Du réseau de Petri temporel étendu vers les automates hybrides linéaires pour l'analyse des systèmes

Yamen Etouati, Moez Yeddes, Nejib Hadj Alouane, Hassane Alla

Du réseau de Petri temporel étendu vers les automates hybrides linéaires pour l'analyse des systèmes

Yamen EL TOUATI[1], Moez YEDDES[2] , Nejib BEN HADJ ALOUANE[1], Hassane ALLA[3]

[1]Laboratoire OASIS
Ecole Nationale des Ingenieurs de Tunis, BP 37, Le Belvedère 1002 Tunis, Tunisie

[2]Laboratoire CRISTAL
Ecole Nationale des Sciences de l'Informatique, Campus de La Manouba 2100, Tunisie

[3]GIPSA-Lab
Domaine Universitaire, BP46, F-38402 Saint Martin d'Hères cedex Grenoble, France

yamen.eltouati@yahoo.fr, yeddes@yahoo.fr, nejib_bha@yahoo.com, Hassane.ALLA@lag.ensieg.inpg.fr

Ce travail de recherche a été réalisé avec le soutien financier du Comité Mixte pour la Coopération Universitaire Franco-Tunisienne

Résumé— **Nous avons proposés dans [1] de modéliser une catégorie de systèmes hybrides à aspect cumulatif à travers le modèle réseau de Petri temporel étendu (RdP-TE). Ce dernier, est une application des réseaux de Petri temporels (TPN) auxquels nous avons étendu le concept d'horloge vers des dynamiques plus complexes. Ceci a permis de modéliser une catégorie de systèmes plus large que celle couverte par les TPN. Dans cet article, nous proposons une méthode systématique de conversion des RdP-TE vers les automates hybrides linéaires (AHL) à travers un algorithme de transformation. L'AHL obtenu reflète le comportement temporel du système modélisé par le RdP-TE. Cette transformation est proposée dans un but d'analyse du comportement et des propriétés du système.**

Mots-clés— **Réseau de Petri temporel, système dynamique hybride, modélisation, supervision, automate hybride linéaire.**

I. Introduction

Dans le cadre de notre étude, nous nous intéressons à une catégorie des systèmes dynamiques hybrides ayant un aspect cumulatif et dont la composante discréte est celle qui pilote le systéme. Nous avons proposé dans [1] de modélisér ces systèmes à l'aide des réseaux de petri temporels étendu (RdP-TE). Le RdP-TE, basé sur le réseau de Petri temporel (TPN)[2], permet de modéliser certains phénomènes telque la préemption et la reprise des actions, ainsi que la dépendance des actions séquentielles. Le RdP-TE permet de modéliser, en plus de l'écoulement du temps, des variables continues par morceaux régies par des équations de type $\dot{x} = k$. Ainsi, nous pouvons modéliser une succession de traitements qui, éventuellement, peuvent être dépendants.

Dans le but d'appliquer des outils et théories existantes sur l'analyse des systèmes hybrides, nous proposons un algorithme permettant la transformation d'un RdP-TE en un automate hybride linéaire (AHL). On pourra par la suite utiliser les automates hybrides, obtenus suite à l'application de l'algorithme de transformation, pour la supervision et la synthése de la commande. Nous pouvons ainsi, utiliser certains logiciels [3], [4], [5] pour l'analyse des systémes modélisés par les RdP-TE.

L'article est organisé de la façon suivante : dans la section 2, nous présentons formellement les modèles necessaires à la transformation : l'AHL et le RdP-TE. Cette section s'entame par un exemple illustratif d'un systéme de chauffe modélisé par un RdP-TE. Dans la section 3, nous construisons l'algorithme de transformation des RdP-TE vers les automates hybrides linéaires. Certaines analyses, à l'aide de l'outil PHAVer de l'exemple du système de chauffe, sont donnés à la section 5.

II. Présentation des modèles automate hybride et réseau de Petri temporel étendu

Dans cette section, nous introduisons le modèle automate hybride et réseau de Petri temporel étendu. Ceci est nécessaire pour illustrer par la suite le principe de la transformation des RdP-TE vers les AHL.

Définition 1: [6], [7] un automate hybride linéaire à dérivée constante est défini par
$\mathcal{A} = (X,\ S,\ inv,\ dyn,\ E,\ garde,\ aff)$ où :

- $\mathcal{X}$ est un ensemble fini de variables.
- $\mathcal{L}$ est un ensemble fini de sommets.
- inv est la fonction qui associe à chaque sommet $s \in S$ un prédicat sur les variables appelé l'invariant de sommet.
- $dyn : S \times X \longrightarrow \mathbb{Q}$ est la fonction qui décrit l'évolution des variables dans les sommets. l'évolution est toujours du type $\dot{x} = k,\ k \in \mathbb{Q}$.
- $\mathcal{T}$ est un ensemble fini de transitions. Chaque transition $e = (s, s') \in \mathcal{T}$ identifie un sommet de départ $s \in S$ et un sommet d'arrivée $s' \in S$
- $garde$ est une fonction qui associe à chaque transition

$e = (s, s')$ un prédicat C_e appelé garde. une transition $e = (s, s')$ ne peut être exécutée que si la garde C_e est vérifiée.

