Réseaux de Petri à chronomètres Post et Préinitialisés

Adib Allahham, Hassane Alla

Réseaux de Petri à chronomètres Post et Pré-initialisés

Adib Allahham, Hassane Alla

GIPSA-Lab, Département d'Automatique,
961, rue de la Houille Blanche – B.P. 46
38402, Saint Martin d'Hères
{adib.al-lahham, hassane.alla}@inpg.fr

RÉSUMÉ. *Un paradigme de modélisation appelé Réseaux de Petri à chronomètres Post et Pré-initialisés SWPN est introduit. Il étend les réseaux de Petri T-temporels en incluant le concept de chronomètre dans sa sémantique. Un mécanisme d'initialisation des chronomètres appelé Post-initialisation est utilisé. Ce mécanisme repose sur l'initialisation du chronomètre après le franchissement de la transition correspondante. Le modèle résultant permet de modéliser de façon naturelle la suspension et la reprise des tâches dans les systèmes temps-réel. Nous présentons d'abord la sémantique formelle de ce modèle sous la forme d'un système de transitions temporisé. Nous proposons ensuite une méthode d'analyse temporelle de ce modèle en le traduisant en automate à chronomètres. La vérification de propriétés temporelles peut alors être effectuée en utilisant l'automate à chronomètres ainsi obtenu. L'avantage de ce modèle est qu'il combine la concision du modèle réseau de Petri et la puissance d'analyse de l'automate à chronomètres.*

ABSTRACT. *A modeling paradigm called Post and Pre-initialized Stopwatch Petri Nets SWPN is introduced. It extends Time Petri Nets to the concept of Stopwatch with a mechanism of stopwatches reset called Post-initialization. It makes the reset of stopwatches dependent on the firing of the corresponding transitions. The resulting model permits natural description of so-called preemption-resume behavior. First we give the formal semantics of this extended model SWPN as a timed transition system. Then we propose a method for its analysis consisting in the computation of its equivalent stopwatch automaton SWA. Thus the verification of timing properties can be conducted and comes to analyze the such obtained SWA. The advantage of Petri Nets for modeling the complex system in concise way is combined with the power analysis of SWA.*

MOTS-CLÉS : *Réseaux de Petri T-temporels, Automate à chronomètres, Analyse temporelle, modélisation des systèmes interruptibles.*

KEYWORDS: *T-time Petri Net, Stopwatch automata, Timing analysis, Interruptible systems modeling.*

1. Introduction

Les systèmes interruptibles sont répandus dans les systèmes temps-réel. Ils sont généralement composés de plusieurs tâches qui interagissent. Une tâche peut être suspendue puis reprise au même endroit un peu plus tard. Un problème important consiste à assurer que les tâches peuvent être exécutées de telle façon qu'elles respectent certaines contraintes temporelles, comme la durée maximale d'exécution. Donc, l'influence des interruptions doit être considérée lorsqu'on modélise et vérifie ces systèmes. Cela exige de décrire la suspension et la reprise des tâches. Pour répondre à ces besoins, plusieurs modèles basés sur la notion de chronomètre ont été proposés dans la littérature. Ils étendent les modèles temps dense classiques au concept du chronomètre : Automate temporisé et Réseaux de Petri temporel.
Parmi les extensions des automates temporisés, les automates à chronomètres SWA [CAS 00], sont définis comme une sous-classe des automates hybrides linéaires pour laquelle les dérivées des variables par rapport au temps ne peuvent prendre que deux valeurs exprimant la progression (1) ou la suspension (0).
Les réseaux de Petri T-temporels TPN (T-Time Petri Nets) [BER 91] constituent un autre modèle largement répandu pour modéliser les systèmes temps réel. Ce sont des modèles de spécifications formelles qui allient les avantages d'une description graphique puissante et d'une sémantique formelle. Ils permettent d'introduire les spécifications de manière lisible, concise et simple. Ceci n'est pas le cas de la modélisation directe pour les SWA (composition synchrone). Par conséquent, un certain nombre d'auteurs ont proposé d'étendre les TPN afin de prendre en compte les aspects liés à l'interruption et à la reprise d'actions : Les Scheduling-TPN [LIM 04], les Preemptive-TPN [BUC 04], TPN à hyperarcs inhibiteurs ($IHTPN$) [ROU 04] et les réseaux de Petri Temporels à chronomètres ($SWTPN$) [BER 05]. Les deux premiers ajoutent des ressources et des priorités au modèle TPN, alors que les $IHTPN$ introduisent des arcs inhibiteurs qui contrôlent la progression des transitions. Nous pouvons remarquer que les attributs ajoutés augmentent la difficulté de modélisation des systèmes complexes comme les systèmes manufacturiers. Le besoin d'un outil simple qui permet de modéliser l'interruption et la reprise des tâches, nous a motivé à proposer une extension des TPN que nous appelons réseau de Petri à chronomètres post- et pré-initialisés. Dans la suite, nous appelons ce modèle réseau de Petri à chronomètres avec l'abbreviation $SWPN$.
Une contribution importante dans notre modèle repose sur les concepts de *Pré-* et *Post-initialisation* des horloges. Dans le modèle TPN, seulement le concept de *Pré-initialisation* est utilisé. Les horloges sont initialisées lorsque ses transitions correspondantes sont nouvellement sensibilisées. Dans le modèle $SWPN$, nous introduisons le concept de *Post-initialisation* où les horloges sont initialisées lorsque les transitions correspondantes sont franchies. Ce mécanisme d'initialisation des variables associées aux transitions est utilisé dans [BOB 00] pour les les réseaux de Petri stochastique SPN. Il est ainsi possible de modéliser la suspension et la reprise d'action. La différence principale par rapport à notre formalisme est celle qui existe entre les TPN et SPN. En conséquence, cette différence est liée à la méthode d'analyse.
Nous considérons le problème de calcul de l'espace d'états pour le nouveau modèle

$SWPN$, celui-ci est indécidable comme toutes les extension des TPN ci-dessus mentionnées. Dans ce but, nous proposons une méthode basée sur la traduction de $SWPN$ en SWA. Ensuite, une analyse en avant sera appliquée sur l'automate à chronomètres ainsi obtenu en utilisant un model-checker $PHAVer$.
La Section 2 présente les réseaux de Petri à chronomètres, sa syntaxe et sa sémantique. Une comparaison entre $SWPN$ et quelques extensions de TPN est présentée en Section 3. La Section 4 présente l'algorithme de traduction d'un $SWPN$ en un SWA et le calcul de l'espace des états de $SWPN$. L'analyse de l'automate à chronomètres est effectué en utilisant un model-checker $PHAVer$. La bisimulation entre le réseau de Petri à chronomètres initial et l'automate résultant est établi dans Section 5. Un exemple permettant d'illustrer notre travail est donné en Section 6.

