HAL open science

Post and pre-initialized stopwatch Petri Nets

Adib Allahham, Hassane Alla

To cite this version:

Adib Allahham, Hassane Alla. Post and pre-initialized stopwatch Petri Nets. DCDS 2007 - 1st IFAC Workshop on Dependable Control of Discrete-event Systems, Jun 2007, Cachan, France. pp.69-74. hal-00157454

HAL Id: hal-00157454

https://hal.science/hal-00157454v1

Submitted on 26 Jun 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

1st IFAC Workshop on Dependable Control of Discrete Systems (DCDS'07)
ENS Cachan, France - June 13-15, 2007

POST AND PRE-INITIALIZED STOPWATCH PETRI NETS

Adib Allahham * **Hassane Alla** *

* *GIPSA-Lab, Dpartement d'Automatique - BP 46*
38402 Saint Martin d'Hères, France
Email: {Adib.al-lahham,hassane.alla}@inpg.fr

Abstract: A modeling paradigm called Post and Pre-initialized Stopwatch Petri Nets ($SWPN$) is introduced. It extends Time Petri Nets TPN to the concept of stopwatch with an original mechanism of stopwatches reset. $SWPN$ makes this reset dependent on the firing of the corresponding transitions. The resulting model permits natural description of so-called preemption-resume behavior. We give the formal semantics of $SWPN$ as a timed transition system. We propose also a method for its analysis consisting in the computation of its equivalent stopwatch automaton SWA. The advantage of Petri Nets for modeling the complex system in concise way is combined with the power analysis of SWA.*Copyright ©2007 IFAC*

Keywords: T-time Petri Net, Stopwatch automata, timing analysis.

1. INTRODUCTION

Interruptible systems are often found in real-time systems that are typically composed of several tasks that interact. The behavior of these systems is so-called preemption-resume behavior where the some of its tasks are suspended and resumed latter. An important problem consists in ensuring that the tasks can be executed in such a way that they respect the constraints they are subjected to, as deadlines for example. So the influence of interruptions must be considered in modeling and verifying these systems. This requires to describe the suspension and resuming of tasks. To fulfill these needs, several models based on the notion of Stopwatch (a clock that can be stopped and resumed) have been proposed in the literature. One can find stopwatch extensions of classical dense time model: Timed Automata TA and time Petri Nets $TPNs$. Among the extensions of the timed automata, the Stopwatch Automata SWA (Cassez and Larsen (2000)) are defined as a subclass of the linear hybrid automata. In this model the increasing clock rate can be switched between (1) and (0) in order to express the progression or the suspension of a task across different logical locations. Petri Nets are another widely used mode for real-time system. They permit generally both, to facilitate the description of conditions, and to enhance the readability of the specifications which is not the case for direct modeling in SWA. Therefore, several efforts have been done to model the preemptive behavior for scheduling purposes by using some extensions of Time Petri Nets TPN. Lime and Roux (2004) proposed an extension called Scheduling Extended Time Petri Nets that consists in mapping into a PN model the way the different schedulers of a system activate or suspend tasks. For a fixed priority scheduling policy, Scheduling-$TPNs$ introduce two new attributes associated with each place that respectively represent allocation and priority. Bucci et al. (2004) proposed a similar model: Preemptive Time Petri Nets. The two attributes are associated with transitions instead of places. Roux and Lime (2004) defined Time Petri Nets with Inhibitor Hyperarcs as an extension of T-TPN. The stopwatch

associated with a transition can be reset, stopped and started by using classical arcs and branch inhibitor hyperarcs. Firing of a transition may be interrupted if there is a nonempty place connected to this transition by an inhibitor arc.

The Petri net extensions mentioned above introduce in the model either the priority or the inhibitor arcs for modeling the suspension and resuming. These attributes increase the difficulties of modeling complex systems as manufacturing systems. The need of a simple modeling tool to model preemption-resume behavior motivates us to propose the Post and Pre-initialized Stopwatch Petri nets. This model is referred in the sequel as *Stopwatch Petri Nets*(*SWPN,s* for short). An important contribution in our proposed model *SWPN* is the notion of pre-initialization and post-initialization of the clocks. In a *TPN*, only the pre-initialization is used. The clocks are initialized when its transitions are newly enabled. In *SWPN*, we introduce the post-initialization where the clocks are initialized when their associated transitions are fired. This mechanism of initialization of the variables associated with transitions is used in (Bobbio et al. (2000)) to widen the field of applicability of the stochastic petri nets, and thus this allows to model different preemption policies. The key difference with respect to our formulation is that between *TPN* and stochastic PN (there is no actual conflict in a stochastic Petri nets. For a detailed presentation of timed conflicts see (David and Alla (2005))). In turn, this difference reflects to the analysis method.