– aff est une fonction qui associe à chaque transition $e = (s, s')$ une relation aff_e appelée affectation. Elle est utilisée pour la mise à jours des valeurs des variables aprés l'exécution de la transition.

□

Dans un AHL l'évolution des variables est toujours du type $\dot{x} = k$.

A. Le modéle RdP-TE

Avant de définir un RdP-TE, nous commençons par la notion de tâche. Pour plus de détail, le lecteur peut se réferer à [1]. Une tâche est un réseau de Petri particulier délimité par une transition source et une transition puits, représentant le début et la fin de la tâche. Une tâche permet de modéliser l'évolution d'une variable continue par morceau. Une instance de la variable est créée lors de la sensibilisation de la transition de début de tâche. La valeur est conservée à travers le(s) jeton(s) causant la validation. Dans ce sens, l'étendu de la variable est considéré global par rapport à la tâche. Cette notion sera utilisée pour définir le modéle RdP-TE.

Définition 2: ($Tache_k$)
Etant donné deux ensembles $P = \{P_1, P_2, \ldots, P_n\}$ et $T = \{T_1, T_2, \ldots, T_m\}$, une tâche $Tache_k$ construite sur P et T est un 4-uplet (P^k, T^k, $Pré_k$, $Post_k$) telle que :

- $P^k \subset P$: Ensemble des places de la $Tache_k$
- T^k : Ensemble des transitions de la $Tache_k$, $T^k = T^i(k) \cup \{T^s(k), T^e(k)\}$ où :
  - $T^i(k)$: Ensemble des transitions à l'intérieur de la $Tache_k$.
  - $T^s(k)$: Transition de début de la $Tache_k$.
  - $T^e(k)$: Transition de fin de la $Tache_k$.

  Notons de plus que :$T^i(k) \cap \{T^s(k), T^e(k)\} = \phi$, de plus, Si $T^s(k) = T^e(k)$ alors $Ti(k) = \phi$
- $Pré_k$: Application d'incidence avant

$$Pré_k : P^k \times T^k \longrightarrow \{0, 1\}$$

- $Post_k$: Application d'incidence arriére

$$Post_k : P^k \times T^k \longrightarrow \{0, 1\}$$

□

Toute transition de la $Tache_k$ appartient à un chemin qui a pour origine la transition de début de la tâche, $T^s(k)$, et pour extrémité la transition de la fin de la tâche, $T^e(k)$.

Pour définir le modèle RdP-TE [1], nous allons utiliser les notations suivantes :

Notations :

- $P_t = \bigcup_k P^k$: Ensemble des places appartenant à toutes les tâches.
- $T_t = \bigcup_k T^k$: Ensemble des transitions appartenant à toutes les tâches.
- T_{dyn} : Ensemble des transitions de T_t régies par des équations différentielles. Nous désignons les transitions de cet ensemble par ***transition linéaire***.
- Nous désignons par ***transition temporelle*** toute transition à laquelle on associe une contrainte temporelle. l'ensemble des transitions temporelles est égale à $T \backslash T_{dyn}$.

Définition 3: Réseau de Petri Temporel Etendu
Un RdP-TE est un 8-uplet (P, T, $Taches$, $Pré$,$Post$, M_0, D, $Init$) avec :

- $P = \{P_1, \ldots, P_n\}$ ensemble fini non vide de places.
- $T = \{T_1, \ldots, T_m\}$ ensemble fini non vide de transitions.
- $Taches = \{Tache_1, \ldots, Tache_p\}$ ensemble fini de tâches construites sur l'ensemble P et T conformément à la définition 2, tel que : $\forall\ Tache_j, \forall\ Tache_k,\ \ P^j \cap P^k = \phi,\ T^j \cap T^k = \phi\ ,\ k \neq j$ et $k, j \in \{1, \ldots, p\}$
- $Pré$: Application d'incidence avant :

$$Pré = \begin{cases} (P \setminus P_t) \times (T \setminus T_t) \longrightarrow \{0, 1\} \\ \quad Pré(p, t) \text{ app. d'incidence avant.} \\ P^k \times T^k \longrightarrow \{0, 1\} \ \ (\forall k \in \{1, \ldots, p\}) \\ \quad Pré = Pré_k, \text{ pour les arcs d'une tâche.} \end{cases}$$

