $t_n \in neighbT$ it must hold $\forall \boldsymbol{m}, enab(t_n, \boldsymbol{m}) \geq \mu_n$ or $\forall \boldsymbol{m}, enab(t_n, \boldsymbol{m}) \leq \mu_n$.

In order to take into account every possible combination of transitions t_n *above* and *below* the threshold, we define T_{cn} as the subset of transitions whose enabling degree is forced to be higher or equal to the threshold ("$\geq$") in a given homogeneous region, $T_{cn} = \{t_{cn} \in neighbT | enab(t_{cn}, R) \geq \mu_{cn}\}$. The rest of transitions, $t_{dn} \in neighbT \setminus T_n$, are forced to have its enabling degree lower or equal to μ_{dn}.

Then, to consider all the possibilities, the algorithm iterates over $T_{cn} \in 2^{neighbT}$. Given $neighbT$ and a subset T_{cn}, the set of markings d which are reached from R with the discrete firing of t_j is obtained with (7), see line 13:

$$\begin{aligned} discreteM_A(\mathcal{N}, R, t_j, neighbT, T_{cn}) = \{&\boldsymbol{m} | \boldsymbol{m} = \boldsymbol{m}_0 + C[P, t_j], \boldsymbol{m}_0 \in R, \boldsymbol{m} \geq \boldsymbol{0}, 1 \leq enab(t_j, \boldsymbol{m}_0) \leq \mu_j, \\ &\forall t_{cn} \in T_{cn}, enab(t_{cn}, \boldsymbol{m}) \geq \mu_{cn}, \\ &\forall t_{dn} \in neighbT \setminus T_{cn}, enab(t_{dn}, \boldsymbol{m}) \leq \mu_{dn}\}. \end{aligned} \tag{7}$$

Once d is obtained, the arc from R to d is computed (line 15), where the guard $[\boldsymbol{m}_0 | \boldsymbol{m}_0 = \boldsymbol{m} - C[P, t], \boldsymbol{m} \in d]$ is specified. If the set of markings dFM has been divided in several *homogeneous regions* dealing to different nodes, then there will be several D-arcs from R with different guards and with different target nodes (see Example 15). Functions $continuousM_A$ and $frontierM_A$ force explicitly the markings to be either over (or equal to) or below (or equal to) the threshold for every transition (i.e. $\forall t_j \in T, enab(t_i, \boldsymbol{m}) \geq \mu_i \Leftrightarrow enab(t_i, R) \geq \mu_i$), and function $discreteM_A$ forces the markings to be over (or equal to) the threshold for the transitions in T_{cn} and below (or equal to) the threshold for the rest of the transitions. This guarantees that all the computed nodes are *homogeneous regions*.

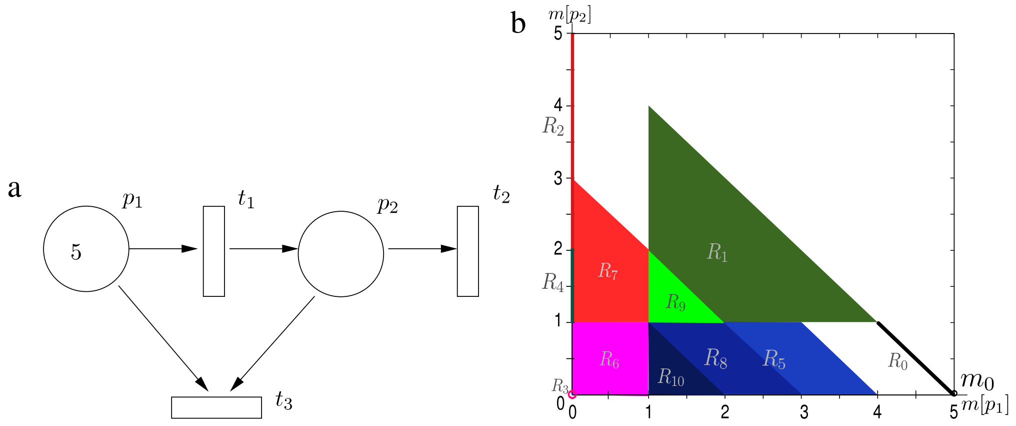


Fig. 7. (a) Example of a HAPN system. (b) Its Reachability Set, with $\boldsymbol{\mu} = (1, 1, 5)$ and $\boldsymbol{m}_0 = (5, 0)$. It is represented in the axes $m[p_1]$ and $m[p_2]$, because $m[p_3]$ is linearly dependent, more precisely $m[p_3] = 10 - 2\,m[p_1] - m[p_2]$.