We tackle the reachability problem for the *SWPN* which it is, as all the extension of *TPN* mentioned above, undecidable. Our method for timing analysis is based on translation the *SWPN* into an equivalent *SWA*. Then, a forward analysis will be applied on the resulting *SWA* using *PHAVer* model-checker (Frehse (2005)). Thus, we bring the techniques used in *PHAVer* to force the termination, when the state space is not computable.

In Section 2 the *SWPN* model, its syntax and semantics are presented. The translation algorithm into a *SWA* is described in Section 3. Section 4 illustrates the proposed model and its timing analysis method by an example.

2. STOPWATCH PETRI NETS

We define the Stopwatch Petri Nets *SWPNs* which extends the basic model of T-time Petri Nets (Berthomieu and Diaz (1991)), to the concept of stopwatch with an original mechanism of stopwatches reset. It makes the reset of some timed transition dependent on the firing of the transitions (notion of post-initialization). In *SWPN*, there are two types of transitions: interruptible and non-interruptible transitions. The

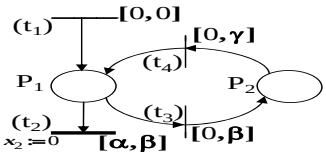

Fig. 1. *SWPN* of an interruptible task

firing of an interruptible transition resets the associated stopwatch, while the firing of another non-interruptible transition making the interruptible transition not enabled, suspends the progress of the time to fire which is then resumed when the transition is enabled again.

Example: Suppose there is an interruptible task having duration of execution $[\alpha, \beta]$ (without the durations of the interruption). After each interruption, the task resumes in the same place when it was interrupted. The interruption can occur at any instant during the task's execution. The maximal duration of each interruption is $\gamma \in \mathbb{R}^+$. The proposed *SWPN* models this behavior as shown in Figure 1. Place P_1 represents the execution of the task while place P_2 represents the interrupted state of task. Transitions t_3 and t_4 represent respectively the occurrence of interruption and resuming of the task. Transition t_2 which is an interruptible transition represents the task's execution. It is associated with stopwatch x_2. A graphical representation to distinguish between interruptible and non-interruptible transitions is to draw the former one in bold and to associate it by the assignment $x_2 := 0$. When an interrupt occurs, the token emerges from P_1 into P_2 and x_2 becomes inactive. When the interruption ends, transition t_4 fires, the task resumes in the same place where it was interrupted and x_2 comes back active and restores its saved value when the interruption happened. Firing t_2 resets the stopwatch x_2. At the initial marking, all the stopwatches of *SWPN* model reset to 0.

Definition 1. A Stopwatch Petri Nets is a tuple $\langle P, T, {}^\bullet(.), (.)^\bullet, M_0, I_s \rangle$, where:

- P is a non-empty finite set of places;
- T is a non-empty finite set of transitions. The set $T = T_{int} \cup T_{no-int}$ is composed of two disjoints sets: the interruptible and non-interruptible transitions;
- ${}^\bullet(.)$ and $(.)^\bullet$ are respectively the backward and forward incidence function;
- $M_0 \in \mathbb{N}^{card|P|}$ is the initial marking of the net, associating with each place p a natural number of tokens;
- I_s associates with each transition t_i a firing interval delimited by an earliest firing time $EFT(t_i) \in \mathbb{R}^+$ and a latest firing time $LFT(t_i) \in \mathbb{R}^+ \cup \{\infty\}$. □

2.1 Semantics:

A marking M of the net is an element of $\mathbb{N}^{card|P|}$ such that $\forall p \in P$, $M(p)$ is the number of token in the place p. A transition t_i is to said to enabled by marking M if $M \geq {}^{\bullet}t$ and denoted by $t_i \in enabled(M)$. □

Let be $\upsilon(t_i)$ represents the values of the stopwatch associated to t_i. $\upsilon(t_i)$ is a mapping $\upsilon \in (R^+)^{card|T|}$ such that:

• $\forall t_i \in T_{int}$, $\upsilon(t_i)$ is the time elapsed since t_i was first enabled and during which t_i remained enabled. t_i is first enabled when it is enabled and its value $\upsilon(t_i) = 0$. Remind that $\upsilon(t_i) \leftarrow 0$ at each time t_i fires;

• $\forall t_i \in T_{no-int}$, $\upsilon(t_i)$ is the time elapsed since t_i was last enabled. □

A transition $t_i \in T_{int}$ is said to be suspended by the marking M and denoted by $susp(M)$ if :

$t_i \in T_{int} : t_i \in susp(M)$ if ${}^{\bullet}(t_i) > M \wedge \upsilon(t_i) > 0$.□

A transition t_i is said to be $firable$ by marking M and denoted by $t_i \in firable(M)$ if:

$t_i \in enabled(M) \wedge EFT(t_i) \leq \upsilon(t_i) \leq LFT(t_i)$. □

A transition $t_i \in T_{no-int}$ is said to be newly enabled by firing the transition t_k from the marking M and denoted by $\uparrow enabled(t_i, M, t_k)$ if: $(t_i \notin enabled(M) \vee (t_i = t_k)) \wedge ({}^{\bullet}(t_i) \leq M - {}^{\bullet}(t_k) + (t_k)^{\bullet})$.

Notion of pre-initialization: The transition t_i is called Pre-initialized, if we initialize it *before* its using. The concept of Pre-initialization is possible when t_i is newly enabled, i.e, $t_i \in \uparrow enabled(t_i, M, t_k)$. □

Notion of Post-initialization: The transition t_i is called Post-initialized, if we initialize it *after* its using. The concept of Post-initialisation is possible only if $t_i \in T_{int}$. □

The semantics of $SWPN$ as *Timed Transition Systems* (TTS) are defined as follows.

Definition 2. (Semantics of a $SWPN$) The semantics of a Stopwatch Petri Net N is defined as a TTS $S_N = (Q, q_0, \rightarrow)$ such that:

• $Q = \mathbb{N}^p \times (\mathbb{R}^+)^{card|T|}$.

• $q_0 = (M_0, \overline{0})$. In this state, the value of all the stopwatches is reset.

• $\rightarrow \in Q \times (T \cup \mathbb{R}) \times Q$ is the transition relation including a continuous transition relation and a discrete transition relation.

• the continuous transition relation is defined $\forall d \in \mathbb{R}^+$ by:

$$(M,\upsilon) \xrightarrow{d} (M, \upsilon') \text{ iff } \forall t_i \in T$$
$$\begin{cases} \upsilon'(t_i) = \begin{cases} \upsilon(t_i) + d \text{ if } t_i \in enabled(M) \\ \upsilon(t_i) \text{ Otherwise} \end{cases} \\ M \geq {}^{\bullet}(t_i) \Rightarrow \upsilon' \leq LFT(t_i) \end{cases}$$

• the discrete transition relation is defined $\forall t_i \in T$ by:

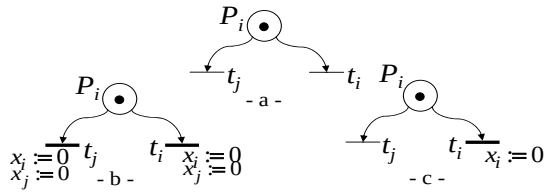

Fig. 2. Stopwatch reset in different conflict cases

$$(M,\upsilon) \xrightarrow{t_i} (M', \upsilon') \text{ iff } \forall t_i \in T$$
$$\begin{cases} t_i \in firable(M), \\ M' = M - {}^{\bullet}(t_i) + (t_i)^{\bullet}, \\ \upsilon'(t_i) := 0 \text{ if } t_i \in T_{int}, (Post - initialization) \\ \forall t_k \in T, \upsilon'(t_k) = \begin{cases} \bullet\ 0 \text{ if } t_k \in \uparrow enabled(t_k, M, t_i) \\ (pre - initialization) \\ \bullet\ \upsilon(t_i) \text{ if } t_k \in enabled(M) \\ \bullet\ \theta(t_k) \text{ if } t_k \in susp(M) \\ \wedge t_k \in susp(M') \end{cases} \end{cases}$$ □

In discrete transition relation, the value υ' of any transition in marking M' is different for persistent-progressing transitions and persistent-suspended transition. For the first one, it represents the time elapsed in the previous state while for the second one υ' represents the value of the stopwatch when it was suspended in a previous state (at the value $\theta(t_k)$), remains frozen in the previous state characterized by the marking M and stays frozen in the current state M'.