- $Post$: Application d'incidence arriére :

$$Post = \begin{cases} (P \setminus P_t) \times (T \setminus T_t) \longrightarrow \{0, 1\} \\ \quad Post(p, t) \text{ app. d'incidence arriére.} \\ P^k \times T^k \longrightarrow \{0, 1\} \ \ (\forall k \in \{1, \ldots, p\}) \\ \quad Post = Post_k, \text{ pour les arcs d'une tâche.} \end{cases}$$

- M_0 : Marquage initial.
- D : Application qui définit la dynamique (voir remarque II.2) :

$$D = \begin{cases} T_t \longrightarrow \mathbb{Q} \times \mathbb{Q} \times \mathbb{Q} \\ \quad D(T_i) = (a, b, c),\ T_i \in Tache_k\ . \\ T \setminus T_t \longrightarrow \mathbb{Q} \times \mathbb{Q} \\ \quad D(T_i) = (b, c),\ T_i \notin Tache_k. \end{cases}$$

- $Init$: fonction d'initialisation $Init$: On associe pour une transition T_i une expression $EXPRESSION(T_i)$ en fonction des places en amont de la transition.

□

Remarque II.1: Pour toute transition appartenant à une tâche donnée, nous associons une initialisation (affectation) sous forme d'expression mathématique construite sur l'ensemble des places en amont de la transition. Cette expression permet d'établir la valeur d'une variable soit par une constante, soit en fonction des valeurs déjà atteinte dans les éventuelles activités précédentes. Dans le cas de synchronisation, les places participant à cette synchronisation, doivent appartenir à une même tâche.

Remarque II.2: La fonction D définit la dynamique de la transition. Les dynamiques continues doivent appartenir à une tâche. On trouve les dynamiques temporelles au niveau des tâches et au niveau de toutes les transitions n'appartenant à aucune tâche. Ainsi, nous avons deux cas : **Cas 1 :** $D(T_k) = (a, b, c)$ avec $T_k \in T_t$.

– Si $a = 0$ T_k est une transition temporelle, dans la transition T_k, b et c représente les bornes min et max de l'intervalle de franchissement.

– Si $a = k \neq 0$ T_k est une transition linéaire), , dans la transition T_k, k représente la dynamique avec la quelle elle évolue, b et c représentent les deux bornes (min et max) délimitant l'intervalle de franchissement.

Cas 2 : $D(T_k) = (b, c)$ avec $T_k \in T \backslash T_t$, T_k est une transition temporelle, b et c représente les bornes min et max de l'intervalle de franchissement de la transition T_k.

B. Etat d'un Réseau de Petri Temporel étendu

Un état d'un RdP-TE est le triplet $E = (M, V, I)$

1. M : vecteur du marquage identifiant le nombre de jetons pour chaque place. $(\forall P_i \in P, M(P_i) \geq 0)$.

2. V : Vecteur de validation, une composante $V_j = i$ si la transition T_j est validée i fois. La composante $V_j = 0$ si T_j n'est pas validée.

3. I : Vecteur des ensembles des intervalles de franchissements, $I_j, j \in \{1, \ldots, m\}$, de T_j. Le nombre de ces intervalles est égale à V_j, c'est à dire au nombre de fois qu'elle est validée. Un élément de I_j est soit l'intervalle de temps dans lequel elle peut être franchie si $T_j \in T \backslash T_{dyn}$, soit l'intervalle des valeurs de la variable si $T_j \in T_{dyn}$.

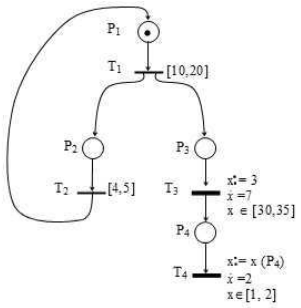

Fig. 1. Etat du RdP Temporel étendu.

Pour illustrer cette notion nous considérons l'exemple de la figure 1, l'état initial du RdP-TE est $E_0 = (M^0, V^0, I^0)$ avec :

- Le marquage $M^0 = (1, 0, 0, 0)$
- La transition T_1 est une fois validée (une seule marque est présente dans P_1), donc $V^0 = (1, 0, 0, 0)$
- $I_1^0 = \{[10, 20]\}$

Définition 4: (Condition de franchissement d'une transition)
Une transition $T_j \in T, j \in \{1, \ldots, m\}$ est franchissable à un instant t depuis un état $E = (M,\ V,\ I)$ si :

1. T_j est validée par le marquage M.

2.

(a) Si $T_j \in T \backslash T_{dyn}$, il faut que $t \in I_j$.
(b) Si $T_j \in T_{dyn}$, il faut que x à l'instant t appartient à I_j.