2. Réseaux de Petri à chronomètres $SWPN$

Les réseaux de Petri à chronomètres $SWPN$ étendent les réseaux de Petri temporels TPN [BER 91] en incluant dans sa sémantique le comportement des systèmes interruptibles. Cela implique la suspension et la reprise de l'exécution des tâches, lors des interruptions. Le temps s'arrête pour les tâches interrompues, donc ce modèle s'appuie sur le concept de chronomètre, horloge pour laquelle le temps peut être arrêtée et redémarré. Dans un $SWPN$, il y a deux types de transitions : interruptibles et non-interruptibles. Un mécanisme d'initialisation des chronomètres appelé *Post-initialisation* est utilisé. Ce mécanisme repose sur le franchissement de la transition interruptible correspondante. Le franchissement d'une transition interruptible met à zéro le chronomètre associé à cette transition, tandis que le franchissement d'autre transition qui désensibilise la transition interruptible, suspend ce chronomètre. Il reprend lorsque la transition interruptible est sensibilisée de nouveau.
exemple : Considérons l'exemple d'une tâche interruptible ayant une durée d'exécution $[\alpha, \beta]$ (sans compter les interruption). L'interruption peut se produire n'importe quand interrompant une tâche. Celle-ci reprend après chaque interruption au même endroit. Le $SWPN$ de cette tâche est présenté dans la figure 1.a. Dans ce modèle, la place P_2 représente l'exécution de la tâche tandis que la place P_3 représente l'état d'interruption de la tâche. Les transitions t_3 et t_4 représentent respectivement l'occurrence de l'interruption et la reprise de la tâche. La transition t_2 étant une transition interruptible représente l'exécution de la tâche. Afin de distinguer la transition interruptible des autres, nous la dessinons en gras. Lorsqu'une interruption a lieu, le jeton bascule de P_2 à P_3 et x_2 devient inactif. Lorsque l'interruption se termine, la transition t_4 est franchie. La tâche reprend au même endroit et le chronomètre x_2 devient actif à nouveau et récupère sa valeur atteinte et sauvegardée lors de l'interruption. Le franchissement de t_2 met à zéro x_2. A l'état initial du $SWPN$, tous les chronomètres utilisés sont mis à zéro.

Definition 1 *(Syntaxe d'un $SWPN$) Un réseau de Petri à chronomètres est un 6-uplet $\langle P, T, {}^\bullet(.), (.)^\bullet, M_0, I_s \rangle$, où :*

– *P est un ensemble fini et non vide de places ;*

– T est un ensemble fini et non vide de transitions. L'ensemble $T = T_{int} \cup T_{no-int}$ est composé de deux sous-ensembles disjoints : les transitions interruptibles T_{int} et non-interruptibles T_{no-int} ;

– ${}^\bullet(.)$, $(.)^\bullet$ sont respectivement les fonctions d'incidence amont et aval ;

– $M_0 \in \mathbb{N}^{card|P|}$ est le marquage initial du réseau ;

– I_s est une fonction associant à chaque transition un intervalle donnant son instant de tir au plus tôt $EFT(t_i) \in \mathbb{R}^+$ et au plus tard $LFT(t_i) \in \mathbb{R}^+ \cup \{\infty\}$. □

2.1. *Sémantique*

Un marquage M du réseau est un élément de $\mathbb{N}^{card|P|}$ tel que $\forall p \in P$, $M(p)$ est le nombre de jetons dans la place p.
Une transition t est dite sensibilisée par le marquage M si $M \geq {}^\bullet t$. Nous notons $t_i \in enabled(M)$.
Nous noterons $\upsilon(t_i)$ la valeur d'un chronomètre associée à t_i : $\upsilon(t_i) \in (R^+)^{card|T|}$.
Lorsqu'une transition t_i est franchie, le marquage M' est déduit de celui de M en retirant des jetons de chaque place en amont of t_i, et en ajoutant des jetons à chaque place en aval de t_i : $M_{temp} = M - {}^\bullet(t_i)$ et $M' = M_{temp} + (t_i)^\bullet$. □
Les transitions qui sont sensibilisées par le marquage temporaire M_{temp} et M, sont dites persistantes sensibilisées. Les transitions non-interruptibles qui sont sensibilisées par M' mais ne sont pas sensibilisées par M_{temp}, sont dites nouvellement sensibilisées. Nous noterons une transition t_i nouvellement sensibilisée par le franchissement de la transition t_k à partir du marquage M, comme $\uparrow enabled(t_i, M, t_k)$. □
Une transition interruptible est dite premièrement sensibilisée par M' si : $\forall t_i \in T_{int}$, t_i n'est pas sensibilisée par M_{temp} et $\upsilon(t_i) = 0$ au M_{temp} mais t_i est sensibilisée par M'. Rappelons que $\upsilon(t_i) \leftarrow 0$ à chaque fois t_i est franchie. □
Une transition $t_i \in T_{int}$ est dite suspendue au marquage M, ce que nous noterons $susp(M)$, $t_i \in T_{int}$: $t_i \in susp(M)$ si ${}^\bullet(t_i) > M \wedge \upsilon(t_i) > 0$. □
La valeur d'un chronomètres associée à $t_i \in T$ est :

– $\forall t_i \in T_{int}$, $\upsilon(t_i)$ est le temps écoulé depuis l'instant de sa première sensibilisation, pendant lequel la transition reste active (sensibilisée).

– $\forall t_i \in T_{no-int}$, $\upsilon(t_i)$ est le temps écoulé depuis l'instant pour lequel la transition t_i a été nouvellement sensibilisée.

Une transition t_i est dite franchissable par le marquage M, ce que nous noterons $t_i \in franchissable(M)$, si $t_i \in enabled(M) \wedge EFT(t_i) \leq \upsilon(t_i) \leq LFT(t_i)$.
Pré-initialisation :Le mécanisme d'initialisation d'une transition non-interruptible $t_i \in T_{no-int}$ est dit Pré-initialisation car son chronomètre est mis à zéro lorsque t_i est nouvellement sensibilisée, c-à-d : avant l'utilisation de t_i. □
Post-initialisation : Le mécanisme d'initialisation d'une transition interruptible $t_i \in T_{int}$ est dit Post-initialisation car son chronomètre est mis à zéro par le franchissement de t_i, c-à-d : après l'utilisation de t_i. □

Nous définissons la sémantique des réseaux de Petri à chronomètres $SWPN$ sous la forme d'un système de transitions temporisé TTS.

