6

Modeling and Analysis of Hybrid Dynamic Systems Using Hybrid Petri Nets

Latéfa Ghomri[1] and Hassane Alla[2]
[1]University Aboubekr Belkaïd
[2]University Joseph Fourier
[1]Algeria,
[2]France

1. Introduction

Hybrid dynamic systems (HDSs) are currently attracting a lot of attention. The behavior of interest of these systems is determined by the interaction of a continuous and a discrete event dynamics. The hybrid character of a system can owe either to the system itself or to a discrete controller applied to a continuous system. Several works have been devoted to the modeling of HDSs. These topics were tackled from three different angles. The first kind of models are tools initially conceived for continuous systems that were adapted to be able to deal with switched systems. This approach consists of integrating the event aspect within a continuous formalism. Introducing commutation elements in the Bond-graph formalism is an example of this approach. The second kind of models is discrete event systems tools that were extended for HDSs modeling. In this approach, a continuous aspect is integrated in discrete event formalism. An example of such formalism is hybrid Petri nets. The last kind of formalisms are hybrid models, they combine explicitly a discrete event model and a continuous model. The most known model of this category is hybrid automata (HA). This model presents a lot of advantages. The most important is that it combines, explicitly, the basic model of continuous systems, which are differential equations, with the basic model of discrete event systems, which are finite state automata, which facilitate considerably its analysis. The existence of automatic tools for some classes of HA reachability analysis, such as HyTech[1] confer to this formalism a great analysis power. Most verification and controller synthesis techniques use HA as the investigation tool. This makes that the analysis of several hybrid systems formalisms is made after their translation in HA.

In this chapter, we consider the extension of PN formalism, initially a model for discrete event systems, so that it can be used for modeling and control of HDS. The systems studied correspond to discrete event behaviors with simple continuous dynamics. PNs were introduced, and are still used, for discrete event systems description and analysis (Murata, 1989). Currently, much effort is devoted to adapting this formalism so that it can deal with

[1] HyTech: *http://www-cad.eecs.berkeley.edu/_tah/HyTech/*

Source: Petri Net,Theory and Applications, Book edited by: Vedran Kordic, ISBN 978-3-902613-12-7, pp. 534, February 2008, I-Tech Education and Publishing, Vienna, Austria

HDSs, and many hybrid PN formalisms were conceived (Demongodin *et al* 1993; Demongodin & Koussoulas, 1998).

The first steps in this direction were taken by David & Alla (1987), by introducing the first continuous PN model. Continuous PNs can be used either to describe continuous flow systems or to provide a continuous approximation of discrete event systems behavior, in order to reduce the computing time. The marking is no longer given as a vector of integers, but as a real number vector. Thus, during a transition firing, an infinitesimal quantity of marking is taken from upstream places and put in the downstream places. This involves that transition firing is no longer an instantaneous operation but is now a continuous process characterized by a speed. This speed can be compared to a flow rate. All continuous PN models defined in the literature differ only in the manner of calculating instantaneous firing speeds of transitions.

From continuous PNs, the hybrid PN formalism was defined by David & Alla (2001), and since it is the first hybrid formalism to be defined from PNs, the authors, simply, gave it the name of hybrid PN. This formalism combines in the same model a continuous PN, which represents the continuous flow, and a discrete T-timed PN (Ramchandani, 1974), to represent the discrete behavior.

We consider in this chapter the extensions of the PN formalism in the direction of hybrid modeling. Section 2 briefly presents hybrid dynamic systems. Section 3 presents the hybrid automata model. In section 4 we discuss continuous Petri nets. These models are obtained from discrete PNs by the fluidification of the markings. They constitute the first steps in the extension of PNs toward hybrid modeling. Then, Section 5 presents two hybrid PN models, which differ in the class of HDS they can deal with. The first one is used for deterministic HDS modeling, whereas the second one can deal with HDS with nondeterministic behavior. Section 6 addresses briefly the general control structure based on hybrid PNs. Finally, Section 7 gives a conclusion and the main future research.

2. Hybrid dynamic systems

A dynamic system is especially characterized by the nature of its state variables. The latter can be of two kinds:

- Continuous state variables are variables defined on a real interval. Time, temperature, pressure, liquid level in a tank…, are examples of continuous variables.
- Discrete variables take their values in a countable set such as natural numbers or Boolean numbers. The state of a valve, the number of parts in a stock, are examples of discrete variables

Figure 1 illustrates the difference between the evolutions of a continuous and a discrete variable as a function of time.

According to the kind of state variables, we can classify the dynamic systems in three categories: continuous systems are systems which exclusively require continuous state variables for their modeling. Discrete event dynamic systems are systems whose modeling requires only discrete state variables. And finally hybrid dynamic systems which are modelled at the same time by continuous state variables and discrete state variables.

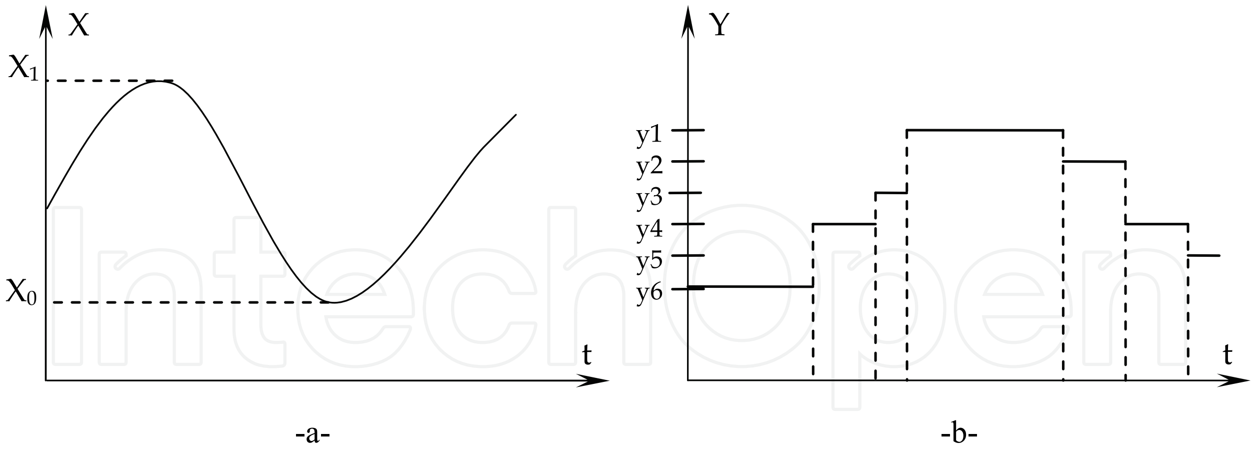


Fig. 1. –a- X is a continuous variable, it takes its values in the real interval [X_0 X_1]. –b- Y is a discrete variable which takes its values in the countable set {y_1, y_2, y_3, y_4, y_5, y_6}

2.1 Continuous dynamic systems

Chronologically, continuous dynamic systems were the first to be studied. They treat continuous values, like temperature, pressure, flow… etc. The modeling of the dynamic evolution of these systems as a function of time is represented mathematically with continuous models such as: recurrent equations, transfer function, state equations … etc, but the model which is generally used are differential equations of the form:

$$\dot{x} = f(x) \tag{1}$$

Where X is a vector representing the state of the system. The behavior of a continuous system is characterized by the solution of the differential equation $\dot{x} = f(x)$ starting from an initial state x_0.

A continuous dynamic system is said to be linear if it is modelled by a differential equation of the form.

$$\dot{x} = A.x \tag{2}$$

Where A is a constant matrix.

2.2 Discrete event dynamic systems

A discrete events system is described by discrete state variables, which take their values in a countable set. This kind of systems could be either autonomous (not timed) or timed. In the case of an autonomous discrete event system, the variable time is just symbolic, i.e. it is just used to define a chronology between the occurrences of events. In the case of a timed discrete event system, time is explicitly used to define the date of events occurrence. It can be either continuous (dense) or discrete. In the first case, to each event is attached the moment of its occurrence which takes its values in $\Re$, the set of real numbers. In the second case of timed discrete event systems time is only defined on a discrete set. The execution of a sequence of instructions on a processor belongs to this last category, since the executions may take place only with signals of the processor clock. A discrete event system can be modeled by automata, Petri nets, Markov chains, (max, +) algebra … etc.