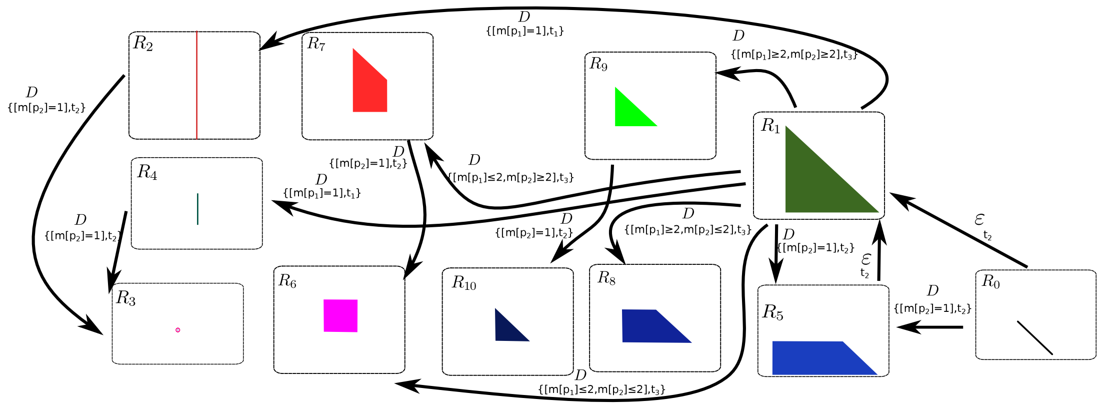


Fig. 8. RG of the HAPN system in Fig. 7(a), with $\boldsymbol{\mu} = (1, 1, 5)$ and $\boldsymbol{m}_0 = (5, 0)$. The homogeneous regions of the nodes correspond with the ones depicted in Fig. 7(b).

Example 15. Let us illustrate the algorithm with the PN in Fig. 7(a), with thresholds $\boldsymbol{\mu} = (1, 1, 5)$ and initial marking $\boldsymbol{m}_0 = (5, 0)$. Notice that with this $\boldsymbol{m}_0$, the marking of p_1 will be never higher than 5, thus for any reachable marking $\boldsymbol{m}$, $enab(t_3, \boldsymbol{m}) = \min\{m[p_1], m[p_2]\} \leq 5 = \mu_3$, and hence t_3 will be always in discrete mode.

At the initial marking $\boldsymbol{m}_0 = (5, 0)$, only transition t_1 is enabled and it is in continuous mode ($m[p_1] > \mu_1$). When t_1 is fired as continuous (and t_2, t_3 remain discrete and not enabled) all the markings contained in the segment from $(5, 0)$ to $(4, 1)$ can be reached. This segment is the starting set of the RG (see node R_0 in Fig. 8).

Let us show how ε-arcs are created (lines 4–9). When p_2 (see R_1 in Fig. 8) reaches a marking higher to 1, the enabling degree of t_2 is higher than its threshold, and t_2 becomes continuous ($enab(t_2, m) > \mu_2$). This mode change is represented in the RG as an ε-arc, in which no transition firing is considered. The transition which changes its node, t_2, is indicated in the ε-arc. This arc can only be taken from $R_1 \cap [enab(t_2) = \mu_2]$, which corresponds with the mode change.