• ***Stopwatches reset in conflict case***

Figure 2 represents parts of $SWPN$. There is an effective conflict, namely $\langle P_i, \{t_i, t_j\}\rangle$.

With regards to the reset of stopwatches associated with the transitions affected by a conflict, we distinguish the following cases according to the transitions types in conflict. The stopwatches which are initialized by a transition firing are represented by a set of assignments $x_i := 0$ shown close to this fired transition:

• $t_i, t_j \in T_{no-int}$. This is the general behavior of T-Time Petri Nets (Fig.2.a).

• $t_i, t_j \in T_{int}$. The two stopwatches associated with t_i and t_j will be reset by firing one of the two transitions (Fig.2.b).

• $t_i \in T_{int}$ and $t_j \in T_{no-int}$. Only the stopwatch associated with t_i will be reset by its firing and the other will be reset when the corresponding transition is newly enabled (Fig. 2.c).

2.2 Timing Analysis of SWPN

In TPN, if the limits of the timed intervals associated to the transitions are rational numbers, and if the associated PN is bounded then the TPN is bounded. Outside of precedent sufficient boundedness conditions, only the computation of the state space allows to determine if a $SWPN$ is bounded or not. Techniques for reducing the infinite state space to a finite one are necessary.

Thus, we are interested in bounded $SWPN$ for which we propose a technique for timing analysis. The values of stopwatches in a state of a $SWPN$ can be described by a complex non-convex polyhedron. So, computing its state space leads to polyhedra of increasing complexity and the number of polyhedra may grows without bounds and the reachability algorithm does not terminate. For that, we propose firstly to translate the $SWPN$ to an equivalent stopwatch automaton. Then, thanks to the hybrid systems analysis tools such as model-checker *PHAVer* (Frehse (2005)), the manipulation of these polyhedron is possible. In *PHAVer*, if the invariants of the automaton is bounded, then the analysis termination can be forced by using the simplification techniques of a complex polyhedra. The simplification is done in a strictly conservative approach of the reachable set of states.

3. FROM STOPWATCH PETRI NET TO STOPWATCH AUTOMATA

We aim to translate a $SWPN$ into a SWA in order to perform a timing analysis of this $SWPN$, using efficient analysis SWA techniques and tools.

3.1 Stopwatch Automata

The stopwatch automata is basically defined as a class of linear hybrid automaton where the time derivative of a variable in a location can be either 0 or 1 (Cassez and Larsen (2000)).

Definition 3. A Stopwatch automaton is a 7-tuple $(L, l_0, X, \Sigma, A, I, Dif)$ where

- L is a finite set of locations,
- l_0 is the initial location,
- X is a finite set of positive real-valued stopwatches,
- Σ is a finite set of labels,
- $A \subset L \times C(X) \times \sigma \times 2^X \times L$ is a finite set of arcs. $a = (l, \delta, t, R, l') \in A$ is the arc between the locations l and l', with the guard δ, the label t and the set of stopwatches to reset R. $C(X)$ is the set of constraints over X.
- $I \in C(X)^L$ maps an invariant to each location,
- $Dif \in (\{0,1\}^X)^L$ maps an activity to each location, $\dot{X}$ being the set of derivatives of the stopwatches w.r.t time.$\dot{X} = Dif(l)(x)_{x \in X}$. Given a location l and a clock x, we will denote $Dif(l)(x)_{x \in X} = \{0, 1\}$. □

3.2 Reachability techniques analysis for SWA

A state of the SWA is a pair (L, E) where L is a location of SWA and E is a polyhedron representing its timed state space. There are two types of evolutions from a state (L, E), namely the continuous evolution by letting the time progress and the discrete evolution by firing an arc. Accordingly, there are two types of successors: continuous-successors denoted by $succ_t$ and discrete-successors denoted by $succ_d$. The interested readers can refer to (Alur et al. (1995)) for more details.