Definition 2 *(Sémantique d'un SWPN) La sémantique d'un réseau de Petri à chronomètres $SWPN$ est définie sous la forme d'un système de transitions temporisé TTS $S_N = (Q, q_0, \rightarrow)$ tel que :*
• Q = $\mathbb{N}^p \times (\mathbb{R}^+)^{card|T|}$.
• q_0 = $(M_0, \overline{0})$. *A cet état, tous les chronomètres est mise à zéro* $\forall t_i \in T : \upsilon(t_i) := 0$.
• $\rightarrow \in Q \times (T \cup \mathbb{R}) \times Q$ *est la relation de transition incluant des transitions continues et des transitions discrètes :*
• *la relation de transition continue est définie* $\forall d \in \mathbb{R}^+$ *par :*

$$(M,\upsilon) \xrightarrow{d} (M, \upsilon') \textit{ ssi } \forall t_i \in T$$

$$\begin{cases} \upsilon'(t_i) = \begin{cases} \upsilon(t_i) + d \textit{ si } t_i \in enabled(M) \\ \upsilon(t_i) \textit{ sinon} \end{cases} \\ M \geq {}^\bullet(t_i) \Rightarrow \upsilon' \leq LFT(t_i) \end{cases}$$

• *la relation de transition discrète est définie* $\forall t_i \in T$ *par :*

$$(M,\upsilon) \xrightarrow{t_i} (M', \upsilon') \textit{ ssi } \forall t_i \in T$$

$$\begin{cases} t_i \in franchissable(M), \\ M' = M - {}^\bullet(t_i) + (t_i)^\bullet, \\ \upsilon'(t_i) := 0 \textit{ si } t_i \in T_{int}(Post - initialisation) \\ \forall t_k \in T, \upsilon'(t_k) = \begin{cases} \bullet\ 0 \textit{ si } t_k \in \uparrow enabled(t_k, M, t_i) \\ \bullet\ \upsilon(t_k) \textit{ si } t_k \in enabled(M) \\ \bullet\ \theta \textit{ si } t_k \in susp(M) \end{cases} \end{cases} \quad \square$$

Dans la relation de transition discrète, les *Post-* et *Pré-initialisation* sont présentées. Dans la première, le chronomètre est mis à zéro par le franchissement de la transition interruptible correspondante, tandis que dans la pré-initialisation le chronomètre est mis à zéro lorsque la transition non-interruptible est nouvellement sensibilisée.
Le cas où la transition t_k a été suspendue dans un état précédent M'' (qui précède M et M'), puis elle reste suspendue dans l'état M est aussi présenté dans la définition 2. Supposons que la valeur de t_k à l'instant de la suspension est θ. Donc, La valeur de t_k dans le marquage M' est celle de t_k à l'instant de la suspension : $\upsilon'(t_k) = \theta$.
Franchissement multiple et sensibilisation multiple :Le franchissement simultané de plusieurs transitions et la sensibilisation multiple d'une transition peuvent être envisagés. Mais pour avoir une présentation simple, nous nous limitons ici aux franchissements simples (un seul franchissement à la fois) et à une seule sensibilisation d'une transition.

3. Comparaison entre quelques extension des réseaux de Petri et $SWPN$

Dans ce qui suit, nous comparons le $SWPN$ présenté ici avec d'autres modèles réseaux de Petri, à savoir TPN, Scheduling-TPN, Preemptive-TPN et $IHTPN$.

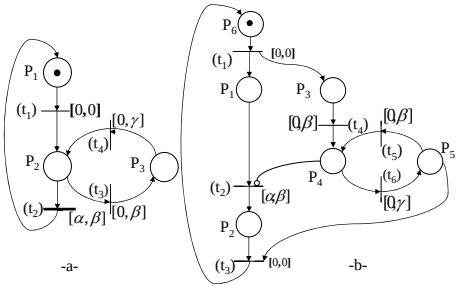

Figure 1. *a- SWPN d'une tâche interruptible b- Modélisation de la même tâche interruptible par $IHTPN$*

Relation entre réseaux de Petri temporels TPN ***et*** $SWPN$ ***:***
Un TPN peut être décrit par un $SWPN$. Un TPN est un $SWPN$ sans transitions interruptibles. Un TPN est strictement une sous-classe des $SWPN$.
Relation entre $IHTPN$ ***et*** $SWPN$ ***:***
Le modèle $IHTPN$ [ROU 04] étend TPN en contrôlant la progression des transitions. Le franchissement d'une transition peut être interrompue, s'il y a une place non vide connectée à cette transition par un arc inhibiteur.
La figure 1.b présente le modèle $IHTPN$ modélisant la tâche dans la figure 1.a. Dans cette figure, les transitions t_2 et t_4 seront sensibilisées par le franchissement de t_1. Le chronomètre associé à t_2 compte la durée effective de l'exécution. Ce chronomètre devient inactif lorsque P_4 est marquée. La sensibilisation de t_4 signifie que l'interruption peut arriver à importe quel instant pendant l'exécution. Lorsqu'une interruption se produit par le franchissement de t_4, t_6 sera validée et son chronomètre compte la durée d'une interruption. L'interruption et la reprise peuvent arriver plusieurs fois pendant l'exécution. Ceci est présenté par le franchissement successivement de t_5 et t_6. Lorsque la valeur $v(t_2)$ appartient à l'intervalle $[\alpha, \beta]$ et P_5 est marquée (la tâche n'est pas interrompue), les transitions t_2 et t_3 sont franchies et la place P_6 devient de nouveau marquée. Nous pouvons remarquer que notre extension $SWPN$ offre un formalisme graphique classique où la seule modification apportée concerne l'initialisation des horloges qui est un mécanisme simple à appréhender. Cela permet de représenter facilement les systèmes interruptibles.
Relation entre Scheduling-TPN ***ou Preemptive-***TPN ***et*** $SWPN$ ***:***
Les Scheduling-TPN [LIM 04] étendent les TPN en associant aux places deux nouveaux attributs, les ressources et priorités. Les preemptive-TPN [BUC 04] associent les mêmes attributs aux transitions au lieu des places. Les Scheduling-TPN et Preemptive-TPN sont particulièrement bien adaptés à la modélisation des systèmes préemptifs pour des objectifs d'ordonnancement. La façon d'ordonnancer des tâches réparties sur différents processeurs est prise en compte pour une priorité fixe. Un Scheduling-TPN ou Preemptive-TPN peut être modélisé par un $SWPN$. Supposons qu'il y a deux tâches différentes *tâche*$_1$ et *tâche*$_2$ représentées par les places P_1 et P_2. Ces deux tâches sont associées au même processeur $\gamma = 1$. La *tâche*$_1$ a

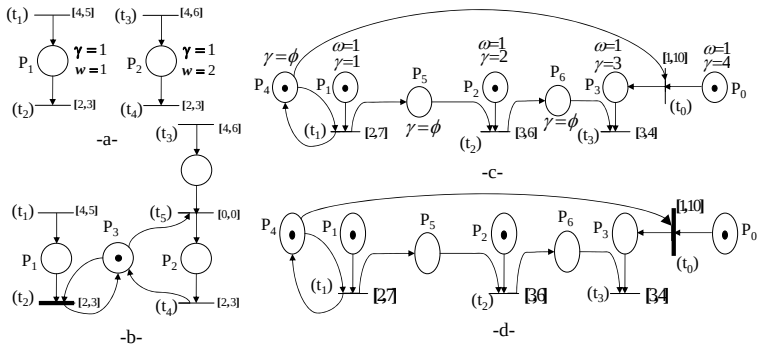

Figure 2. *Modélisation de deux tâches avec des priorités fixes différentes sur un processeur par : a- Scheduling-TPN b- $SWPN$. Modélisation d'une relation d-inhibition circulaire : c- Scheduling-TPN d- $SWPN$*