□

Ainsi, le franchissement d'une transition T_k depuis un état $E = (M, V, I)$ à un instant t, conduit à un nouvel état $E' = (M', V', I')$ telle que :

1. Le marquage M' est défini par l'équation suivante :

$$\forall j \in \{1, \ldots, n\}, \quad M'(P_j) = M(P_j) - Pré(P_j, T_k) + Post(P_j, T_k)$$

2. Les intervalles de franchissement I' sont donnés comme suit :

(a) Si la transition T_j est une transition temporelle validée par le marquage M et si elle n'est pas en conflit avec la transition T_k alors,

$$I'_j = [max(0, a_j - t), b_j - t].$$

(b) Si T_j est une transition linéaire validée par le marquage M et si elle n'est pas en conflit avec la transition T_k et définie par l'initialisation $x := m$ et la dynamique $\dot{x} = k\,;\, x \in [a_j, b_j]$:

$$I'(j) = [a_j - kt, b_j - kt].$$

(c) Pour toute autre transition T_j :
- Si T_j est une transition temporelle à avec l'intervalle $[a_j,\ b_j]$, alors $I_j = [a_j, b_j]$
- Si T_j est une transition linéaire sur la variable x avec une dynamique $x := m\,;\, \dot{x} = k\,;\, x \in [a_j,\ b_j]$, alors $I_j = [a_j - m,\ b_j - m]$

Exemple II.1: systéme de traitement thermique

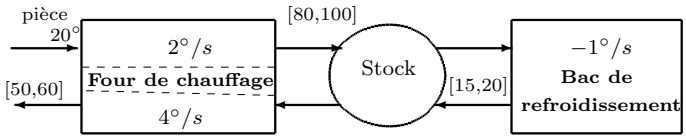

Fig. 2. Systéme de traitement thermique.

Pour illustrer le RdP-TE, nous considérons un systéme de traitement thermique ([1]) conformément à la figure 2 . Une piéce subit une succession de chauffes et de refroidissements. Nous supposons, de plus, qu'on dispose de deux palettes pour transporter les piéces et d'un stock, de capacité non limitée, entre le four de chauffe et le bac de refroidissement. Les transferts intermédiaires prennent de zéro à une unité de temps. Initialement, la piéce a pour température 20°, elle subit une action de chauffe dans le four avec une vitesse de $2°/s$ jusqu'à ce qu'elle atteigne une température entre 80° et 100°. A la fin du chauffage, la piéce est déposée dans le stock thermiquement isolé. Ensuite, elle est déposée dans un bac pour subir un refroidissement avec une dynamique $-1°/s$. Le refroidissement s'arrête lorsque la piéce atteint une température entre 15° et 20°. Aprés le dépôt de la piéce dans le stock intermédiaire, la piéce est chauffée dans le four une derniére fois avec une vitesse de $4°/s$. Enfin, la tâche du systéme pour cette piéce se termine lorsque la température de la piéce atteint une valeur comprise entre 50° et 60°. Notons aussi que le four et le bac de refroidissement ne peuvent contenir qu'une seule piéce à la fois.

Nous décrivons, dans la figure 3, la modélisation du systéme de traitement thermique par un RdP temporel étendu conformément à la définition 3. Dans notre cas, il existe une seule tâche correspondant au traitement thermique d'une piéce. Les transitions de début et de fin de tâche sont représentées par des traits en gras et correspondent successivement aux transitions T_1 et T_6.

Remarque II.3: Plusieurs extensions des RdP Temporels ont été proposées pour permettre d'exprimer la préemption et la reprise d'une action les Scheduling-TPN [8], les

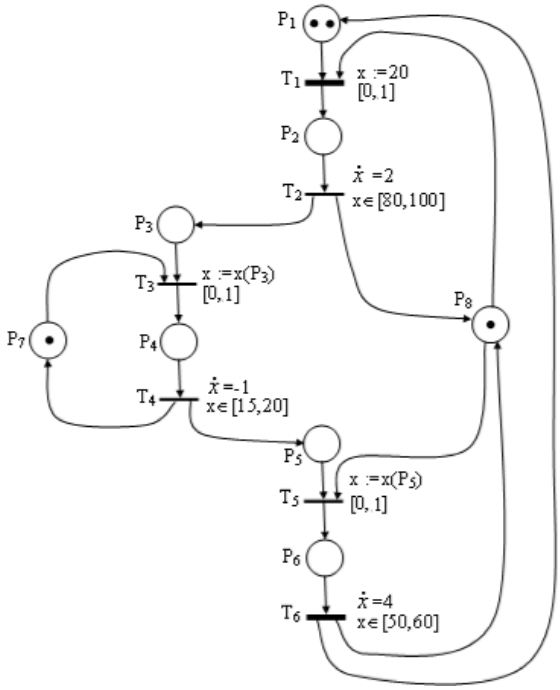

Fig. 3. RdP temporel étendu du systéme de traitement thermique.