3.3 A Forward Algorithm to Compute the State Space of a bounded SWPN

The proposed method in this paper is an adaptation of the region based method for Timed Automaton. The algorithm starts from the initial state and explores all possible evolutions of the $SWPN$ by firing transitions or by elapsing a certain amount of time.

3.3.1. Labeling of marking graph: Let G be the marking graph of the $SWPN$. It can be easily be labeled to generate a Stopwatch automaton SWA bisimilar to the $SWPN$.
The pair $G = (M, A)$ is composed of:
• M is the set of possible markings of the $SWPN$: M_0,....,M_p,
• A is the set of arcs between the nodes of the marking graph : a_0,....,a_q.

The Stopwatch Automaton will be obtained by associating with each marking the differential equations that express the dynamics of stopwatches in this state (marking), and an invariant. Each arc between two reachable markings are associated with a guard and some clocks assignments. Each of these arcs corresponds to firing an enabled transition of the $SWPN$.
• ***Dynamic of stopwatches:*** A set of differential equations in the form $\dot{x} = c$ where $c = \{0, 1\}$ is associated with each marking M_k.
$\forall t_i \in enabled(M_k) : \dot{x_i} = 1$ and $\forall t_j \in suspended(M_k) : \dot{x_j} = 0$. The inactive transitions are not considered (t_m is inactive if if $(t_m \in T_{int} \wedge t_m \notin enabled(M_k) \wedge \upsilon(t_m) = x_m = 0)$ or $(t_m \in T_{no-int} \wedge t_m \notin enabled(M_k)$.
• ***Invariant:*** An invariant is associated with each marking M_k. By construction, in each marking, only the possible evolution of time is computed. In other words, only the active or suspended stopwatches will be represented in each location. Let X_k be the set of stopwatches associated with the enabled and suspended transitions for the marking M_k of the $SWPN$. Then, the invariant associated with M_k is defined by:
$I(M_k) = \{x_i \leq LFT(t_i) \mid t_i \in enabled(M_k)$ or $susp(M_k)\}$.
• ***Guard:*** Each arc a_k of the graph G corresponds to the firing of a transition t_i in $SWPN$. Then, we label a_k by:

□ the action name t_i,
□ the guard: $EFT(t_i) \leq x_i \leq LFT(t_i)$,
□ the clocks assignments: $x_k \leftarrow 0$ for all clocks x_k associated with a newly enabled transition t_k where $t_k \in T_{non-int}$ (Pre-initialization). If the fired transition $t_i \in T_{int}$, we associated also the clock assignment $x_i \leftarrow 0$ (Post-initialization). If $t_i \in T_{int}$ is in an effective conflict with $t_j \in T_{int}$ then we add $x_j \leftarrow 0$.

SWA_G denotes the stopwatch automaton obtained from labeling the marking graph. In this automaton, we denote each location according to its corresponding marking, i.e. $L_0....L_k$ correspond M_0,....M_k.

• ***The Algorithm for one iteration:***
The algorithm proposed is based on the forward analysis to find the reachable location of SWA_G. These reachable locations are accumulated in the pile Reach. At the initial state, Reach = $\{L_0\}$.
The computation of the reachable states from L_0 and the polyhedral E_0^e containing the values of stopwatches at the entry of L_0 is done as follows:

- Compute the possible evolution of time according to the active stopwatches in L_0 or the values of stopwatches for which L_0 could exist, i.e. values of stopwatches must not be greater than the latest firing time of enabled transitions: $E_0 = Succ_t(E_0^e)$.
- Determine the firable arcs which leaves L_0. $a_{0,k}$ is the arc which leaves L_0 to L_k. $a_{0,k}$ which is associated with the label t_i is firable if $E_0 \cap \{EFT(t_i) \leq x_i \leq EFT(t_i)\}$ is a non empty-polyhedron. Then, Update the set Reach by adding L_k: Reach = $\{L_0, L_k\}$.
- For each firable arc $a_{0,k}$ leading to a marking L_k, compute the values at the entering of L_k:
$$S_{0,k} = Succ_d(E_0)$$
$$E_k^e = S_{0,k} \wedge [X_e := 0]$$
where X_e is the set of stopwatches annotated with the affectation of $a_{0,k}$. E_k^e is a polyhedron for which the new marking L_k is reachable.
- Compute the possible evolution of time for which L_k could exist: $E_k = Succ_t(E_k^e)$.