une priorité (w = 1) inférieure à celle de *tâche*$_2$ (w = 2) (Fig. 2.a). Lorsque les places P_1 et P_2 sont marquées, c'est le chronomètre associé à la transition t_4 qui est actif tandis que celui associé à la transition t_2 est suspendu. Le réseau $SWPN$ modélisant ces deux tâches est représenté sur Figure 2.b. Ce modèle représente la disponibilité du processeur par la place p_3. Le fait que la priorité des tâches sur le processeur est différente, est modélisé par des arcs entre p_3 et les transitions t_2, t_4 et t_5 où la transition t_2 est une transition interruptible.
Les Scheduling-TPN et Preemptive-TPN sont dédiés au problème d'ordonnancement des tâches pour une relation de priorité fixe, tandis que le problème de la suspension et la reprise d'une tâche est plus général. A titre d'un exemple, ces modèles ne sont pas capables de modéliser une relation circulaire de la priorité pouvant arrêter définitivement l'écoulement de temps des transitions, tandis que les ressources nécessaires associées aux transitions ou aux places sont disponibles. Dans la Figure 2.c, la fin de la tâche présentée par la place P_0 bloque les tâches présentées par P_1, P_2 et P_3 même si ses ressources sont disponibles. Mais, le franchissement de t_1 avant de t_0 met à zéro la valeur $v(t_0)$. Ceci signifie que l'état atteint de la tâche présentée par P_0 lors de la validation précédente de t_0 est perdu lorsque la tâche présentée par P_1 finit. Donc, le Scheduling-TPN n'est pas capable de modéliser cette relation de priorité tandis que le modèle $SWPN$ l'est. Dans la figure 2.d, la transition t_0 est interruptible. Par conséquent, les Scheduling-TPN et Preemptive-TPN sont strictement des sous-classes du $SWPN$.

4. Analyse temporelle de $SWPN$

Les problèmes de l'accessibilité d'un marquage ou d'un état, et le caractère borné (des classes d'états) sont indécidables pour les Scheduling-TPN [BER 05]. Il s'ensuit que ces problèmes sont également indécidables pour $SWPN$ car le Scheduling-

TPN est une sous-classe du $SWPN$. Pour l'analyse temporelle des $SWPN$, nous nous intéressons aux $SWPN$ bornés pour lesquels les bornes des intervalles temporels associés aux transitions sont des valeurs entières et le réseau de Petri associé est borné.
Pour l'analyse temporelle des $SWPN$, nous traduisons les réseaux de Petri à chronomètres bornés $SWPN$ en automate à chronomètres. Ces automates pourront être ensuite analysés à l'aide des techniques et des outils du SWA comme $PHAVer$ [FRE 05]. Cet outil permet l'analyse symbolique des automates hybrides et est basé sur la manipulation symbolique de régions décrites par des polyèdres. $PHAVer$ fournit un certain nombre d'opérations pour manipuler les régions, incluant le calcul des états accessibles depuis une région, le calcul des successeurs et l'enveloppe convexe.

4.1. *Traduction des* $SWPN$ *vers les stopwatch automata* SWA

Dans cette section, nous montrons comment traduire un $SWPN$ borné en un automate à chronomètres SWA. Le calcul de l'automate SWA se fait en deux étapes. Dans la première, nous allons calculer le graphe des marquages du $SWPN$. A partir de ce graphe, nous déduirons syntaxiquement l'automate à chronomètres du $SWPN$. Le sommet initial est défini par le marquage initial et les chronomètres sont associés aux transitions. Un chronomètre x_i associé à la transition t_i correspond à la valeur $v(t_i)$ présentée dans la section 2. Un algorithme d'analyse en avant sera ensuite appliqué sur cet automate. Cet algorithme commence de l'état initial et explore toutes les évolutions possibles du $SWPN$ soit par le franchissement des transitions ou en laissant le temps s'écouler. Cette méthode est une adaptation du graphe des régions dans l'automate temporisé [ALU 94].

4.1.1. *Automate à chronomètres*

Les automates à chronomètres sont définis comme une sous-classe des automates hybrides linéaires pour laquelle les dérivées des variables par rapport au temps dans un sommet sont soit 0 ou 1 [CAS 00].

Definition 3 *Un automate à chronomètres est un 7-tuple* $(L, l_0, X, \Sigma, A, I, Dif)$ *où :*

– L *est un ensemble fini de sommets.* l_0 *est le sommet initial,*

– X *est un ensemble fini de chronomètres à valeurs réelles positives,*

– Σ *est un ensemble fini d'actions,*

– $A \subset L \times C(X) \times \sigma \times 2^X \times L$ *est un ensemble fini des arcs. Soit* $a = (l, \delta, t, R, l') \in A$ *est un arc reliant le sommet* l *au* l'*, avec la garde* δ*, l'action* t *et l'ensemble de chronomètres à remettre à zéro* R*.* $C(X)$ *est l'ensemble des contraintes sur* X*.*

– $I \in C(X)^L$ *associe un invariant à chaque sommet,*

– $Dif \in (\{0,1\}^X)^L$ *associe à chaque sommet la dynamique des chronomètres dans ce sommet,* $\dot{X}$ *désigne l'ensemble des dérivées par rapport au temps des va-*

riables de X *.*$\dot{X} = Dif(l)(x)_{x \in X}$. □

4.1.2. *Techniques d'analyse d'atteignabilité de* SWA

Un état d'un SWA est un couple (L, E), où L est un sommet du SWA et E est un polyèdre représentant l'espace temporel dans L. Il y a deux types d'évolution d'un automate à partir d'un état (L, E). L'automate peut rester dans le même sommet en laissant le temps s'écouler, ou il y peut avoir franchissement d'une transition discrète. Par conséquent, un état d'un automate peut avoir deux types de successeurs : continus $succ_t$ et discrets $succ_d$. On peut trouver les détails de ces techniques dans [ALU 95].

4.2. *étiqueter le graphe des marquages :*

Supposons que G est le graphe des marquages du $SWPN$. Nous rappelons que le graphe des marquages est défini par le couple $G = (M, A)$, où :

– M est l'ensemble des marquages possibles du $SWPN$: M_0,...M_p. Chaque marquage caractérise un état de validation ou de suspension des transitions ;

– A est l'ensemble des arcs entre les noeuds du graphe des marquages : a_0,....a_q.