Preemptive-TPN [9]. Les Scheduling-TPN (RdP étendu à l'ordonnancement) introduisent la notion de transition active. L'activité d'une transition dépend d'une fonction *Act* qui active ou désactive la transition. Ainsi, on peut modéliser la préemption d'une action mais en dépit de la puissance graphique du réseau de Petri. En effet la synchronisation de transitions est réalisée par une fonction mathématique ajoutée dans le modéle graphique. Dans le même esprit, les Preemptive-TPN ajoutent des priorités au places. Une transition en progression (validée, l'horloge en marche) peut être suspendue si une transition en conflit et plus prioritaire devient validée. Le tir de cette derniére permet de reprendre la progression de la transition. La modélisation de la dépendance entre plusieurs actions est plus difficile à représenter.

III. Du RdP-TE vers l'Automate hybride linéaire

Dans cette section, nous commonçons par expliquer le principe de la transformation. d'une manière intuitive. Cela est finalisé par la suite à travers l'algorithme de transformation.

Les réseaux de Petri ont connu depuis leur invention un réel succés en raison de leurs simplicité mathématique, des avantages que procurent leurs représentations graphiques ainsi que leurs compacité. Les automates hybrides constituent un outil puissant d'analyse. Dans la littérature, nous pouvons constater plusieurs travaux concernant la transformation des réseaux de Petri vers les automates en allant des RdP ordinaires aux graphes de marquages ; RdP temporel s aux automates temporisés [10], [11] et [12], [13], ainsi que des RdP hybrides aux automates hybrides [14], [15].

Dans le même objectif d'analyse et de synthèse de la commande, nous proposons de transformer les RdP-TE à une classe d'automate hybride linéaire. Nous commençons par illustrer le principe de façon intuitive. Ensuite, nous donnons l'algorithme de transformation.

L'état d'un RdP-TE est déterminé par l'évolution du marquage des places due au franchissement des transitions. Une transition franchie doit vérifier la condition de marquage et obéir à des contraintes temporelle ou à des conditions imposées sur la valeur de la variable. Le marquage des places reste inchangé entre deux franchissements successifs. Ainsi, on associe pour chaque marquage du RdP-TE, un sommet de l'automate hybride équivalent. A chaque franchissement d'une transition du RdP-TE on associe une transition de l'automate hybride équivalent.

Les informations correspondant à l'écoulement du temps et à l'évolution des variables peuvent également être modélisées par les éléments suivants de l'automate hybride équivalent :

1. Le temps écoulé à partir de l'instant de validation d'une transition est mesuré par une horloge (une variable dans le sommet évoluant avec la dynamique $\dot{t} = 1$)

2. La valeur de la variable (du RdP-TE) atteinte suite à la validation d'une transition est mesurée par une variable associée au sommet évoluant suivant la dynamique $\dot{x} = k$ associée à la transition.

3. L'intervalle de franchissement d'une transition correspond à la garde de la transition équivalente construite dans l'automate hybride.

4. L'initialisation des horloges des transitions nouvellement validées est modélisée par une initialisation de l'horloge correspondante

5. L'initialisation d'une variable d'une transition nouvellement validée est également modélisée par une initialisation au niveau de la transition d'entrée correspondante.

A. Identification des éléments de l'automates hybride linéaire

Les différents éléments de l'AHL modélisant le comportement d'un RdP-TE sont identifiés comme suit :

A.1 Variables

Deux types de variables sont identifiable :

- Pour chaque transition temporelle T_j n'appartenant à aucune tâche, on associe une horloge t_j qui permet de compter le temps écoulé depuis la derniére validation de la transition T_j. Ceci permet de vérifier si la condition de franchissement de la transition est vérifiée.
- Pour chaque tâche $Tache_k$, on associe une variable x_k. Une instance de x_k est créée à chaque fois que la transition de début de tâche est validée (l'instance est morte quand on franchit la transition de fin de tâche). Pour une transition validée T_j appartenant à une tâche $tache_k$:
  - Si la transition T_j est linéaire, nous associons à T_j une instance de la variable x_k
  - Si la transition T_j est temporelle, nous associons à T_j une instance de la variable x_k et une horloge t_j. Dans ce cas le sommet de l'automate est régi par l'équation $\dot{x_k} = 0$ et $\dot{t_j} = 1$ puisque la transition est associée à une contrainte temporelle.

A.2 Sommets

Pour chaque marquage M_i, nous associons un sommet unique L_i de l'automate hybride linéaire correspondant. Le marquage évolue par franchissement de transitions. Une transition non validée ne peut être franchie. Ainsi, au sein d'un sommet L_i (correspondant au marquage M_i), on ne considére que les horloges des transitions validées (en plus des éventuelles instances des variables des tâches).

A.3 Transition

Pour chaque franchissement d'une transition dans le graphe de marquage du RdP-TE, nous associons une transition au niveau de l'AHL. La garde de la transition modélise l'intervalle de franchissement de la transition du RdP-TE.