In the example, $frontierM_A(\mathcal{N}, R_0, t_2)$ calculates the set $\{(4, 1)\}$, that is the marking from R_0 at which the mode of t_2 changes. Given this set, an ε-arc is created (see Fig. 8) labelled with t_2, from R_0 to R_1. The node R_1 (see Figs. 7(b) and 8) is created from $\boldsymbol{m} = (4, 1)$ in the same way as before: considering the continuous markings of t_1, t_2 from $(4, 1)$, i.e., by computing $continuousM_A(\mathcal{N}, \{(4, 1)\})$.

In order to illustrate the creation of the D-arcs, let us consider node R_1, in which transition t_3 is enabled as discrete for every marking $\boldsymbol{m} \in R_1$, because $1 \leq enab(t_3, \boldsymbol{m}) < 5 = \boldsymbol{m}_3$. The firing of t_3 from R_1 would reach the triangle obtained by the union of $R_6 \cup R_7 \cup R_8 \cup R_9$. However, it would not be a homogeneous region, for example $\forall \boldsymbol{m} \in R_6$, $enab(t_1, \boldsymbol{m}) \leq \mu_1$, while $\forall \boldsymbol{m} \in R_8$, $enab(t_1, \boldsymbol{m}) \geq \mu_1$. In this case, $neighbT = \{t_1, t_2, t_3\}$, but t_3 will never be in continuous mode, as explained before. Four nodes R_6, R_7, R_8, R_9 are obtained, which are reached from R_1 through D-arcs (lines 12–17 in Algorithm 5). For example, when $T_{cn} = \emptyset$, node R_6 is obtained, which is reached with the D-arc $\langle R_1, R_6, [m[p_1] \leq 2, m[p_2] \leq 2], t_3 \rangle$. It means that when transition t_3 is fired from $R_1 \cap [m[p_1] \leq 2, m[p_2] \leq 2]$, the homogeneous region R_6 is reached, at which the modes of all the transitions are discrete (forced by $T_{cn} = \emptyset$). Node R_7 (resp. R_8 and R_9) is obtained when $T_{cn} = \{t_2\}$ (resp. $\{t_1\}$ and $\{t_1, t_2\}$).

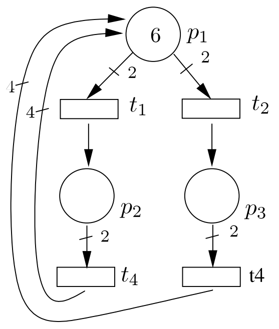


Fig. 9. Equal Conflict net system. As DPN, it is deadlock-free. As HAPN, with $\boldsymbol{\mu} = \mathbf{1}$, given the initial marking $\boldsymbol{m}_0 = (6, 0, 0)$, it can reach a deadlock $\boldsymbol{m}_d = (0, 1.5, 1.5)$ by firing the following firing sequence: $0.5t_1 0.5t_2 1t_1 1t_2$. Hence, deadlock-freeness is not preserved.

Due to the fact that HAPN includes HPN, and HPN includes DPN, the complexity of building the RG of HPN and HAPN can be at least the one of building the RG of a DPN. It is reached when the thresholds are high enough to make transitions to behave always in mode D. Consequently, in the worst case the complexity of the proposed algorithm is at least exponential in the number of places of the PN. By contrast, when every threshold is equal to 0, the resulting RG has only one node, corresponding with the RS of the CPN. The complexity of obtaining this unique node is equal to applying the function $continuousM_A$ once. Between these two situations (very high thresholds or thresholds equal to 0) a trade off between high and low complexity should be obtained. This high complexity is not only a characteristic of the proposed algorithm, but a property of the RG itself.

Stopping condition. As discussed for HPN, the RG of a HAPN system may also have an infinite number of nodes. It can be due to the same situation identified for HPN, hence the same stopping condition $stoppingCond(R, setofNodes)$ is checked in the HAPN algorithm.

Reachability set computation. The RS of a HAPN system, $RS_A(\mathcal{N}, \boldsymbol{m}_0)$, is the union of all the homogeneous regions in $RG_A(\mathcal{N}, \boldsymbol{m}_0)$ which have been computed with Algorithms 4 and 5. The RS of the HAPN in Fig. 8 is depicted in Fig. 7(b).