2.3 Hybrid dynamic systems

For a long time the automatic separately treated the continuous systems and the discrete event systems. For each one of these two classes of systems exist a theory, methods and tools to solve problems which arise for them. However, the boundaries between the world of continuous systems and that of discrete event systems, are not so clear, the majority of real life systems present at the same time continuous and discrete aspects. Indeed, the majority of the physical systems cannot be classified in one of the two homogeneous categories of the dynamic systems; and state variables of interest may contain simultaneously discrete and continuous variables. In this case the systems are known as hybrid dynamic systems, they are heterogeneous systems characterized by the interaction of a discrete dynamics and a continuous dynamics. The rise of these systems is relatively new, it dates from the 1990s. Figure 2 illustrates the structure of a hybrid dynamic system.

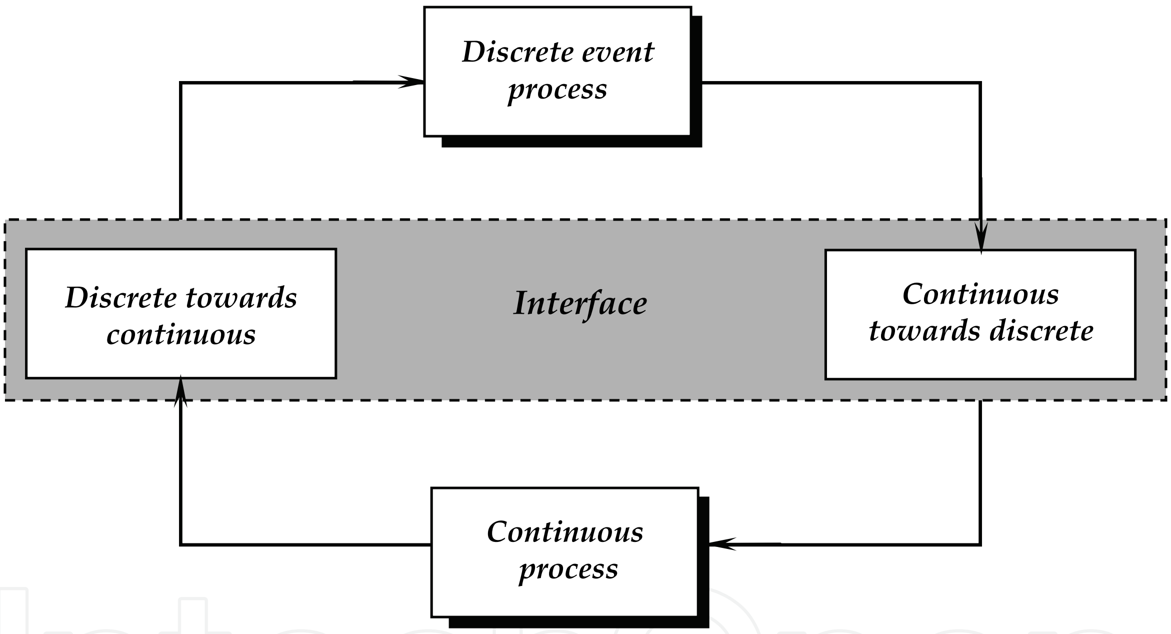


Fig. 2. Structure of a hybrid dynamic system

Research on hybrid dynamic systems is articulated around three complementary axes (Branicky *et al.* 1994; Petterson & Lennartson, 1995): Modeling relates to the formalization of precise models that can describe their rich and complex behavior. Analysis consists in developing tools for their simulation, validation and verification. Control consists in the synthesizing of a discrete (or hybrid) controller on the terms of the performance objectives.

In the sequel, we are interested in a particular class of hybrid dynamic systems; it is the class of continuous flows systems supervised by discrete events systems. This class comprises positive and linear per pieces hybrid systems. A hybrid system is said to be positive if its state variables take positive values in time. And it is said to be linear per pieces if the differential equations describing its continuous evolution are all linear. The particular interest given to the study of this class of systems has two principal reasons. First, it is sufficiently rich to allow a realistic modeling of many problems. Then, its relative simplicity allows an easy design of tools and models for its description and its analysis. Examples of this class of hybrid systems are given below.

2.4 Illustrative examples

As previously mentioned, a system is said to be hybrid if it implies continuous processes and discrete phenomena. By extension, we can state that physical systems whose certain components vary very quickly (quasi–instantaneously) compared to the others, are also hybrid. A hybrid modeling for this category of physical systems is possible and gives often good results compared to a discrete modeling . We will present two examples of hybrid systems here, the first is a system of tanks implying a (continuous) flow of liquid and the second is a manufacturing system treating a flow of products (discrete dynamics approached by a continuous description).

Example 1: Figure 3 represents a system of tanks. It comprises two tanks which are emptied permanently (except if they are empty) with a flow of 5 and 7 litres/second respectively. The tanks are also supplied in turn, with a valve whose flow is 12 litres/second. The latter has two positions, when it is in position A, it feeds tank 1 and it supplies tank 2 if it is in position B. To commutate between positions A and B the valve needs 0.5 seconds, during which, the valve behaves as if it is in its precedent position.

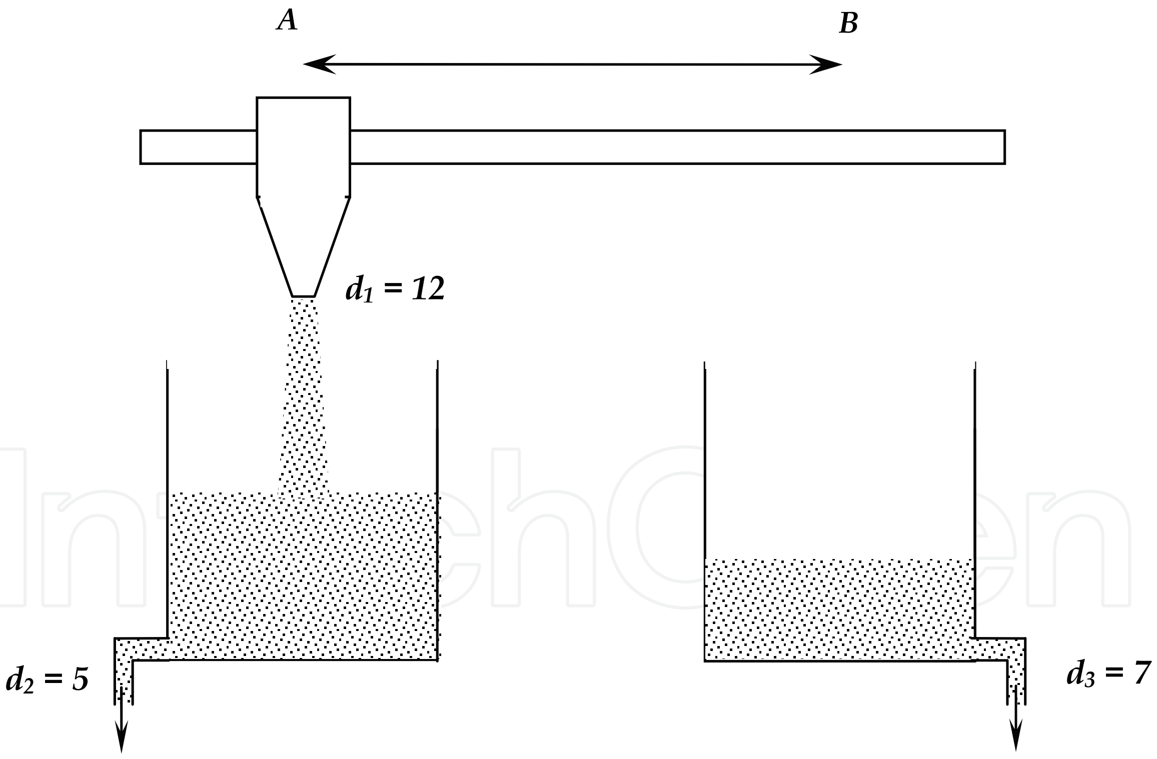


Fig. 3. System of tanks

Example 2: Figure 4 represents a manufacturing system comprising 3 machines and 2 buffers. This system is used to satisfy a periodic request, with a period of 20 time units. Machines 1 and 2 remain permanently operational, while machine 3 can be stopped for the regulation of manufacturing rate. The actions of stopping and starting machine 3 take 0.5 time units. The machines have manufacturing rates of 10, 7, and 22 parts/time units, respectively. In this system the flow of parts is supposed to be a continuous process, while the state of machine 3 as well as the state of the request is discrete variables.