L'automate à chronomètres sera obtenu en associant à chaque marquage des équations différentielles qui expriment la dynamique des chronomètres dans chaque marquage, et un invariant. Chaque arc est associé par une garde et des affectations. Comme nous l'avons supposé le nombre de franchissements simultanés est limité à un.
Nous indiquons l'automate à chronomètres obtenu à partir du graphe des marquages G par SWA_G. Dans cet automate, nous indiquons aussi chaque sommet par son marquage, c-à-d : L_0....L_k correspondent à M_0,....M_k.
Dynamique des chronomètres : Un ensemble d'équations différentielles sous la forme $\dot{x} = c$ où $c = \{0, 1\}$ est associé à chaque marquage M_k.
$\forall t_i \in enabled(M_k) : \dot{x_i} = 1$ and $\forall t_j \in susp(M_k) : \dot{x_j} = 0$. Les transitions inactives qui ne participent pas à l'évolution de l'automate, ne sont pas considérées. (t_m est inactive si $(t_m \in T_{int} \wedge t_m \notin enabled(M_k) \wedge v(t_m) = x_m = 0)$ ou $(t_m \in T_{no-int} \wedge t_m \notin enabled(M_k))$.
Invariant : Un invariant est associé à chaque marquage M_k. Supposons que X_k est l'ensemble des chronomètres associés à des transitions actives ou suspendues dans le marquage M_k d'un $SWPN$. L'invariant associé à M_k est défini par : $I(M_k) = \{x_i \leq LFT(t_i) \mid t_i \in enabled(M_k)$ or $susp(M_k)\}$
Garde : Chaque arc a_k du graphe G correspond au franchissement d'une transition t_i du $SWPN$. Donc, nous étiquetons a_k par :

– Le nom d'action : t_i ;

– La garde : $EFT(t_i) \leq x_i \leq LFT(t_i)$

– $\forall t_k \in \uparrow enabled(t_k, M, t_i)$ où M et le marquage d'état d'origin : nous ajoutons l'affectation $x_k \leftarrow 0$. Si $t_i \in T_{int}$, nous ajoutons aussi l'affectation $x_i \leftarrow 0$.

4.3. *Une itération de l'algorithme d'analyse en avant de* SWA_G *:*

L'algorithme proposé est basé sur l'analyse en avant pour trouver les états atteignables de SWA_G. Ces états atteignables sont accumulés dans une pile $Reach$. A l'état initial, $Reach = L_0$. Par la suite, nous présentons une itération de l'algorithme que nous proposons pour calculer des états atteignables à partir de (L_0, E_0^e) où le polyèdre E_0^e contient les valeurs des chronomètres à l'entrée de L_0.

– Calculer l'évolution temporelle possible des chronomètres actifs dans L_0. Autrement dit : les valeurs possibles des chronomètres pendant le séjour de l'automate dans L_0. Ces valeurs sont contenues dans le successeur continu de E_0^e. $E_0 = Succ_t(E_0^e)$

– Déterminer les arcs franchissables depuis L_0. Supposons que $a_{0,k}$ est un arc de L_0 à L_k. $a_{0,k}$ auquel est associé l'action t_i. Cette transition est franchissable si : $E_0 \cap \{EFT(t_i) \leq x_i \leq EFT(t_i)\}$ est un polyèdre non vide. Ajouter à l'ensemble $Reach$ le sommet L_k : $Reach = \{L_0, L_k\}$.

– Pour chaque arc franchissable $a_{0,k}$, déterminer les valeurs des chronomètres à l'entrée de L_k :

$$S_{0,k} = Succ_d(E_0)$$
$$E_k^e = S_{0,k} \wedge [X_e := 0]$$

où X_e est l'ensemble des chronomètres qui sont mis à zéro par le franchissement de $a_{0,k}$. E_k^e est le polyèdre pour lequel le sommet L_k est atteignable.

– Calculer les évolutions temporelles possibles pendant le séjour dans L_k :

$$E_k = Succ_t(E_k^e)$$

Nous avons proposé ci-dessus un semi-algorithme afin d'analyser temporellement un $SWPN$. Si cet algorithme d'analyse ne converge pas, nous proposons d'utiliser les techniques du logiciel de vérification formelle $PHAVer$ afin de forcer la terminaison. Si les invariants de l'automate sont bornés, la terminaison se fait en utilisation des techniques de simplifications d'un polyèdre complexe (par exemple, limiter le nombre de bits utilisé pour représenter les contraintes d'un polyèdre et limiter le nombre des contraintes décrivant un polyèdre convexe [FRE 05]). En conséquence, il n'y a qu'un nombre fini de contraintes possibles pour définir un polyèdre quelconque, et donc il n'y a qu'un nombre fini des résultats pour l'algorithme de l'atteignabilité. Il s'ensuit qu'on calcule une surapproximation conservative de l'espace d'état. En effet, la surapproximation n'est pas gênante pour la vérification de propriété de sûreté. Puisque nous voulons assurer que les mauvais états ne sont jamais atteints. D'un autre côté, puisqu'il s'agit d'une surapproximation, cela présente le risque de permettre le franchissement de transitions qui ne le sont normalement pas. Dans la section suivante, nous allons montrer formellement que cet automate résultant est toujours temporellement bisimilaire au $SWPN$ initial. Les états supplémentaires ajoutés par la surapproximation ne sont pas accessibles puisque les gardes et invariants sont calculés de manière syntaxique. Cela est illustré par le réseau de la figure 3.a et la partie de son automate présenté dans la figure 3.b. L'espace temporel atteignable obtenu en L_5 par $PHAVer$ (avec l'utilisation des opérateurs de forçage de la terminaison) est :

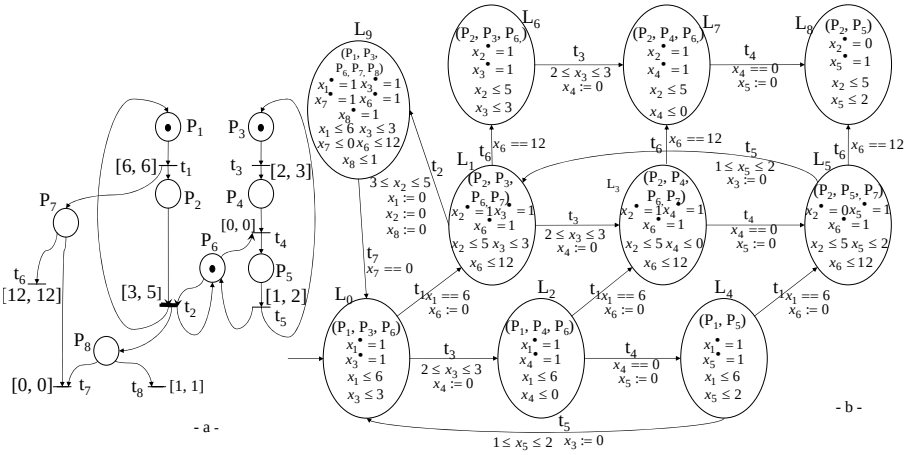

Figure 3. *a- Un réseau de Petri à chronomètres b- Sa traduction partielle en SWA*

$-7x_2 - 5x_5 + 5x_6 \geq -10 \wedge -x_2 + x_6 \geq 0 \wedge 0 \leq x_2 \leq 5 \wedge 0 \leq x_5 \leq 2 \wedge x_6 \leq 12$ $\wedge\ 2x_2 + x_5 - x_6 \geq -1$. Dans cet space, la garde $x_2 = 12$ vérifie cette space et le sommet L_8 est atteignable.