B. Construction de l'automate hybride linéaire associée au RdP-TE

Le principe de construction de l'AHL est inspiré des travaux dans [10], [11] concernant la conversion du RdP temporel en un automate temporisé. Le marquage initial est associé au sommet initial de l'automate. La construction de l'AHL se poursuit en analysant l'évolution du marquage.

B.1 Création d'un sommet

Considérons le marquage actuel M_n du RdP-TE. M_n sera modélisé par un sommet L_n dans l'automate correspondant.

Nous recherchons les transitions validées par le marquage M_n. Supposons que les transitions validées par M_n sont T_i et T_j, avec T_i et T_j deux transitions temporelles n'appartenant à aucune tâche. Dans ce cas, L_n sera régi par deux horloges t_i et t_j (avec les deux équations suivantes : $\dot{t_i} = 1$ et $\dot{t_j} = 1$. Supposons qu'une des transitions validés (T_k) est une transition de début de tâche ($Tache_l$), une instance de la variable x_l ($x_{l,1}$) sera créée. Si $D(T_k) = (a, b, c)$, alors on aura l'équation $\dot{x_{l,1}} = a$. Si en plus $a = 0$ (le cas où T_k est une transition temporelle) alors L_n est regie en plus de l'horloge t_k.

B.2 Calcul des invariants des sommet

L'invariant d'un sommet L_n, $I(L_n)$, est calculé à partir des intervalles de franchissement des transitions validées. Deux cas se présentent :

Cas 1 : La transition T_i est une transition temporelle avec l'intervalle de franchissement $[b_i,\ c_i]$. Ainsi, T_i n'est plus validée aprés l'écoulement de c_i de c_i unités de temps. l'invariant de sommet correspondant serait $0 \leq t_i \leq c_i$.

Cas 2 : La transition T_i est une transition linéaire. T_i est alors une transition de tâche ($Tache_l$). Si $D(T_i) = (a_i, b_i, c_i)$ alors T_i est associée à une contrainte de variable du type $x \in [b_i,\ c_i]$. Soit $x_{l,1}$ l'instance de la variable x_l de la tâche $tache_l$. L'invariant de sommet équivalent dépend du signe de la dynamique :

1. Si $a_i > 0$ alors l'invariant de sommet équivalent serait $x_{l,1} \leq c_i$
2. Si $a_i < 0$ alors l'invariant de sommet équivalent serait $x_{l,1} \geq b_i$

B.3 Création d'une transition

Soit M_{n+1} le nouveau marquage atteint suite au franchissement d'une transition T_i. On crée un nouveau sommet L_{n+1}, le sommet correspondant à M_{n+1}. Le franchissement de la transition T_i est modélisé au niveau de l'automate par la transition $T_{n,n+1}$ qui relie L_n à L_{n+1}.

Cas 1 : La transition T_i est une transition temporelle n'appartenant à aucune tâche. La contrainte temporelle $[b_i,\ c_i]$est associée à T_i. La garde est $b_i \leq t_i \leq c_i$.

Cas 2 : La transition T_i est une transition appartenant à une tâche. Supposons que $D(T_i) = (a_i, b_i, c_i)$, la garde associé à T_i est calculé comme suit :

- Si $a_i = 0$ alors dans ce cas une contrainte temporelle est associée à la transition temporelle T_i . La garde est $b_i \leq t_i \leq c_i$
- Si $a_i \neq 0$ alors dans ce cas la garde associée à la transition linéaire T_i est $bi \leq x_{l_j} \leq c_i$. Avec j dépend de l'instance de la variable.

L'évolution du marquage peut mener vers des transitions nouvellement validées. Supposons que T_k est une transition nouvellement validée. L'affectation au niveau de $T_{n,n+1}$ dépend du type de la transition T_k :

Cas 1 : Si la transition T_k est une transition temporelle n'appartenant à aucune tâche, alors l'horloge t_k doit être mise à zéro au niveau de $T_{n,n+1}$

Cas 2 : Si la transition T_k appartient à une tâche et $D(T_i) = (a_i, b_i, c_i)$ alors :

- Si $a_i \neq 0$ (transition linéaire), on associe l'affectation correspondante à $init(T_k)$ à $T_{n,n+1}$
- si $a_i = 0$ (transition temporelle), dans ce cas la variable est toujours active car on est toujours au sein de la tâche et la transition produit un effet de retard, donc une horloge t_k est également active. Ainsi, deux affectations serons associées à la transition $T_{n,n+1}$: $x_{l,1} := init(T_k)$ et $t_k := 0$

C. Algorithme de transformation

Nous utilisons les notations suivantes :

- $\mathcal{M}$ désigne l'ensemble des marquages du RdP-TE.
- $\mathcal{L}$ désigne l'ensemble des sommets du l'AHL.
- $\mathcal{T}$ désigne l'ensemble des transitions de l'AHL.
- $\mathcal{P}$ désigne la pile qui enregistre les visites à faire sous forme de couple (sommet, transition).