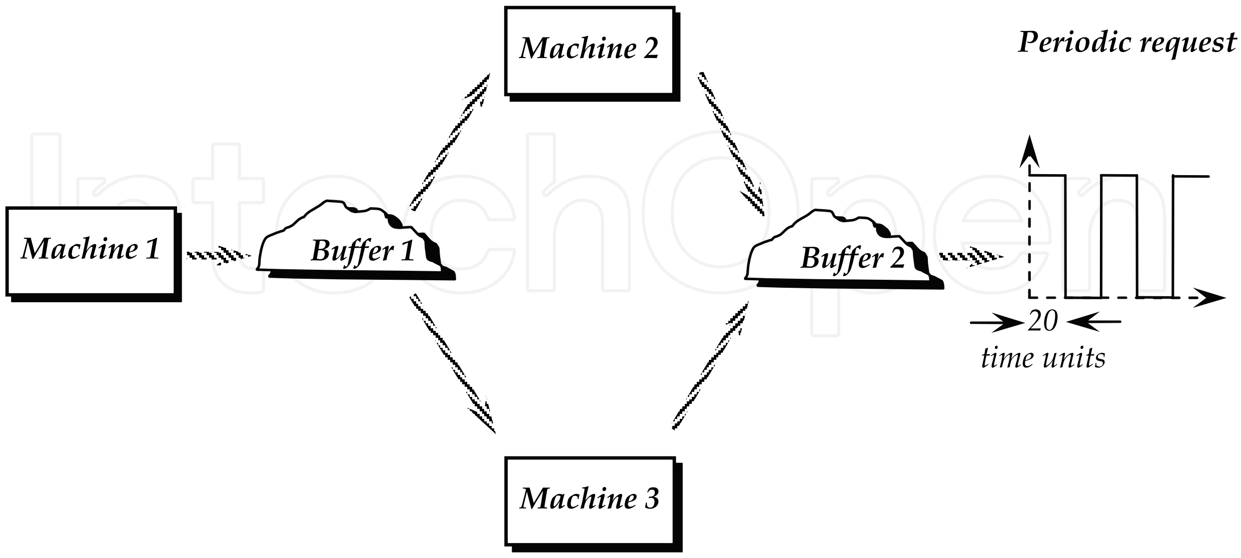


Fig. 4. Manufacturing system

3. Hybrid automata

To integrate the discrete and continuous aspects within the same model, three approaches were presented in the literature. They depend on the dominant model, i.e. the model from which the extension was carried out. We distinguish:

- The continuous approach which consists in integrating the discrete aspect within a continuous formalism. It is an extension of formalisms of continuous systems.
- The discrete approach which consists in integrating the continuous aspect within a discrete events model. The integration of the continuous aspect within the Petri nets model is an example of this approach.
- The hybrid approach which explicitly combines a continuous model and a discrete event model in the same structure. The hybrid aspect is dealt with in the interface between the two parts. An example of such formalisms is hybrid automata that we will present below.

Hybrid automata were introduced by Alur *et al*. (1995) as an extension of finite automata, which associate a continuous dynamics with each location. It is the most general model in the sense that it can model the largest continuous dynamics variety. A HA is defined as follows.

Definition 1 (Hybrid Automata): An n-dimensional HA is a structure $HA = (Q, X, L, T, F, Inv)$ such that:

1. Q is a finite set of discrete locations;
2. $X \subseteq R^n$ is the continuous state space; it is a finite set of real-valued variables; A valuation v for the variables is a function that assigns a real-value $v(x) \in R$ to each variable
   $x \in X$; V denotes the set of valuations;
3. L is a finite set of synchronization labels;

4. δ is a finite set of transitions; Each transition is a quintuple T = (q, a, μ, γ, q') such that:
    - $q \in Q$ is the source location;
    - $a \in L$ is a synchronization label associated to the transition;
    - μ is the transition guard, it is a predicate on variables values; a transition can be taken whenever its guard is satisfied;
    - γ is a reset function that is applied when taking the corresponding transition;
    - $q' \in Q$ is the target location;
5. *F* is a function that assigns to each location a continuous vector field on X; While in discrete location *q*, the evolution of the continuous variables by the differential equation

$$\dot{\mathrm{x}} = \mathrm{f}_{\mathrm{q}}(\mathrm{x}) \tag{3}$$

This equation defines the dynamics of the location *q*;

6. *Inv* is a function that affects to each location *q* a predicate *Inv* (*q*) that must be satisfied by the continuous variables in order to stay in the location *q*;

A state of a HA is a pair (*q*, *v*) consisting of a location q and a valuation *v*.

This model present a lot of advantages: It combines, explicitly, the basic model of continuous systems, which are differential equations, with the basic model of discrete event systems, which are finite state automata, this facilitate considerably its analysis; It can model the largest variety of HDSs; It has a clear graphical representation; indeed, the discrete and continuous parts are well identified; The existence of automatic tools for HA reachability analysis, such as HyTech, CMC[2], UPPAAL[3] and KRONOS[4], confer on this formalism a great analysis power. Most verification and controller synthesis techniques use HA as the investigation tool. Several problems, related to analysis of HA properties, could be expressed as a reachability problem. Note that this problem is generally undecidable unless strong restrictions are added to the basic model, to obtain special sub-classes of HA (Henzinger *et al.* 1995). The existence of computer tools allowing the analysis of the reachability problem for some classes of HA makes that the analysis of several hybrid systems formalisms is made after their translation in HA (Cassez and Roux, 2003; Lime and Roux 2003).

4. Continuous Petri nets

Continuous Petri nets were introduced by David and Alla, (1987) as an extension of traditional Petri nets where the marking is fluid. A transition firing is a continuous process and consequently the state equation is a differential equation. A continuous PN allows, certainly, the description of positive continuous systems, but it is also used to approximate modeling of discrete event systems (DES). The main advantage of this approximation is that the number of events occurring is considerably smaller than for the corresponding discrete PN. Moreover, the analysis of a continuous PN does not require an exhaustive enumeration of the discrete state space.

[2] CMC: *http ://www.lsv.ens-cachan.fr/_fl/cmcweb.html/*

[3] UPPAAL : *http ://www.uppaal.com/*

[4] KRONOS: *http ://www-verimag.imag.fr/TEMPORISE/kronos/*

As for classical (discrete) Petri nets. We can define two types of continuous Petri nets, namely: autonomous continuous Petri nets and non-autonomous continuous Petri nets.

An autonomous continuous PN allows a qualitative description of continuous dynamic systems, it is defined as follows:

Definition 2 (autonomous continuous Petri Net): An autonomous continuous Petri net is a structure PN = (P, T, Pre, Post, M_0) such that:

1. P = {P_1, P_2, …, P_m} is a nonempty finite set of m places ;

2. T = {T_1, T_2, …, T_n} is a nonempty finite set of n transitions ;

3. Pre : P x T → R + is the pre-incidence function that associates a positive rational weight for each arc (T_j, P_i) ;

4. Post : P x T → R + is the post-incidence function that associates a positive rational weight for each arc (P_i, T_j) ;

5. M_0 : P → R + in the initial marking ;

The following notations will be considered in the sequel:

°T_J is the set of input places of the transition T_J.

T°$_J$ is the set of output places of the transition T_J.

As in a classical PN, the state of a continuous PN is given by its marking; however, the number of continuous PN reachable markings is infinite. That brought David and Alla (2004) to group several markings into a macro-marking. The notion of macro-marking is defined as follows:

Definition 3 (macro-marking): Let PN be an autonomous continuous PN and M_k its marking at time k. M_k may divide P (the set of places) into two subsets:

1. $P^+(M_k)$: The set of places with positive marking ;

2. $P^0(M_k)$: The set of places whose marking is null ;

A Macro-marking is the set of all markings which have the same subsets P + and P^0. A macro-marking can be characterized by a Boolean vector as follows:

V : P → {0, 1}

$$P_i \rightarrow \begin{cases} 1 \ \text{si} \ \ \text{Pi} \ \in P^+ \\ 0 \ \text{si} \ \ \text{Pi} \ \in P^0 \end{cases}$$

The concept of macro-marking was defined as a tool that permits to represent in a finite way, the infinite set of states (markings) reachable by a continuous PN. The number of reachable macro–marking of an n–place continuous PN is less than or equal to 2^n, even if the continuous PN is unbounded, since each macro marking is based on a Boolean state. A macro–marking is denoted m^*_j

Example 3: Let us consider again the hydraulic system of example 1, and consider that the supplying valve is in position A. In this position only the tank 1 is supplied, it is also emptied. While tank 2 is only emptied. The levels of liquid in tanks 1 and tanks 2 are, initially, of H_1 and H_2 respectively.