4.3. Exploiting the Reachability Graph and property preservation

The RG of HAPN can also be used to check some system properties, with the techniques explained in Section 3.3 for HPN: bound of a place, mutual exclusion property, deadlock-freeness or the deviation bound of two transitions.

Moreover, in the particular case of HAPN, it is interesting to consider under which conditions a property of the original discrete PN system is preserved by its hybrid adaptive counterpart.

Some results can be obtained for specific system subclasses, such as *ordinary* (i.e., nets in which arc weights are equal to 1) and *Choice Free* (i.e., $\forall p \in P, |p^\bullet| \leq 1$) (thus of *Marked Graphs*, ordinary PNs in which $\forall p \in P, |^\bullet p| = |p^\bullet| = 1$, a particular case of ordinary and Choice Free). It has been proved in [19] that deadlock-freeness property of an ordinary Choice Free PN system $\langle \mathcal{N}, \boldsymbol{m}_0 \rangle_D$ is a necessary and sufficient condition for deadlock-freeness of $\langle \mathcal{N}, \boldsymbol{m}_0 \rangle_A$ with $\boldsymbol{\mu} \in \mathbb{N}^{|P|}$.

However, this condition is not true when more general net subclasses are considered such as *Equal Conflict* (see a counterexample in Fig. 9). Then, the thresholds have to be selected *ad hoc* for each system, following different criteria.

An interesting criterion is to identify which transitions are involved in a *bad behaviour* that might happen in the CPN but not in the DPN. Then, the thresholds of the transitions related to such bad behaviour should be higher than 0, to preserve some discrete behaviour, while the rest of the transitions can be continuous (i.e., threshold equal to 0).

A *bad behaviour* can be the existence of a spurious deadlock which becomes reachable by the emptying of a trap (see [4]), such as in the PN considered in the example in Section 4.4. In that case, the undesired deadlocks occur in the limit, when the firings of the transitions can be as small as desired. This can be avoided by setting any positive threshold for the transitions, as it is illustrated in the example.

Appropriate thresholds have been selected for the examples considered in this work. However, a general method to obtain the thresholds for any net system requires more investigation.

4.4. An example. A signal transduction network modelled with HAPN

Here, an example in which HAPNs are used for the partial fluidization of DPN is presented. Selecting the appropriate threshold values, deadlock-freeness is preserved. The RG and RS of the obtained HAPN are computed, and compared with the discrete and continuous ones.

Fig. 10 presents a *signal transduction network* that can be appropriately modelled by HAPN. In such a network, three different reactions can occur (Fig. 10(a)). This behaviour is sketched in Fig. 10(b). If the places (transitions) with the same label are merged into one place (transition), the net system in Fig. 10(c) is obtained.

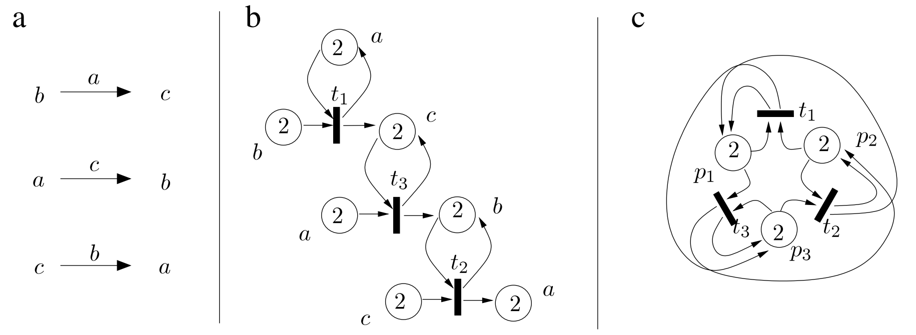


Fig. 10. (a) Signal transduction network. (b) Sketch of its behaviour. (c) PN modelling the signal transduction network.