L'algorithme de transformation d'un RdP-TE vers un AHL est donnée comme suit :

Algorithme 1 :
Debut

PAS 1 : Initialisation

- Créer L_0 correspondant à M_0
- Créer $T_{0,0}$ la transition d'entrée à L_0
- Empiler le couple $(L_0, T_{0,0})$ dans une pile $\mathcal{P}$
- Mettre à jour les ensembles $\mathcal{M} := \{M_0\}$, $\mathcal{L} := \{L_0\}$ et $\mathcal{T} := \{T_{0,0}\}$

PAS 2 : Analyse du sommet de la pile $\mathcal{P}$. Supposons qu'il s'agit du couple $(L_n, T_{m,n})$, càd la visite du sommet L_n suite au franchissement de la transition $T_{m,n}$

- Dépiler $\mathcal{P}$
- Déterminer l'ensemble des transitions validées
- Analyser l'évolution engendrée par le franchissement de chaque transition. Pour chaque transition T_j, on

détermine le marquage M_{n+1} du RdP-TE atteint par son franchissement :
SI $M_{n+1} \notin \mathcal{M}$ ($\mathcal{M}$ étant l'ensemble des marquages visités)
- Créer un sommet L_{n+1} correspondant à M_{n+1}
- Créer une transition $T_{n,n+1}$ correspondant au franchissement de T_j
- Empiler $(L_{n+1}, T_{n,n+1})$ dans $\mathcal{P}$
- Mettre à jour les ensembles $\mathcal{M} := \mathcal{M} \cup \{M_{n+1}\}$, $\mathcal{L} := \mathcal{L} \cup \{L_{n+1}\}$ et $\mathcal{T} := \mathcal{T} \cup \{T_{n,n+1}\}$

SI $M_{n+1} \in \mathcal{M}$, Alors
- Créer une nouvelle transition correspondante au franchissement de T_j (si elle n'existe pas déjà)
- $\mathcal{T} := \mathcal{T} \cup \{T_{n,k}\}$ avec L_k le sommet correspondant à M_{n+1}.

PAS 3 : Si $P \neq \emptyset$ aller à PAS 2 Sinon Aller à Fin
Fin

Proposition 1: L'algorithme converge pour les RdP-TE bornés.

preuve : L'algorithme se termine lorsqu'il n'y a aucune nouvelle visite d'un sommet à analyser. Une visite d'un sommet est complètement caractérisée par le marquage du RdP-TE. Ainsi l'algorithme converge si le nombre de sommets et le nombre de fois que chaque sommet est visité est fini.

Un RdP-TE bornée est caractérisé par le fait que le nombre de marques de chaque place est fini. Par conséquent le nombre de vecteurs de marquage est fini. Chaque marquage est représenté par un sommet de l'AHL. Ainsi, l'AHL associé au RdP-TE dans la classe considérée a un nombre finie de sommets.

La visite d'un sommet du AHL correspond à l'exécution d'une transition. Si le sommet n'est pas encore crée alors on empile le sommet et la visite dans la pile. Si le sommet est déjà créé, alors on crée la transition et on ne sauvegarde pas la visite dans la pile. Puisque le nombre de sommets est fini, alors le nombre de transition est aussi fini.

Le comportement temporel de l'AHL obtenu est similaire au comportement du RdP-TE puisque le nombre de sommets est égal au nombre de marquages du graphe de marquage. Les horloges actives dans un sommet reflètent le comportement des transitions validées. Les variables des tâches sont aussi représenté dans les sommets leur valeurs sont éventuellement préservées par les affectations associées au AHL.

IV. Exemple de transformation : systéme de traitement thermique

Nous allons construire l'AHL $\mathcal{A}$ équivalent au RdP-TE $\mathcal{R}$ de la figure 3, en appliquant l'algorithme 1 de la section III-C. Le RdP $\mathcal{R}$ contient une seule tâche, et par conséquence, une seule variable (qui peut posséder, éventuellement, plusieurs instances). Initialement, la transition T_1 du RdP $\mathcal{R}$ est validée par le marquage initial. Ceci revient à créer un sommet initial conformément à la figure 4. La transition T_1 est une transition de début de tâche, alors on crée une nouvelle instance de la variable x_1, soit $x_{1,1}$. Nous créeons une transition d'entrée pour ce sommet. L'instance $x_{1,1}$ sera initialisée par l'initialisation statique de T_1, soit $x_{1,1} := 20$. Les dynamiques qui seront actives dans L_0 seront celle des transitions validées. Dans ce cas de figure, seule T_1 est validée.

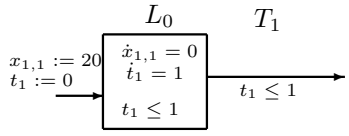

Fig. 4. Construction du premier sommet de l'automate.