The continuous PN shown in Figure 5(b) describes the behavior of the system of tanks. Note that the numerical values of the valves flows cannot be represented in an autonomous CPN. The continuous transitions, T1, T2, and T3 represent only a positive flow for the three valves. Places and transitions of the continuous PN are represented with double line to distinguish them from places and transitions of a discrete PN. The firing of transitions T_1, T_2 and T_3 represents material flow through the valves. The marking of places P_1 and P_2 represents quantities of liquid in tank 1 and tank 2 respectively. Figure 5(c) represents the reachability graph; it contains all macro-marking reachable by the continuous PN.

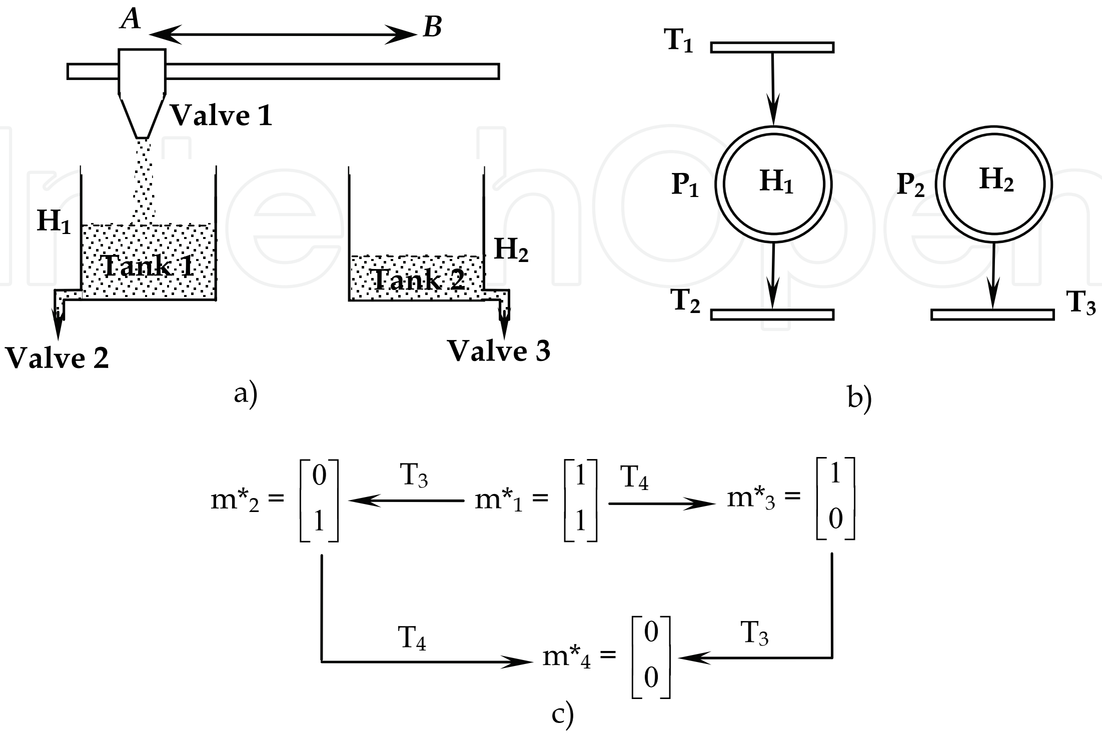


Fig. 5. a) System of tanks, b) Continuous PN describing the system of tanks, c) Reachability graph for the continuous PN

From the basic definition of autonomous continuous PNs, several researchers have defined several timed continuous PNs formalisms. Among these formalisms, we will present the first model to be defined which is always the most studied model, which is constant speed continuous Petri nets. It is defined as follows:

Definition 4 (Constant speed continuous Petri nets): A constant speed continuous Petri net is a structure PN_C = (PN, V) such that:

- PN is an autonomous continuous PN.
- V : $T \rightarrow R^+$
  $T_j \rightarrow V_j$
  is a function that associates to each transition T_j its maximal firing speed V_j.

In a CCPN, a place marking is a real number that evolves according to transitions instantaneous firing speeds. An instantaneous firing speed $v_j(t)$ of a continuous transition T_j can be seen as the flow of markings that crosses this transition. It lies between 0 and V_j for the transition T_j. The concept of validation of a continuous transition is different from the traditional concept met in discrete PNs. We consider that a transition of a CCPN can have two states:

1. The state strongly enabled, if

$$\forall\ P_i \in {}^{\circ}T_j,\ P_i \in P^+$$

   Here, the transition T_j is fired at its maximal firing speed V_j;
2. The state weakly enabled, if

$$\exists\ P_i \in {}^\circ T_j,\ P_i \in P^0$$

In this case, the transition T_j is fired at a speed v_j lower than its maximum firing speed.
The state equation in a CCPN is as follows:

$$\dot{m} = W.v(t) \tag{4}$$

Where W is the PN incidence matrix. This implies that the evolution in time of the state of a CCPN is given by the resolution of the differential equation (4), knowing the instantaneous firing speed vector. The evolution of a CCPN in time is given by a graph whose nodes represent instantaneous firing speed vectors. Each node is called a phase. In addition, each transition is labeled with the event indicating the place whose marking becomes nil and causes the changing of the speed state. The duration of a phase is also indicated. For more details, see (David and Alla, 2004).

Example 4: Let us consider again the system of tanks, where we associate to each valve its flow rate (figure 6 (a). Moreover, we consider that tank 1 and tank 2 contain initially 70 litres and 36.4 litres respectively. This system is described with the CCPN in Figure 6 (b). The only difference between this model and the autonomous continuous PN in Figure 5 (b) is that with each transition is associated a maximal firing speed.

Since all the places are initially marked, all the instantaneous firing speeds are equal to their maximal value. The marking balance for each place is given by the input flow minus the output flow; then:

At initial time t = 0, v_1 = 12, v_2 = 7, v_3 = 3, then $\dot{m}_1 = 7$ and $\dot{m}_2 = -7$.

Markings m1 and m2 evolve initially according to the following equations, respectively:

$m_1 = 70 + 7.t$

$m_2 = 36.4 – 7.t$

At time t = 5.2 the marking m_2 becomes nil, which defines a new dynamics for the system, as follows:

v_1 = 12, v_2 = 5, v_3 = 7, then $\dot{m}_1 = 7$ and $\dot{m}_2 = 0$.

And after time 5.2, $m_1 = 106.4 + 7.t$ and $m_2 = 0$

This last dynamics is a stationary behavior for the modelled system.

The curves in Figure 7(a) and 7(b) schematize marking m_1 and m_2 dynamics. These plots are made with the software SIRPHYCO[5]. This tool permits the simulation of discrete, continuous and hybrid PNs. The evolution of this model in time can be described thanks to the evolution graph in Fig. 6-c-. It can be noticed that the marking of place P_1 is unbounded while the number of nodes is finite and equal to 2.

5. Hybrid Petri nets

Continuous PNs are used for modeling continuous flow systems; however, this model does not allow logical conditions or discrete behavior modeling (e.g. a valve may be open or closed). For permitting modeling of discrete states, hybrid PNs were defined (David and Alla, 2001). In a hybrid PN, the firing of a continuous transition describes the material flow,

[5] SIRPHYCO: *http://www.lag.ensieg.inpg.fr/sirphyco/*

while the firing of a discrete transition models the occurrence of an event that can, for example, change firing speeds of the continuous transitions.

We find in the literature several types of continuous PN (David and Alla, 2004) and several types of discrete PN integrating time (Ramchandani, 1974; Merlin, 1974). In the autonomous hybrid model definition, there are no constraints on discrete and continuous part types. The most used, which is also the first formalism to be defined, is simply called the hybrid Petri net. It combines a CCPN and a T - timed PN. The combination of these two models confers to the hybrid model a deterministic behavior. It is used for the performance evaluation of hybrid systems.