3.4 Reachable stopwatch automaton SWA_{Reach}

The obtained stopwatch automaton by applying the forward analysis to SWA_G is defined as follows:

Definition 4. • L = Reach = $\{L_0, ..., L_k\}$ is the set locations, i.e. the set of reachable marking of *SWPN*. $L_0 = M_0$ is the initial marking,
• $X = \{x_1, x_2, ..., x_q\}$ is the set of stopwatches, i.e. the set of all stopwatches associated with the transitions,
• $\Sigma = \{t_1, ..., t_q\}$ is the set of labels, i.e. the transitions of *SWPN*,
• $A \subset L \times C(X) \times \Sigma \times 2^C \times L$ is the finite set of firable arcs between the reachable locations,
• $I : L \rightarrow C(X)$. □

3.5 Bisimulation between $SWPN$ and SWA_{Reach}

Definition 5. Let Q_N be the set of states of $SWPN$ N and Q_A the set of states of the reachable stopwatch automaton A. Let $R \subset Q_N \times Q_A$ be the relation between a state of the Stopwatch automaton and a state of the Stopwatch Petri Net defined by:

$$\begin{cases} \forall (M, \upsilon) \in Q_N \\ \forall (l, \bar{\upsilon}) \in Q_A \end{cases}, (M, \upsilon) R (l, \bar{\upsilon}) \Leftrightarrow \begin{cases} M = \mathbb{M}(l) \\ \upsilon = \bar{\upsilon} \end{cases}$$

where $\mathbb{M}$ is the function giving the associated marking of a SWA state l. □

***Theorem** 1.* R is a bisimulation: For all (M, υ), $(l, \bar{\upsilon})$ such that (M, υ) R $(l, \bar{\upsilon})$:

$$\bullet (M, \upsilon) \xrightarrow{t_i} (M', \upsilon') \Leftrightarrow \begin{cases} (l, \bar{\upsilon}) \xrightarrow{t_i} (l', \bar{\upsilon}') \\ (M', \upsilon') R (l', \bar{\upsilon}') \end{cases}$$

$$\bullet (M, \upsilon) \xrightarrow{d} (M, \upsilon') \Leftrightarrow \begin{cases} (l, \bar{\upsilon}) \xrightarrow{d} (l, \bar{\upsilon}') \\ (M, \upsilon') R (l, \bar{\upsilon}') \square \end{cases}$$

The proof is detailed in (Allahham and Alla (2006)).

4. AN ILLUSTRATIVE EXAMPLE

The priority problem in case of a sharing resource is considered in an example. Figure 3 represents the manufacturing process of two type of production S_1 and S_2 using two machines M_a, M_b and a sharing robot between S_1 and S_2. The token in P_1 means that machine M_a is operational and executes the task $1-1$ on S_1: transition t_1 can be fired and the interval $[2, 2]$ represents the duration of task $1-1$. Similarly, interval $[5, 7]$ represents the necessary duration for the robot to execute task $1-2$ on S_1. Machine M_b executes task $2-1$ on S_2. $[3, 3]$ represents the duration to execute this task on S_2. Robot executes also the task $2-2$ on S_2 and the duration for that is given by $[3, 4]$.
We suppose here that the task robot on S_2 has priority over the the task robot on S_1. This is modeled in the following way. If M_b finishes task $2-1$ on S_2 while the robot is executing task $1-2$ on S_1, the robot stops task $1-2$ on S_1 and executes immediately task $2-2$ on S_2. When