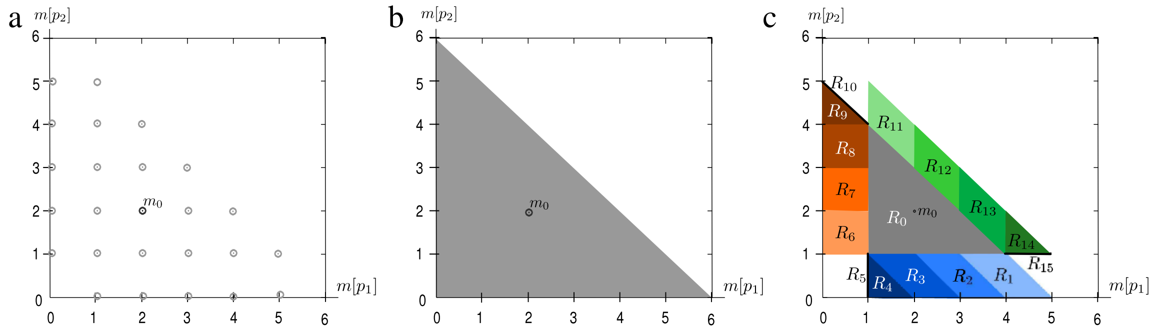


Fig. 11. RS of the PN system in Fig. 10 with $\boldsymbol{m}_0 = (2, 2, 2)$. It is represented in the axes $m[p_1]$ and $m[p_2]$, because $m[p_3]$ is linearly dependent on $m[p_1]$ and $m[p_2]$, more precisely $m[p_3] = 6 - m[p_1] - m[p_2]$. (a) Considered as DPN. (b) Considered as CPN. (c) Considered as HAPN with $\boldsymbol{\mu} = \mathbf{1}$.

This PN system is *deadlock-free* as discrete (see its RS in Fig. 11(a)). However, when considered as continuous, it reaches in the limit a marking[1] that is a deadlock (see its RS in Fig. 11(b)). Given the initial marking $\boldsymbol{m}_0 = (2, 2, 2)$; the following infinite firing sequence can be fired in the CPN: $\sigma = t_2t_3t_2t_3 1t_1 1t_2 1t_2 1t_3 0.5t_1 0.5t_2 0.5t_2 0.5t_3 0.25t_1 0.25t_2 0.25t_2 0.25t_3 \ldots$. When σ is fired, the deadlock marking $\boldsymbol{m}_d = (0, 6, 0)$ is reached in the limit (see more examples in [3]).

In this example, thresholds equal to $\boldsymbol{\mu} = (1, 1, 1)$ are considered, with the aim of avoiding the potential deadlocks in the limit. Such HAPN system behaves as continuous while the enabling degree of the places is greater than 1, and it avoids the potential deadlocks, $\boldsymbol{m}_d = (0, 6, 0)$, $\boldsymbol{m}'_d = (6, 0, 0)$ and $\boldsymbol{m}''_d = (0, 0, 6)$, as it can be seen in its RG (and in its RS).

The RS of the HAPN system with $\boldsymbol{\mu} = \mathbf{1}$ and $\boldsymbol{m}_0 = (2, 2, 2)$ is shown in Fig. 11(c). Its RG, computed with Algorithm 4, is depicted in Fig. 12. At the initial marking, the three transitions are in continuous mode, and they can fire as continuous within the homogeneous region R_0.

Consider the markings of R_0 at which $m[p_1] = 1$. From those markings, t_3 is enabled as discrete, and it can fire, emptying p_1. Exploring the resulting set of markings, the obtained homogeneous region is R_{18} (the trapezoid composed by the union of R_7, R_8 and R_9 in Fig. 11(c)). At R_{18} (indeed, at any region R_6–R_9 and R_{19}), the only transition which is enabled as continuous is t_2. When t_2 is fired, the marking of p_1 increases, and it can reach $m[p_1] = 1$ again. At the markings of R_{18} in which $m[p_1] = 1$, two things can happen: either the marking moves to node R_0 again, through the ε-arc $\langle R_{18}, R_0, t_3\rangle$ (also with the analogous arc ε-arc $\langle R_{18}, R_0, t_1\rangle$, although it is not shown in the figure for simplicity); or transition t_3 is fired as discrete, leading to R_{19} (see the D-arc $\langle R_{18}, R_{19}, [m[p_1] = 1], t_3\rangle$). Notice that the guards in the D-arcs are not shown in the figure for simplicity.