D-elementary hybrid PNs are another type of hybrid PN formalism. They combine a time PN and a constant speed continuous PN (CCPN) (David and Alla 1987). Time PNs are obtained from Petri nets by associating a temporal interval with each transition. They are used as an analysis tool for time dependent systems.

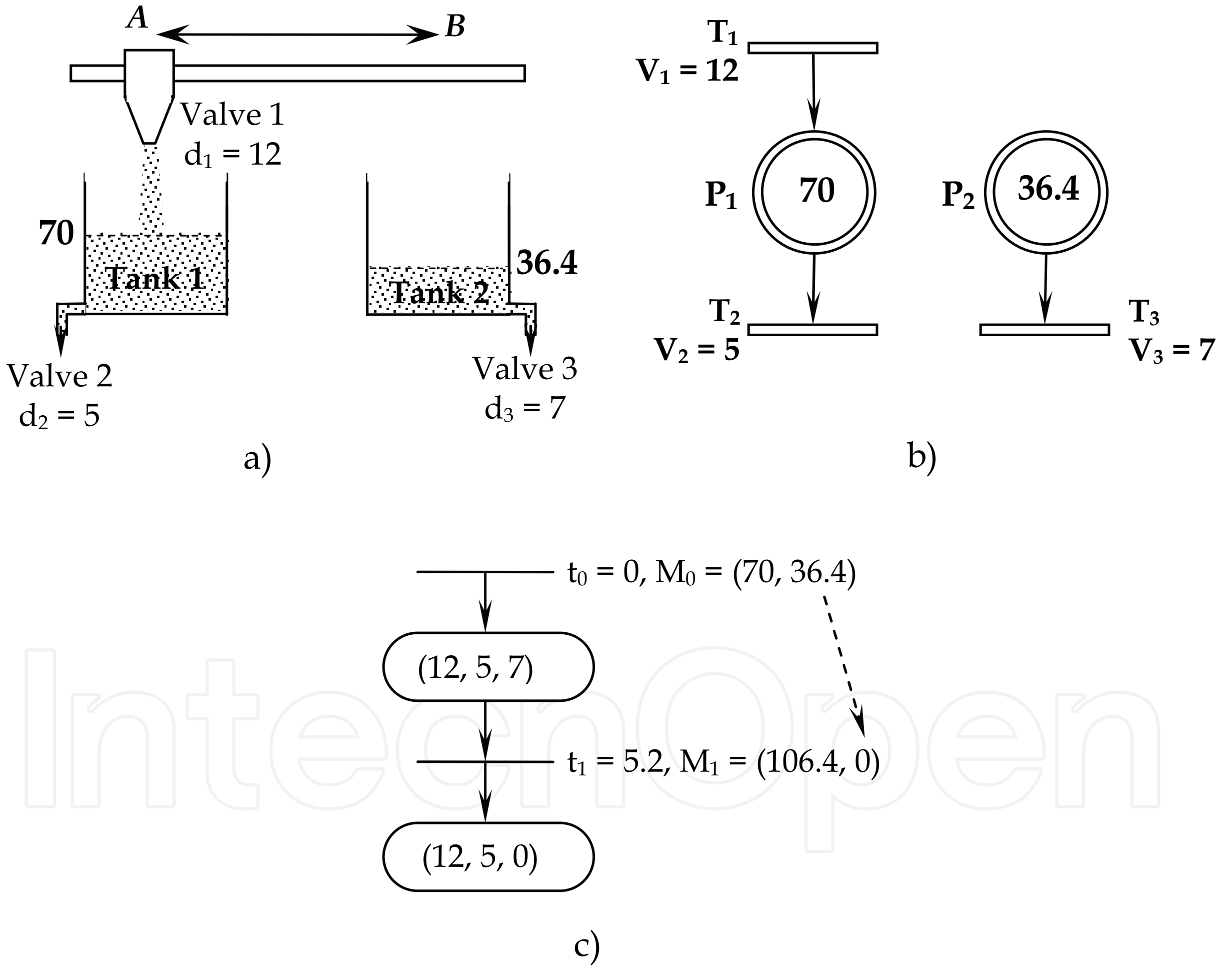


Fig. 6. a) System of tanks, b) Constant speed continuous PN describing the system of tanks, c) the evolution graph for the constant speed continuous PN

However, hybrid PNs were defined before D-elementary hybrid PNs. In order to simplify the presentation, we will start by defining D-elementary hybrid PNs.

5.1 D-elementary hybrid Petri nets

Definition 5 (D-elementary hybrid PNs): A D–elementary hybrid PN is a structure PN_H = (*P*, *T*, *Pre*, *Post*, *h*, *S*, *V*, M_0) such that:

1. *P* = $\{P_1, P_2, \ldots, P_m\}$ is a finite set of m places;
2. *T* = $\{T_1, T_2, \ldots, T_n\}$ is a finite set of n transitions;

We denote PD = {P1, P2,…, Pm′} the set of m′ discrete places (denoted by D–places and drawn as simple circles) and TD {T1, T2,…,Tn′} the set of the n′ discrete transitions (denoted by D–transitions and drawn as black boxes). PC = P – PD and TC = T – TD denote respectively the sets of continuous places (denoted by C–places and drawn with double circles) and continuous transitions (denoted by C–transitions and drawn as empty boxes).

1. *Pre* : P x T → N and *Post* : P x T → N are the backward and forward incidence mappings. These mapping are such that:

$$\forall\ (P_i, T_j) \in P^C \times T^D,\ \text{Pre}\ (P_i, T_j) = \text{Post}\ (P_i, T_j) = 0;$$

$$\text{And: } \forall\ (P_i, T_j) \in P^D \times T^C,\ \text{Pre}\ (P_i, T_j) = \text{Post}\ (P_i, T_j);$$

   This means that no arcs connect C–places to D–transitions, and if an arc connects a D–place P_i to a C–transition T_j, the arc connecting T_j to P_i must exist. This appears graphically as loops connecting D–places to C–transitions.

   These two conditions mean that, in a D–elementary hybrid PN, only the discrete part may influence the continuous part behavior, the opposite never occurs (the continuous part has no influence on the discrete part).
2. *h*: P∪T→{C, D} defines the set of continuous nodes, (h (x) = C) and discrete nodes, (h (x) = D).
3. *S*: $T^D \rightarrow R^+ \times (R^+ \cup \{\infty\})$ associates to each D–transition T_j its firing interval $[\alpha_j, \beta_j]$.
4. *V*: $T^C \rightarrow R^+$ associates a maximal firing speed V_j to each C-transition T_j.
5. M_0 is the initial marking; C–places contain non-negative real values, while D–places contain non-negative integer values.

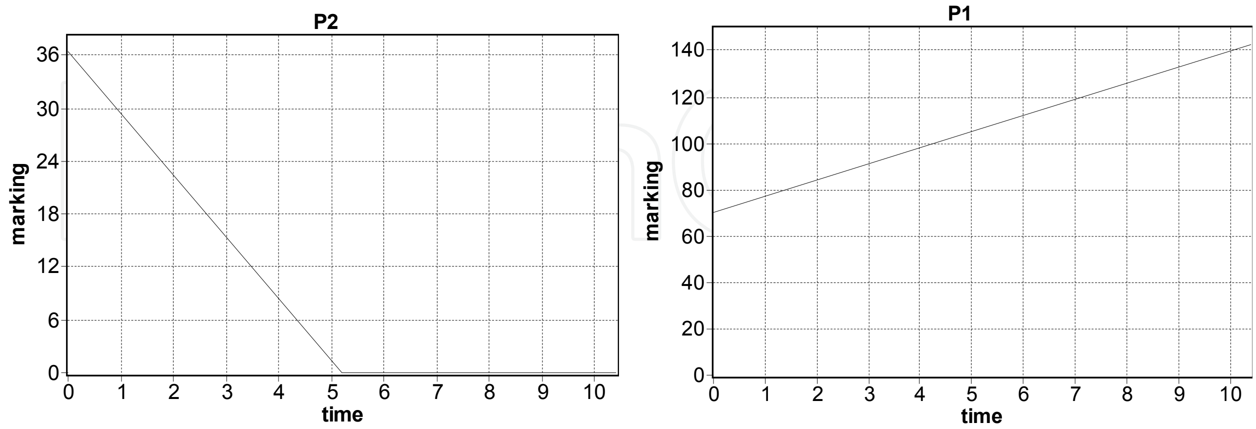


Fig.7. Temporal evolution of the marking of the PN in Fig. 6-b-

Example 5: Consider the system of tanks and suppose that valves 1 may be into the two positions A and B. The passage from position A to position B takes 0.5 seconds, but the commutation decision can be delayed indefinitely for the design of a control. This is why the time interval [0.5 ∞] is associated with the discrete transition T_1. On the other hand, the passage from position B to position A takes place after exactly 10 seconds from the last commutation (A→B). This is why the time interval [10, 10] is associated with the discrete transition T_2. The D-elementary hybrid PN in Figure 8 describes this hybrid system.