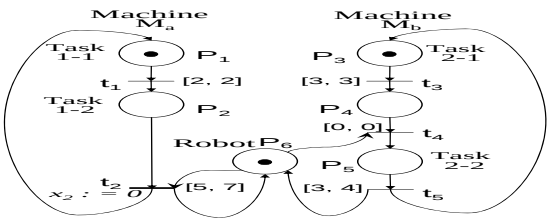

Fig. 3. *SWPN* of two tasks with different priority on one resource

it finishes task $2-2$, task $1-2$ on S_1 is resumed by robot exactly at the point where it was stopped. This behavior is modeled in Figure 3, where only transition t_2 associated with the assignment $x_2 := 0$ is interruptible. Figure 4 shows the stopwatch automaton corresponding to the *SWPN* in Figure 3. This is obtained applying the algorithm presented in Section 4 where the unstable state corresponding to transition t_4 has been suppressed. The Forward analysis has been executed using the model-checker (*PHAVer* Frehse (2005)). It is then possible to verify many properties of the system. For example, we can evaluate the system performances, especially those concerning machine M_a. It appears interesting to calculate the maximal duration where M_a remains inactive. For that, we take advantage of stopwatch automaton and introduce a variable y. It is intended to calculate the duration which separates the firing of transitions t_1 and t_2. Figure 5 shows the reachable state space for the variable y over all the locations of the automaton. We note that the maximum duration for which machine M_a is remaining inactive, is $y = 19$. Seeking properties depend on the system under study, our global approach gives all the needed formal results to achieve this goal.

5. CONCLUSION

Post and Pre-initialized Petri Nets *SWPN* extend T-time Petri nets to allowing the formal modeling of preemption-resume behavior. We have given a method for timing analysis of this *SWPN*. This includes a labeling algorithm of the marking graph to build a Stopwatch Automaton SWA_G. Then, a forward computation of the state space is executed by *PHAVer* model-checker. The resulting stopwatch automaton and *SWPN* are timed bisimilar.

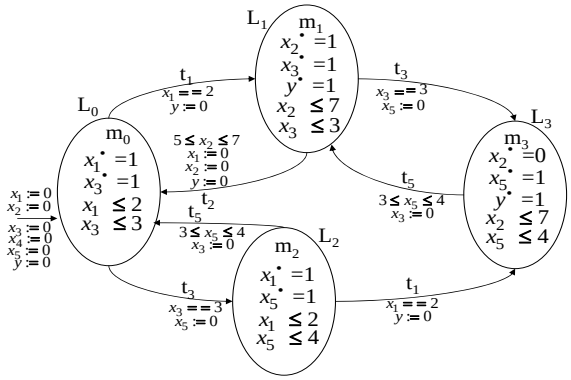

Fig. 4. Stopwatch Automaton of *SWPN* given in Figure 3

REFERENCES

A. Allahham and H. Alla. Monitoring the interruptible discrete events systems. Interne raport, Laboratoire d'Automatique de Grenoble - INPG, France, 2006.

R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzingerd, P.H. Hod, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algoritmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1), 1995.

B. Berthomieu and M. Diaz. Modeling and verification of time dependent systems using time petri nets. *IEEE Transaction Software Engineering*, 17(3), 1991.

Andrea Bobbio, Antonio Puliafito, and Miklós Telek. A modeling framework to implement preemption policies in non-Markovian SPNs. *IEEE Transactions on Software Engineering*, 26 (1):36–54, January 2000.

G. Bucci, A. Fedeli, L. Sassoli, and E. Vicario. Timed state space analysis of real-time preemptive systems. *IEEE Transactions on Software engineering*, 30(2):97–111, 2004.

F. Cassez and K.J. Larsen. The impressive power of stopwatch. pages 138–152, 2000.

R. David and H. Alla. *Discrete, Continuous and Hybrid Petri Nets*. Springer, 2005.

G. Frehse. Phaver: Algorithmic verification of hybrid systems past hytech. In *The Fifth International Workshop on Hybrid Systems: Computation and Control*, pages 258–273, 2005.

D. Lime and O. H. Roux. A translation based method for the timed analysis of scheduling extended time petri nets. In *25th IEEE International Real time Systems Symposuim*, pages 187–196, Lisbon, Portugal, 2004.

O. H. Roux and D. Lime. Time petri nets with inhibitor hyperarcs. formal semantics and state space computation. In *25th International Conference on theory and application of Petri nets*, pages 371–390, Bologna, Italy, 2004.

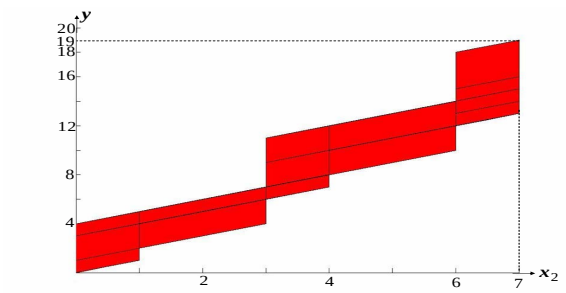

Fig. 5. Reachable space state of the variable y