Moreover, in region R_{18} (and also in R_{19}, R_9 and R_{10}), from the markings in which $m[p_3] = 1$ (the segment from $(0, 5)$ to $(1, 4)$) transition t_2 is fireable as discrete, as indicated by the D-arc $\langle R_{18}, R_{11}, [m[p_3] = 1], t_2\rangle$. This behaviour is analogous for the three places.

Consider this PN system with a parametrized initial marking $\boldsymbol{m}_0 = (c, c, c)$ where $c \in \mathbb{N}_{>0}$. If considered as a discrete system, the number of reachable markings can be calculated as $\sum_{i=1}^{3c+1} i - 3 = \frac{(3c+1)(3c+2)}{2} - 3 = \frac{9}{2}c^2 + \frac{9}{2}c - 2$, i.e., quadratic with respect to c. If considered as a continuous system, the set of markings that can be reached with a finite or infinite firing sequence is characterized by the single convex $\{\boldsymbol{m} | \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma} \geq \mathbf{0}\}$. Unfortunately, this set contains deadlock markings that are not reachable by the original discrete system. Let us finally consider the system as HAPN with $\boldsymbol{\mu} = \mathbf{1}$. For $c = 1$, the system behaves as discrete, and the number of reachable markings is 7. For $c > 1$, the system exhibits some continuous behaviour, and the number of homogeneous regions computed by Algorithm 4 is $1 + 3\big((3c - 2) + (3c - 3)\big) = 18c - 14$, i.e., the number of reachable regions grows linearly with respect to c. Thus, the HAPN system represents an interesting

[1] Notice that it is a *spurious* marking: a solution of the state equation which is not reachable by the DPN system.

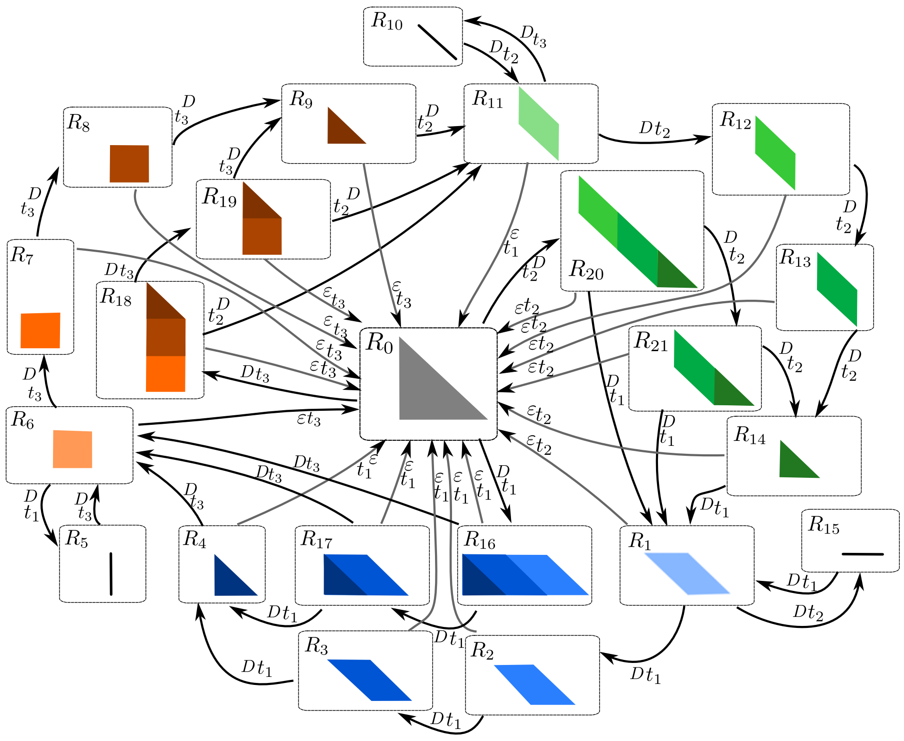


Fig. 12. RG of the PN system in Fig. 10 as HAPN, with thresholds $\boldsymbol{\mu} = \mathbf{1}$.

trade-off between the quality of the approximation (it avoids the spurious deadlocks) and the computational cost of state space exploration.