As a D-elementary hybrid PN combines a discrete and a continuous PN, its state at time t is given by the states of the two models. The strong coupling of these models makes it complex to analyze the hybrid model. Translating it into a hybrid automaton permits the use of tools and techniques developed for HA analysis. Ghomri *et al.* (2005) developed an algorithm permitting translation of a D-elementary hybrid PN into a HA. In the sequel, we briefly present this algorithm.

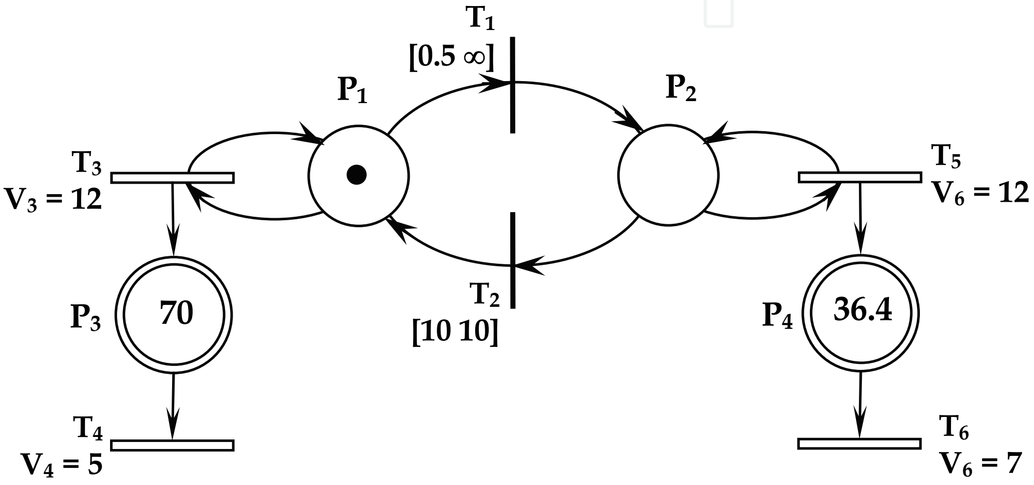


Fig.8. D-elementary hybrid Petri net describing the system if tanks

5.2 Translating D-elementary hybrid Petri nets into hybrid automata

It is, generally, very complex to translate a hybrid PN into a hybrid automaton because of the strong coupling between discrete and continuous dynamics. D-elementary hybrid PNs represent only a class of hybrid PNs, which permits modeling of frequently met actual systems: i.e. the class of continuous flow systems controlled by a discrete event system. The translation algorithm consists in separating the discrete and the continuous parts. Then, the translation into an automaton is performed in a hierarchical way. The algorithm is based on three steps as follows:

1. Isolate the discrete PN of the hybrid model and construct its equivalent timed automaton. Locations of the resulting timed automaton are said macro-locations.
2. Construct the hybrid automaton corresponding to each macro-location of the timed automaton resulting from the previous step.
3. Replace transitions between macro-locations by transitions between internal locations.

We detail these three steps through the following example.

Example 6: Consider the D-elementary HPN in Figure 8. Its discrete part is set again in Figure 9(a). The timed automaton corresponding to this time PN is represented in Figure 9(b).

To each location of the timed automaton, corresponds a marking of the time PN, and therefore a configuration of the CCPN. For instance, if P_1 is unmarked, T_3 may be eliminated from the CCPN in figure 8. The location S_2, for example, corresponds to the time PN marking vector $[m_1\ m_2]^T = [0\ 1]^T$, for which the continuous part is reduced to CCPN in Figure 10(a). This CCPN may be translated into the HA in Figure 10(b).

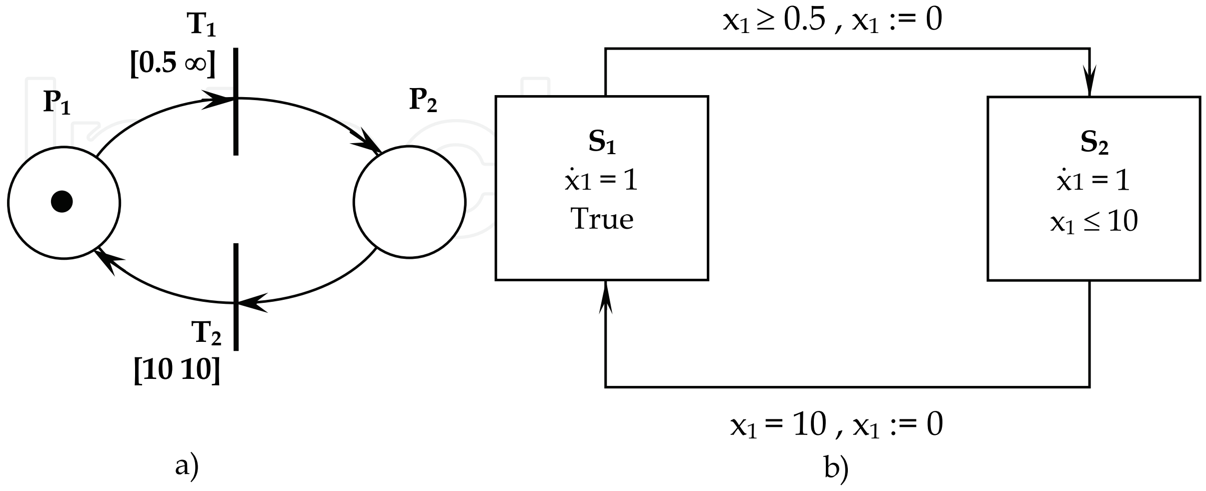


Fig.9. time Petri net and its equivalent time automaton

After the second step of the translation algorithm, we obtain a hierarchical form of a HA, formed from macro-locations each containing a HA describing the continuous dynamics in it. A generic representation of the model resulting after step 2 of the algorithm is given in Figure 11.

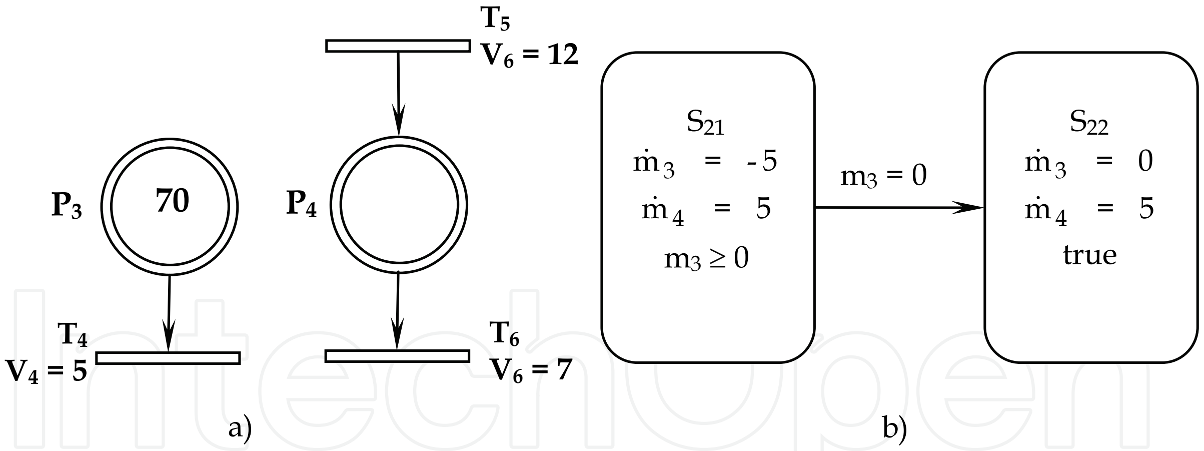


Fig.10. Constant speed continuous Petri net and its equivalent hybrid automaton

The location number of the resulting hybrid automaton depends on two parameters: (i) the location number of the TA describing the discrete part behavior, denoted as n; (ii) the continuous place number of the continuous part, denoted as m. The first parameter n is finite for a bounded time PN; although the propriety of boundedness is undecidable for a time PN, there exist restrictive sufficient conditions for its verification (Berthomieu and Diaz 1991). This first parameter defines the macro-location number. The second parameter m defines the number of locations inside a macro-location. As mentioned before, we can always model the behavior of a continuous PN by a HA with a finite number of locations, even if the continuous PN is unbounded; this number is least or equal to 2^m. We have therefore a resulting HA that contains at the most ($n.2^m$) locations. This is an important result since it is generally impossible to bound a priori the number of reachable states in a hybrid PN.