Nous pouvons constater que dans les sommets L_1, L_3 et L_5 la transition t_6 n'est effectivement pas franchissable, car la transition t_7 sera validée au plus tard après le franchissement de t_1 pour une valeur maximale de x_6 est 11. Cette valeur de x_2 correspond à la séquence suivante de franchissement à partir de L_3. Dans cette séquence, un état atteignable après le franchissement d'une transition est indiqué par $(L_i, (x_2, x_4, x_5, x_6))$ où les valeurs des chronomètres sont celles à l'instant d'atteindre le sommet L_i. La séquence est : $(L_3, (0, , 0, 2, 0)) \xrightarrow{t_4} (L_5, (0, 0, 0, 0)) \xrightarrow{t_5}$ $(L_1, (0, 0, 2, 2)) \xrightarrow{t_3} (L_3, (2, 0, 2, 4)) \xrightarrow{t_4} (L_5, (2, 0, 0, 4)) \xrightarrow{t_5} (L_1, (2, 0, 2, 6)) \xrightarrow{t_3}$ $(L_3, (4, 0, 2, 8)) \xrightarrow{t_4} (L_5, (4, 0, 0, 8)) \xrightarrow{t_5} (L_1, (4, 0, 2, 10)) \xrightarrow{t_2} (L_9, (0, 0, 2, 11))$.

Nous voyons que les sommets L_7, L_8 et L_9 qui correspondent au franchissement de la transition t_6 ne sont en fait pas accessibles. Ceci est dû au fait que la valeur de x_2 dans les sommets L_1, L_3 et L_5 est toujours inférieure ou égale à 11. Donc, la garde $x_6 = 12$ ne sera jamais vérifiée. Pour s'en convaincre complètement, prenons la séquence de l'automate à laquelle nous nous étions intéressés. Supposons que l'automate est à l'entrée du sommet L_5 avec les valeurs : $x_2 = 4$, $x_4 = 0$, $x_5 = 0$ et $x_6 = 8$. Nous pouvons laisser le temps s'écouler τ et il est clair que $\forall \tau \in [1, 2]$: $x_5 + \tau \in [1, 2]$ et $max\{x_6 + \tau\} < 12$. Donc la garde $x_6 = 12$ ne sera jamais vérifiée.

L'automate à chronomètres SWA_{Reach} obtenu en appliquant l'algorithme d'analyse en avant sur SWA_G est défini ci-dessous :

Definition 4 – $L = Reach = \{L_0, ..., L_k\}$*, est l'ensemble des marquages atteignable du SWPN.* $L_0 = M_0$

– *X, est l'ensemble de tous les chronomètres associés aux transitions du $SWPN$,*
– *$\Sigma = \{t_1, ..., t_q\}$, est l'ensemble des actions, c-à-d : les transitions de $SWPN$,*
– *A, est un ensemble fini des arcs franchissables entre les sommets atteignables,*
– *$I : L \rightarrow C(X)$.*

5. Bisimulation entre $SWPN$ et SWA_{Reach}

Definition 5 *Soient N un $SWPN$ et A son automate à chronomètres présenté dans la Définition 4. Soient Q_N l'ensemble des états de N et Q_A l'ensemble des états de A. Soit $R \subset Q_N \times Q_A$ une relation entre un état de N et un état de l'automate à chronomètres A tel que :*

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

où $\mathbb{M}$ est la fonction donnant le marquage associé à un état l de A. □

Théorème 1 *$\forall\ (M, \upsilon_N)$, (l, υ_A) tel que $(M, \upsilon_N) R (l, \upsilon_A)$:*

$$\bullet (M, \upsilon_N) \xrightarrow{t_i} (M', \upsilon'_N) \Leftrightarrow \begin{cases} (l, \upsilon_A) \xrightarrow{t_i} (l', \upsilon'_A) \\ (M', \upsilon'_N) R (l', \upsilon'_A) \end{cases}$$

$$\bullet (M, \upsilon_N) \xrightarrow{d} (M, \upsilon'_N) \Leftrightarrow \begin{cases} (l, \upsilon_A) \xrightarrow{d} (l, \upsilon'_A) \\ (M, \upsilon'_N) R (l, \upsilon'_A) \end{cases}$$

R est une relation de bisimulation. □

Preuve 1 *Soient $s_N = (M, \upsilon_N) \in Q_N$, $s_A = (l, \upsilon_A) \in Q_A$, $(M, \upsilon_N) R (l, \upsilon_A)$.*
• Transition continue – écoulement de temps.
- Supposons que le $SWPN$ N puisse laisser le temps $d \in \mathbb{R}^+$ s'écouler : $s_N \xrightarrow{d} s'_N$ où $s'_N = (M, \upsilon'_N)$ et $\upsilon'_N(t_j) = \upsilon_N(t_j) + d$. Cela signifie que $\forall t_j \in enabled(M) \Rightarrow \upsilon_N(t_j) + d \leq LFT(t_j)$. Par construction, l'invariant du sommet l est la conjunction de temps au plus tard de franchissement des transitions sensibilisés et interrompues $I(l) = \vee\{x_j \leq LFT(t_j)\}$: $\forall t_j \in enabled(M)$ or $susp(M)$. Donc, $\forall \tau \in [0, d]$: $I(t_j)(\upsilon_A(t_j) + \tau) = vrai$. Puisque le $SWPN$ N reste dans le même marquage, et le SWA A dans le même sommet, l'activité des transitions et les conditions sur $\dot{x}$ ne change pas. Par ailleurs, on a $(M, \upsilon_N)\ R\ (l, \upsilon_A)$. Par conséquent, $(M, \upsilon_N + d) = (l, \upsilon_A + d) \Longrightarrow (M, \upsilon_N + d)\ R\ (l, \upsilon_A + d)$.
- Symétriquement, supposons que le SWA A puisse laisser le temps $\tau \in [0, d]$ s'écouler : $s_A \xrightarrow{\tau} s'_A$ où $s'_A = (l, \upsilon'_A)$ et $\upsilon'_A = \upsilon_A + \tau$. Cela signifie que $I(t_j)(\upsilon_A + \tau) \leq LFT(t_j)$ est vrai. Selon la sémantique du $SWPN$, une transition continue peut se

produire ssi $\forall t_j \in enabled(M) : \upsilon_N + d \leq LFT(t_j)$. *Ce qui signifie que* N *peut laisser le temps* d *s'écouler. Puisque, on a* $(M, \upsilon_N)\ R\ (l, \upsilon_A)$, *alors* $(M, \upsilon_N + d)\ R\ (l, \upsilon_A + d)$ *pour* $\tau = d$.