5. Conclusions

As general formalisms for DEDS, PNs suffer from the *state explosion problem*. Here, we consider hybrid (HPNs) and hybrid adaptive PNs (HAPNs), which aim to alleviate this problem by partially relaxing the firing of transitions. In HPNs, some transitions are continuous and the rest of the transitions remain discrete. In HAPNs, a threshold is defined for each transition: It behaves as continuous when its enabling degree is *higher* than its threshold; and as discrete otherwise.

HAPNs can be seen as a conceptual framework which includes discrete (DPN), continuous (CPN) and hybrid PN (HPN). Moreover, by selecting the appropriate thresholds, HAPNs offer the chance of preserving important properties of discrete event systems, as deadlock-freeness, that are not always retained by fully continuous approximations.

This paper proposes a recursive algorithm to compute a Reachability Graph (RG) for autonomous HPNs, and it is compared with an existing method. The algorithm compacts the markings which are due to continuous firings of transitions into nodes of the RG, while the firings of discrete transitions are explicitly represented with arcs connecting the nodes. Then, a more general algorithm is proposed to calculate a RG for HAPNs. The nodes of the RG for HAPNs are *homogeneous regions*, what means that for each transition, its enabling degree is either higher or equal to its threshold, or lower or equal to its threshold. The RG has two types of arcs: ε-arcs, due to mode changes; and D-arcs, due to discrete firings of transitions. Given a marking, the proposed recursive algorithm characterizes the markings reachable due to continuous firings, and the possible ε-arcs and D-arcs from it. These algorithms integrate some reachability concepts and definitions about DPNs, CPNs, and HPNs. The Reachability Set (RS) of both HPNs or HAPNs can be straightforwardly obtained from the obtained RG.

Acknowledgements

This work has been partially supported by CICYT-FEDER grant DPI2010-20413 of the Spanish Government, the European Social Fund and by the grant Ref. T27-GISED research group and the pre-doctoral grant B174/11 of the Gobierno de Aragón.

References

[1] R. David, H. Alla, Discrete, Continuous and Hybrid Petri Nets, Springer, Berlin, 2004, (2nd edition, 2010).
[2] M. Silva, L. Recalde, Petri nets and integrality relaxations: a view of continuous Petri net models, IEEE Trans. Syst. Man Cybern. 32 (4) (2002) 314–327.