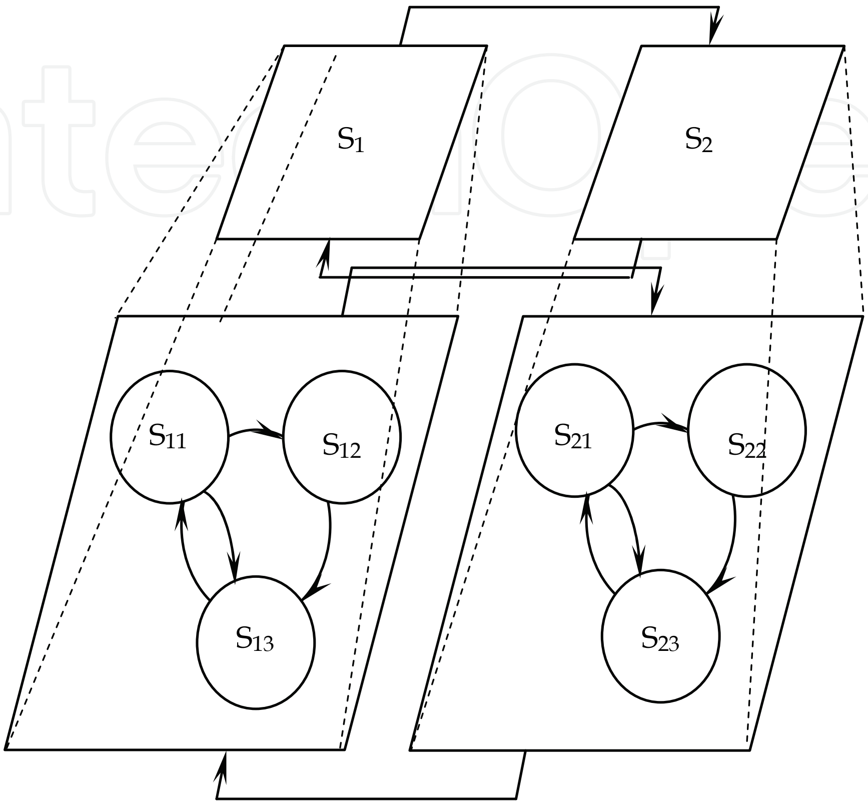


Fig.11. Generic schematization of model resulting from the second step of the algorithm

5.3 Hybrid Petri nets

A hybrid PN is distinguished from a D-elementary hybrid PN by the fact that the former contains a T-timed PN for modeling the discrete part—timed fixed values are associated with each transition—whereas the latter model contains a T-timed PN.

Definition 6 (hybrid Petri Net): A hybrid PN is a structure PN_H = (P, T, Pre, $Post$, h, S, V, M_0) such that :

1. $P = \{P_1, P_2, \ldots, P_m\}$ is a finite set of m places. $P = P^D \cup P^C$;
2. $T = \{T_1, T_2, \ldots, T_n\}$ is a finite set of n transitions. $T = T^D \cup T^C$;
3. *Pre* : P x T → N and *Post* : P x T → N are the backward and forward incidence mappings.

These mapping are such that:

$$\forall\ (P_i, T_j) \in P^D \times T^C, \text{Pre}\ (P_i, T_j) = \text{Post}\ (P_i, T_j);$$

1. $h : P \cup T \rightarrow \{C, D\}$ defines the set of continuous nodes, (h (x) = C) and discrete nodes, (h (x) = D).
2. $S : T^D \rightarrow Q^+$ associates to each D-transition T_j a duration d_j.
3. $V : T^C \rightarrow R^+$ associates a maximal firing speed V_j to each C-transition T_j.

4. *M_0* is the initial marking.

The condition on backward and forward incidence mappings means that, if an arc connects a D-place P_i to a C-transition T_j , the arc connecting T_j to P_i must exist. And vice versa. This appears graphically as loops connecting D-places to C-transitions. It means that a discrete token cannot be split by a continuous transition. The hybrid PN model, as defined below, allows modeling of the logical conditions, but it allows also the modeling of the transformation of a continuous flow into discrete parts and vice versa.

Example 7: Let us consider again the system tanks, and suppose that we have the following control strategy: we want to keep the liquid levels in tank 1 at least than a fixed level H_{max}. The hybrid PN in Figure 12 describes a system that satisfies this specification on the level in tanks.

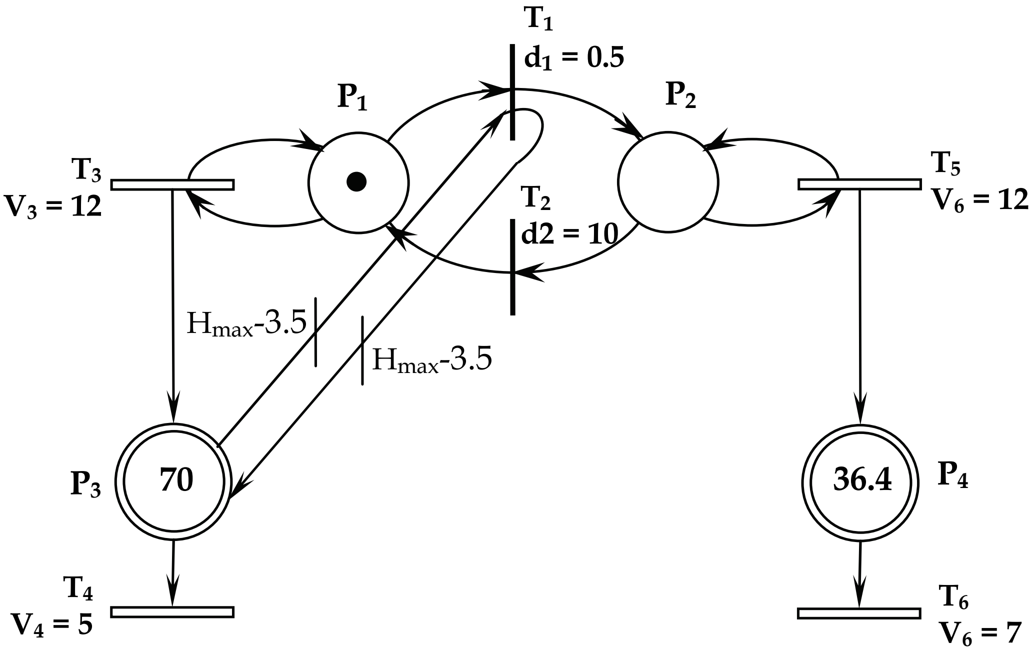


Fig.12. Hybrid Petri net describing the system of tanks with a restriction on its marking.

The weights (H_{max} – 3.5) associated with the arcs correspond to the minimal thresholds of tank 1 taking into account the delay 0.5.

6. Controller synthesis

The controller synthesis of HDS drifts directly from Ramadge and Wonham (1989) theory. They synthesize, from a discrete event system, a controller whose role is to forbid the occurrence of certain events. The controller decision to forbid an event depends only on the past of the system, i.e. of events, which already occurred. The aim is that the system coupled to its controller respects some given criteria.

Many researches were devoted to the problem of controller synthesis autonomous discrete event systems. This problem is thus well solved for this category of systems. The number of works relating to real time system controller synthesis is also very significant (Altisen *et al.* 2005). However, few works were devoted to solving this problem for the HDS (Wong-Toi, 1997; Antsaklis *et al.* 1993; Lennartson *et al.* 1994; Peleties & DeCarlo, 1994).