En conséquent, R *est une relation de bisimilation pour les transitions continues.*

• *Transitions discrètes – franchissement d'une transition.*

- Supposons que le $SWPN\ N$ *puisse franchir la transition* $t_j \in T : s_N \xrightarrow{t_j} s'_N$. *Nous allons montrer que l'arc correspondant du* $SWA\ a$ *est aussi franchissable. Une transition* t_j *peut être franchie si* $t_j \in enabled(M) \wedge EFT(t_j) \leq \upsilon_N(t_j) \leq LFT(t_j)$. *Le nouveau marquage est* $M' = M - {}^\bullet(t_j) + (t_j)^\bullet$ *et les nouvelles valeurs des chronomètres sont :* $\upsilon'_N(t_k) = 0$ *si* $t_k \in \uparrow enabled(t_k, M, t_j)$, $\upsilon'_N(t_j) = 0$ *si* $t_j \in T_{int}$, $\upsilon'_N(t_m) = \upsilon_N(t_m)$ *pour les transitions persistantes actives en* M'. *Si la transition* t_m *est suspendue au marquage* M'' *(qui précède* M *et* M'*) où la valeur du chronomètres associé à* t_m *à l'instant de l'interruption est* $\upsilon''_N(t_m) = \theta$ *et* t_m *reste suspendue au* M, *donc* $\upsilon'_N(t_m) = \theta$. *Il y a un arc du* $A\ a = (l, g, t_j, R, l')$ *where* $\mathbb{M}(l)$=M *et* $\mathbb{M}(l')$=M'. *Par construction, la garde est :* $EFT(t_j) \leq x_j \leq LFT(t_j)$. *Ainsi, lorsque* t_j *est franchissable :* $g(\upsilon_A) = vrai$ *et le* $SWA\ A$ *peut prendre l'arc* a. *Par construction, les chronomètres qui sont mis à zéro par le franchissement de* a *sont les mêmes chronomètres affectés par le franchissement de* t_j *en* N. *La dynamique des chronomètres au sommet* l' *est également définie en accord avec les activités des transition au marquage* M'. $x_k = 1$ *si la transition* t_k *est nouvellement validée ou persistante active de* M. *Or,* $x_k = 0$ *si* t_k *est désactivée par le franchissement de* t_j *ou elle persiste inactive de* M. *D'ailleurs,* $I(l')(\upsilon'_A) = vrai$ *car les chronomètres actifs et suspendus sont représentés dans l'invariant du* l' *par construction. Nous constatons que l'arc* a *en l'automate* A *est possible et* $(M', \upsilon'_N)\ R\ (l', \upsilon'_A)$.

- Symétriquement, supposons que le $SWA\ A$ *puisse franchir l'arc* $a : s_A \xrightarrow{t_j} s'_A$. *Nous allons prouver que la transition* t_j *de* N *est aussi franchissable.*

Tant que $a = (l, g, t_j, R, l')$ *de* SWA_{Reach} *est franchissable, cela signifie que la transition* t_j *est sensibilisée par le marquage* M *correspondant au sommet* l *et que* $g(\upsilon_a)$ *est vraie. Donc, par definition de g,* $EFT(t_j) \leq x_j \leq LFT(t_j)$ *et donc* $EFT(t_j) \leq \upsilon_N(t_j) \leq LFT(t_j)$, *ce qui signifie que* t_j *est franchissable pour le* $SWPN\ N$. *Comme la précédent point,* $(M', \upsilon'_N)\ R\ (l', \upsilon'_A)$ *par construction.*

• *Nous allons prouver que les états du* SWA_{reach} *qui sont ajoutés par la surapproximation, ne sont pas atteignables en le SWPN* N.

Dans SWA_{Reach}, *supposons qu'il existe dans le sommet* (l) *un arc sortant* $a = (l, g, t_j, R, l')$ *car* t_j *est franchissable dans l'espace surapproximé du* (l) *alors qu'il ne l'est pas dans l'espace exacte correspondante. Si nous supposons qu'avant d'atteindre le sommet* (l), *le comportement de l'automate était correct (exact), alors à l'instant précis de l'entrée dans le sommet* (l), *l'automate est dans un état* $a = (l, \upsilon_A)$ *qui est en relation avec un état* (M, υ_N) *du SWPN par R. D'une part, puisque* t_j *n'est pas en fait franchissable, cela signifie qu'il existe une autre transition* t_i *qui doit être franchie avant elle :* $EFT(t_j) - \upsilon_N(t_j) > LFT(t_i) - \upsilon_N(t_i)$. *Donc, par definition de R,* $EFT(t_j) - \upsilon_A(t_j) > LFT(t_i) - \upsilon_N(t_i)$. *Puisque, par definition d'un SWPN,* $LFT(t_i) \geq \upsilon_N(t_i)$, *cela nous donne* $EFT(t_j) - \upsilon_A(t_j) > 0$. *D'autre part, l'arc* a *est franchissable si* $EFT(t_j) - \upsilon_A(t_j) \leq 0$. *Avec ce qui précède, nous pouvons conclure*

que la garde est fausse et l' n'est pas donc accessible.
Cela termine la preuve que R est une relation de bisimulation. □

Nous voyons que le calcul à l'aide des techniques existantes en $PHAVer$ pour converger le calcul, peut ajouter des comportements au SWA_{Reach}. Cependant, ces comportements sont éliminés par l'ajout syntaxique des gardes et invariants pour obtenir l'automate. Le comportement de SWA_{Reach} est donc exactement le même (au sense de la bisimulation temporelle) que celui du $SWPN$ initial.

6. exemple illustratif

La Figure 4.a présente un réseau de Petri à chronomètres $SWPN$ modélisant un processus de production de deux types de produits S_1 et S_2. Les différentes tâches du processus s'exécutent sur les machines M_a, M_b et un robot de manipulation représente une ressource partagée. Les Tâches 1 et Tâche 2 sur S_1 s'exécutent respectivement sur la machines M_a et le robot. Les durées d'exécution sont respectivement de $[3, 3]$ $u.t$ pour Tâche 1 et $[1, 3]$ $u.t$ pour Tâche 2. La Tâche 3 s'exécutant sur la machine M_b pour S_2 est une tâche interruptible. Sa durée d'exécution (sans compter les interruptions) est de $[10, 10]$ $u.t$. L'interruption peut arriver après le début de la tâche dans l'intervalle $[3, 4]$ $u.t$. La durée d'une interruption est de $[1, 2]$ $u.t$. La durée entre la reprise de la tâche et l'interruption suivante est $[3, 4]$ $u.t$. La Tâche 4 s'exécutant sur S_2 par le robot a une durée d'exécution $[4, 5]$ $u.t$. La Tâche 2 a une priorité supérieure de la Tâche 4 par rapport au robot. Lorsque le robot exécute Tâche 4 pour S_2 et la machine M_a termine Tâche 1 pour S_1, le robot exécute immédiatement Tâche 2 pour

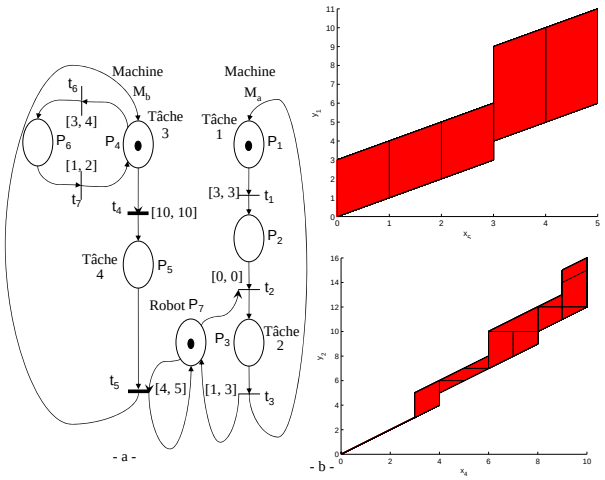

Figure 4. *a- $SWPN$ modélisant le processus de production b- les projection de l'espace temporel atteignable sur les axes (x_2, y_2) et (x_5, y_1)*