[3] L. Recalde, E. Teruel, M. Silva, Autonomous continuous P/T systems, in: J.K.S. Donatelli (Ed.), Application and Theory of Petri Nets 1999, in: Lecture Notes in Computer Science, vol. 1639, Springer, 1999, pp. 107–126.
[4] M. Silva, J. Júlvez, C. Mahulea, C.R. Vázquez, On fluidization of discrete event models: observation and control of continuous Petri nets, Discrete Event Dyn. Syst., Theory Appl. 21 (4) (2011) 427–497.
[5] A. Giua, M.T. Pilloni, C. Seatzu, Modelling and simulation of a bottling plant using hybrid Petri nets, Int. J. Prod. Res. 43 (7) (2005) 1375–1395.
[6] S. Pettersson, B. Lennartson, Hybrid modelling focused on hybrid Petri nets, in: 2nd European Workshop on Real-time and Hybrid Systems, 1995, pp. 303–309.
[7] B. Gudiño Mendoza, E. López-Mellado, H. Alla, Modeling and simulation of water distribution systems using timed hybrid Petri nets, SIMULATION 88 (3) (2012) 329–347.
[8] C. Vázquez, H. Sutarto, R. Boel, M. Silva, Hybrid Petri net model of a traffic intersection in an urban network, in: Control Applications, CCA, 2010 IEEE International Conference on, September 2010, pp. 658–664.
[9] H. Yang, C. Lin, Q. Li, Hybrid simulation of biochemical systems using Hybrid Adaptive Petri Nets, in: VALUETOOLS'09: International ICST Conference on Performance Evaluation Methodologies and Tools, 2009.
[10] F. DiCesare, G. Harhalakis, J.M. Proth, M. Silva, F. Vernadat, Practice of Petri Nets in Manufacturing, Chapman & Hall, 1993.
[11] T. Murata, Petri nets: properties, analysis and applications, Proc. IEEE 77 (4) (1989) 541–580.
[12] J. Ezpeleta, J. Colom, J. Martínez, A Petri net based deadlock prevention policy for Flexible Manufacturing Systems, IEEE Trans. Robot. Automat. 11 (1995) 173–184.
[13] F. García-Vallés, Contributions to the structural and symbolic analysis of place/transition nets with applications to flexible manufacturing systems and asynchronous circuits (Ph.D. thesis), Univ. Zaragoza, 1999.
[14] K.E. Brenan, S.L. Campbell, L.R. Petzold, Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations, SIAM, Philadelphia, PA, 1996.
[15] D. Gillespie, Exact stochastic simulation of coupled chemical reactions, J. Phys. Chem. 81 (1977) 2340–2361.
[16] J.M. Harrison, Stochastic processing networks and activity analysis, in: Y. Suhov (Ed.), Analytic Methods in Applied Probability. In Memory of Fridrik Karpelevich, American Mathematical Society, Providence, RI, 2002.
[17] M.A. Marsan, G. Balbo, G. Conte, S. Donatelli, G. Franceschinis, Modelling With Generalised Stochastic Petri Nets, John Wiley and Sons, 2004.
[18] S. Hennequin, D. Lefebvre, A. El-Moudni, Fuzzy multimodel of timed Petri nets, IEEE Trans. Syst. Man Cybern. Part B 31 (2) (2001) 245–251.
[19] E. Fraca, J. Júlvez, C. Mahulea, M. Silva, On reachability and deadlock–freeness of Hybrid Adaptive Petri Nets, in: 18th IFAC World Congress, Milano, IFAC, Milano, 2011, pp. 6048–6053.
[20] A. Alfonsi, E. Cancès, G. Turinici, B.D. Ventura, W. Huisinga, Adaptive simulation of hybrid stochastic and deterministic models for biochemical systems, ESAIM Proc. 14 (2005) 1–13.
[21] M. Griffith, T. Courtney, J. Peccoud, W.H. Sanders, Dynamic partitioning for hybrid simulation of the bistable HIV-1 transactivation network, Bioinformatics 22 (22) (2006) 2782–2789.
[22] M. Silva, E. Teruel, J.M. Colom, Linear algebraic and linear programming techniques for the analysis of net systems, Lecture Notes in Comput. Sci. 1491 (1998) 309–373.
[23] J. Júlvez, L. Recalde, M. Silva, On reachability in autonomous continuous Petri net systems, in: W. van der Aalst, E. Best (Eds.), 24th Int. Conf. on Application and Theory of Petri Nets, ICATPN 2003, in: Lecture Notes in Computer Science, vol. 2679, Springer, The Netherlands, 2003, pp. 221–240.
[24] E. Fraca, S. Haddad, Complexity analysis of continuous Petri nets, in: J.-M. Colom, J. Desel (Eds.), Application and Theory of Petri Nets and Concurrency, in: Lecture Notes in Computer Science, vol. 7927, Springer, Berlin, Heidelberg, 2013, pp. 170–189.
[25] M. Silva, J.M. Colom, On the computation of structural synchronic invariants in P/T nets, in: G. Rozenberg (Ed.), Advances in Petri Nets 1988, in: Lecture Notes in Computer Science, vol. 340, Springer, 1988, pp. 387–417.