The controller synthesis of a dynamic system (autonomous, timed or hybrid) is generally based on three steps:

1. the behavioral description of the system (called an open loop system) by a model;
2. the definition of specifications required for this behavior;
3. the synthesis of the controller which restricts the model behavior to the required one, using a controller synthesis algorithm.

These algorithms consider the open system S and the specification on its behavior ϕ. and try to synthesize the controller C so that the parallel composition of S and C(S || C) satisfies ϕ. These algorithms use traditionally automata (finite state automata, timed automata and hybrid automata) because of their ease of formal manipulation; however, a model like HPN is preferred in the first step (the step of behavior description).

Consider an open loop Hybrid system; the aim of controller synthesis is to construct a controller that satisfies the specifications for the closed loop hybrid system. These specifications imply, generally, restrictions on the closed loop hybrid system. They can be either (1) specifications on the discrete part (this type of specification forbids certain discrete states); or (2) specification on the continuous part; in this case the specification has the form of an invariant that the continuous state must satisfy. This implies that the continuous state of the closed loop hybrid system is restricted to a specified region. The open problem is synthesizing the guards associated with the controllable transitions so that the specifications are respected leading to a maximal permissive controller.

7. Conclusion

Some extensions of PNs permitting HDS modeling were presented here. The first models to be presented are continuous PNs. This model may be used for modeling either a continuous system or a discrete system. In this case, it is an approximation that is often satisfactory.

Hybrid PNs combine in the same formalism a discrete PN and a continuous PN. Two hybrid PN models were considered in this chapter. The first, called the hybrid PN, has a deterministic behavior; this means that we can predict the occurrence date of any possible event. The second hybrid PN considered is called the D-elementary hybrid PN; this model was conceived to be used for HPN controller synthesis.

Controller synthesis algorithms consider the open system S and the specification on its behavior ϕ and try to synthesize the controller C, so that the parallel composition of S and C(S || C) satisfies ϕ. These algorithms use traditionally automata (finite state automata, timed automata and hybrid automata) because of their ease of formal manipulation; however, this model is not the most appropriate for behavior description. For coupling the analysis power of hybrid automata with the modeling power of hybrid PNs, an algorithm permitting translation of D-elementary hybrid PNs into hybrid automata was presented. Our future research aim is to generalize the existing results to the control of hybrid systems modeled by hybrid PNs.

8. References

Altisen, K. ; Bouyer, P. Cachat, T. Cassez F. & Gardey G. (2005) Introduction au contrôle des systèmes temps-réel, Proceedings of modélisation des systèmes réactifs, (MSR'05), France

Alur, R. Courcoubetis, C. Halbwchs, N. Henzinger, T.A. Ho, P. H. Nicolin, X. Olivero, A. Sifakis J. & Yovine, S. (1995) The Algorithmic Analysis of Hybrid Systems, *Theoretical computer science*, Vol. 138, pp.3-34

Antsaklis, P.J. Stiver, J.A. Lemmon, M.D. Hybrid system modeling and autonomous control systems, in: R.L. Grossman, A. Nerode, A.P. Ravn, H. Rischel, (1993) Hybrid Systems, in: Lecture Notes in Computer Science, vol. 736, Springer-Verlag, pp. 366–392.

Berthomieu, B. & Diaz, M. (1991) Modelling & verification of time dependent systems using time Petri nets, IEEE Transactions on software engineering, vol. 17(3), pp 259-273

Branicky, M. Borkar, V. Mitter, S. (1994) A unified framework for hybrid control, in proceedings of IEEE Conference and Decision and Control, CDC, Lake Buena Vista, USA, pp. 4228–4234.

Cassez, F.; & Roux, O.H. (2003) Traduction Structurelle des Réseaux de Petri Temporel vers les Automates temporisés, *Proceedings of 4ième colloque Francophone sur la modélisation des systèmes réactifs, (MSR'03)*, Metz, France

David, R.; & Alla, H. (2001) On Hybrid Petri Nets. *Discrete Event Dynamic systems: Theory & Applications*. vol. 11, pp. 9-40

David, R.; & Alla, H. (2004) *Discrete, Continuous, & Hybrid Petri Nets,* Springer

David, R.; Alla, H. (1987) Continuous Petri Nets, *Proceedings of the 8th European Workshop on Application & Theory of Petri Nets*, Saragossa, Spain, pp. 275-94

David, R.; Alla, H. Autonomous, (1990) Timed and Continuous Petri Nets, *Proceedings of the 11th Int. Conf. on Application & Theory of Petri Nets,* Paris, France, pp. 367-386

Ghomri, L. Alla H. & Sari, Z. (2005) Structural & hierarchical translation of hybrid Petri nets in hybrid automata, *Proceedings of IMACS'05*, Paris, France

Henzinger,; T.A. Kopke, P.W. Puri, A. & Varaiya. P. (1995) What's decidable about hybrid automata? *Proceedings of* 27th *Annual Symposium on Theory of Computing*, ACM Press, 373-382

Lennartson, B. Egardt, B. Tittus, M. (1994) Hybrid systems in process control, in: Proc. of the 33rd CDC, Orlando, FL, USA, 1994, pp 3587–3592

Merlin, P.; (1974) A study of the recoverability of computer system, PhD thesis, Dep. comput. Sci, Univ. California, Irvine,

Murata, T. (1989) Petri-nets: properties, analysis and applications, in proceedings of IEEE 77 (4) 541–580.

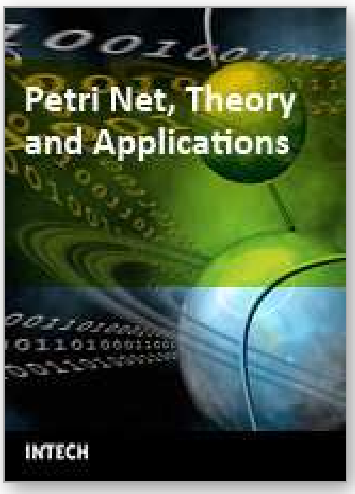
Peleties, P. DeCarlo, R.A. (1994) A modeling strategy for hybrid systems based on event structure, Discrete Event Dynamic Systems: Theory and Applications 30 (9) 1421–1427

Pettersson, S. Lennartson, B. (1995) Hybrid modelling focused on hybrid Petri nets, in: 2nd European Workshop on Real-Time and Hybrid Systems, Grenoble, France, pp. 303–309.

Ramadge, J.G.; & Wonham, W.M. (1989) The Control of Discrete Event Systems, *Proceedings of the IEEE*, vol. 77(1), 81-97

Ramchandani, C., (1974) Analysis of Asynchronous Concurrent Systems by Timed Petri Nets. PhD thesis, MIT, Cambridge, Feb

Wong-Toi, H.; (1997) The synthesis of controllers for linear hybrid automata. *In Proceedings of the 36th Conference on Decision & Control. IEEE* Computer Society Press 4607 – 4612



Petri Net, Theory and Applications

Edited by Vedran Kordic

ISBN 978-3-902613-12-7
Hard cover, 534 pages
Publisher I-Tech Education and Publishing
Published online 01, February, 2008
Published in print edition February, 2008

Although many other models of concurrent and distributed systems have been de- veloped since the introduction in 1964 Petri nets are still an essential model for concurrent systems with respect to both the theory and the applications. The main attraction of Petri nets is the way in which the basic aspects of concurrent systems are captured both conceptually and mathematically. The intuitively appealing graphical notation makes Petri nets the model of choice in many applications. The natural way in which Petri nets allow one to formally capture many of the basic notions and issues of concurrent systems has contributed greatly to the development of a rich theory of concurrent systems based on Petri nets. This book brings together reputable researchers from all over the world in order to provide a comprehensive coverage of advanced and modern topics not yet reflected by other books. The book consists of 23 chapters written by 53 authors from 12 different countries.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Latefa Ghomri and Hassane Alla (2008). Modeling and Analysis of Hybrid Dynamic Systems Using Hybrid Petri Nets, Petri Net, Theory and Applications, Vedran Kordic (Ed.), ISBN: 978-3-902613-12-7, InTech, Available from: http://www.intechopen.com/books/petri_net_theory_and_applications/modeling_and_analysis_of_hybrid_dynamic_systems_using_hybrid_petri_nets

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821