S_1. Lorsqu'il termine, Tâche 4 reprend au même endroit où elle a été interrompue.
La figure 5 montre l'automate à chronomètres correspondant au modèle $SWPN$ où les sommets instables correspondants à la transition t_2 sont supprimés. Cet automate peut être exploité de plusieurs manières, on peut par exemple vérifier des propriétés de performance du système de production. Il nous a paru intéressant de calculer les durées maximales des Tâche 3 et Tâche 4. Pour cette raison, nous introduisons dans l'automate les variables y_1 et y_2. La variable y_1 est activée par le franchissement de t_4. Elle est désactivée par le franchissement de t_5. y_1 calcule la durée d'exécution de Tâche 4. La variable y_2 est active dans les sommets qui suivent l'arc libellé par t_5 et désactivée dans les sommets qui suivent l'arc libellé par t_4. Elle mesure la durée d'exécution de Tâche 3. L'algorithme d'analyse de l'automate de la figure 5 converge sans utiliser les paramètres qui forcent la convergence (l'espace calculé par $PHAVer$ est exact). Les projections de l'espace atteignable sur les axes (x_2, y_2) et (x_5, y_1) sont présentées dans la figure 4.b. Nous trouvons de l'espace atteignable de (x_5, y_1) que la durée maximum afin que Tâche 4 puisse être exécutée est $y_1 = 11$. Nous trouvons aussi de l'espace atteignable de (x_2, y_2) que la machine M_b prend une durée $12 \leq y_2 \leq 16$ afin d'exécuter sa tâche.
On a pu ainsi mettre en évidence un comportement du système que l'on peut difficilement déterminer directement à partir du réseau de Petri à chronomètres.

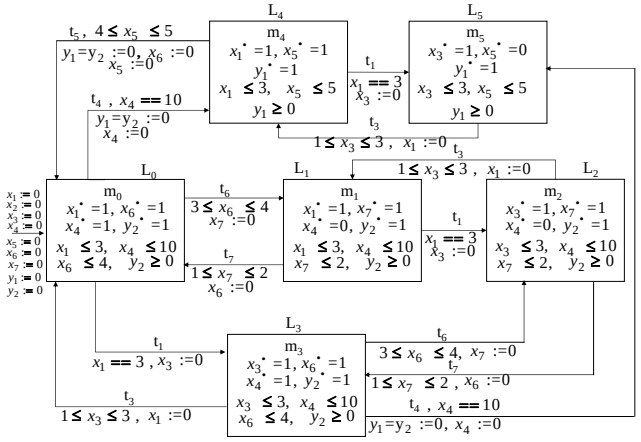

Figure 5. *L'automate à chronomètres du réseau de la figure* 6

7. Conclusion

Nous avons proposé un formalisme permettant de modéliser une sous-classe des systèmes temps-réel appelé les systèmes interruptibles. L'extension proposée conserve les propriétés fondamentales d'un réseau de Petri tout en apportant des possibilités nouvelles et intéressantes de modélisation de l'interruption et la reprise d'une activité. Ce nouveau modèle étend le concept d'horloge au concept de chronomètre. Un

nouveau mécanisme d'initialisation des chronomètres appelé "Post-initialisation" est introduit. Nous avons également présenté une méthode pour l'analyse efficace des réseaux de Petri à chronomètres $SWPN$. Elle consiste dans un premier temps à traduire le $SWPN$ en automate à chronomètres. Dans un second temps, nous appliquons un algorithme sur l'automate ainsi obtenu, basé sur les techniques d'analyse en avant connues dans les automates hybrides et en utilisant le logiciel de vérification formelle $PHAVer$. Cette analyse permet de calculer l'espace d'états temporel de l'automate à chronomètres qui est bisimilaire au réseau de Petri à chronomètres initial. Des calculs d'indices de performances peuvent être effectués sur l'automate et interprétés sur le modèle réseau de Petri. Enfin, nous souhaitons étudier la possibilité de diminuer le nombre de chronomètres dans le SWA_G car le coût de la vérification d'un SWA en utilisant $PHAVer$ est exponentiel par rapport au nombre de chronomètres.

8. Bibliographie

[ALU 94] ALUR R., DILL D., « A theory of timed automata », *Theoretical computer Science*, , n° 126, 1994, p. 183–235.

[ALU 95] ALUR R., COURCOUBETIS C., HALBWACHS N., HENZINGERD T., HOD P., NICOLLIN X., OLIVERO A., SIFAKIS J., YOVINE S., « The algoritmic analysis of Hybrid systems », *Theoretical Computer Science*, vol. 138, n° 1, 1995.

[BER 91] BERTHOMIEU B., DIAZ M., « Modeling and Verification of Time Dependent Systems Using Time Petri Nets », *IEEE Transaction Software Engineering*, vol. 17, n° 3, 1991.

[BER 05] BERTHOMIEU B., LIME D., ROUX O. H., VERNADAT, « Problèmes d'accessibilité et espaces d'états abstraits des réseaux de Petri temporels à chrnomètres », *Journal européen des systèmes automatisés*, vol. 39, 2005, p. 223–238.

[BOB 00] BOBBIO A., PULIAFITO A., TELEK M., « A Modeling Framework to Implement Preemption Policies in Non-Markovian SPNs », *IEEE Transactions on Software Engineering*, vol. 26, n° 1, 2000, p. 36–54.

[BUC 04] BUCCI G., FEDELI A., SASSOLI L., VICARIO E., « Timed State Space Analysis of Real-Time Preemptive Systems », *IEEE Transactions on Software engineering*, vol. 30, n° 2, 2004, p. 97–111.

[CAS 00] CASSEZ F., LARSEN K., « The impressive power of stopwatch », *11th conference on concurrency theory*, 2000, p. 138–152.

[FRE 05] FREHSE G., « PHAVer : Algorithmic Verification of Hybrid Systems past HyTech », *The Fifth International Workshop on Hybrid Systems : Computation and Control*, 2005, p. 258–273.

[LIM 04] LIME D., ROUX O. H., « A Translation Based Method for the Timed Analysis of Scheduling Extended Time Petri Nets », *25th IEEE International Real time Systems Symposuim*, Lisbon, Portugal, 2004, p. 187–196.

[ROU 04] ROUX O. H., LIME D., « Time Petri Nets with Inhibitor Hyperarcs. Formal Semantics and State Space Computation », *25th International Conference on theory and application of Petri nets*, Bologna, Italy, 2004, p. 371–390.