La transition T_1 est une transition temporelle, alors on crée une horloge t_1 initialisée au niveau de la transition d'entrée ; la dynamique associée dans L_0 est $\dot{t_1} = 1$. Etant donné que T_1 est une transition temporelle, la valeur de $x_{1,1}$ n'évolue pas. Ainsi, L_0 est régie par l'equation $\dot{x_{1,1}} = 0$. L'intervalle de franchissement de T_1 est $[0, 1]$. Ainsi, On associe la garde $t_1 \leq 1$ à L_0.

L'algorithme que nous avons proposé génére les sommets sans prendre en compte l'atteignabilité des sommets. Notons que le probléme d'atteignabilité des sommets d'un automate hybride linéaire est indécidable [16]. Néanmoins, avec certains outils informatique se basant sur des approximations comme HyTech [3] et PHAVer [4], [5], on peut calculer l'espace atteignable. Le franchissement de T_1 conduit vers un nouveau marquage où T_2 est validée.

En appliquant l'algorithme 1 (section III-C), on obtient l'automate $\mathcal{A}$ de la figure 5.

V. Analyse de l'automate hybride linéaire

Dans cette section, nous allons analyser l'automate hybride linéaire $\mathcal{A}$ de la figure 5. Nous utilisons pour cela l'outil PHAVer[4], [5]. L'analyse avant de l'automate $\mathcal{A}$ permet d'affirmer que tous les états de l'automate sont atteignables. L'outil PHAVer, exécuté sous cygwin, nécessite 24 itérations pour analyser l'atteignabilité de l'automate. L'analyse avant permet de donner le résultat illustré dans le tableau I.

TABLE I
L'espace atteignable de $\mathcal{A}$.

L_0	$x_{12} = 20$ & $x_{11} = 20$ & $t_5 = 0$ & $t_3 = 0$ & $t_1 \geq 0$ & $-t_1 \geq -1$
L_1	$x_{12} = 20$ & $t_1 = 0$ & $x_{11} \geq 20$ & $-x_{11} \geq -100$
L_2	$x_{12} = 20$ & $x_{11} \geq 20$ & $-t_1 \geq -1$ & $-t_3 \geq -1$
L_3	$t_1 = 0$ & $-x_{11} \geq -100$ & $x_{12} \geq 20$ & $x_{11} \geq 20$ & $-t_3 \geq -1$
L_4	$x_{12} = 20$ & $-t_1 \geq -1$ & $x_{11} \geq 15$
L_5	$-x_{11} \geq -100$ & $x_{12} \geq 20$ & $x_{11} \geq 80$ & $-t_3 \geq -1$
L_6	$x_{12} \geq 20$ & $x_{11} \geq 15$ & $-x_{12} \geq -100$
L_7	$x_{12} = 20$ & $-x_{11} \geq -20$ & $x_{11} \geq 15$ & $-t_1 \geq -1$ & $-t_5 \geq -1$
L_8	$x_{12} \geq 20$ & $x_{11} \geq 15$
L_9	$-x_{11} \geq -20$ & $x_{12} \geq 20$ & $x_{11} \geq 15$ & $-x_{12} \geq -100$
L_{10}	$-x_{11} \geq -100$ & $x_{12} \geq 20$ & $x_{11} \geq 15$ & $-t_3 \geq -1$ & $-t_5 \geq -1$
L_{11}	$t_5 = 0$ & $-x_{11} \geq -60$ & $x_{12} \geq 20$ & $x_{11} \geq 15$ & $-t_3 \geq -1$
L_{12}	$-x_{11} \geq -100$ & $-x_{12} \geq -20$ & $x_{12} \geq 15$ & $x_{11} \geq 15$ & $-t_5 \geq -1$
L_{13}	$x_{11} \geq 15$ & $x_{12} \geq 15$ & $-x_{11} \geq -60$
L_{14}	$-x_{12} \geq -20$ & $x_{12} \geq 15$ & $x_{11} \geq 15$ & $-x_{11} \geq -60$
L_{15}	$x_{12} = 20$ & $t_1 - t_5 = 0$ & $-x_{11} \geq -20$ & $x_{11} \geq 15$ & $t_1 \geq 0$ & $-t_1 \geq -1$
L_{16}	$x_{12} = 20$ & $t_5 = 0$ & $t_1 = 0$ & $x_{11} \geq 15$ & $-x_{11} \geq -100$
L_{17}	$x_{12} = 20$ & $t_5 = 0$ & $-t_1 \geq -1$ & $x_{11} \geq 15$ & $-x_{11} \geq -60$
L_{18}	$t_3 = 0$ & $-x_{11} \geq -100$ & $x_{11} \geq 15$ & $-t_5 \geq -1$ & $x_{12} \geq 15$

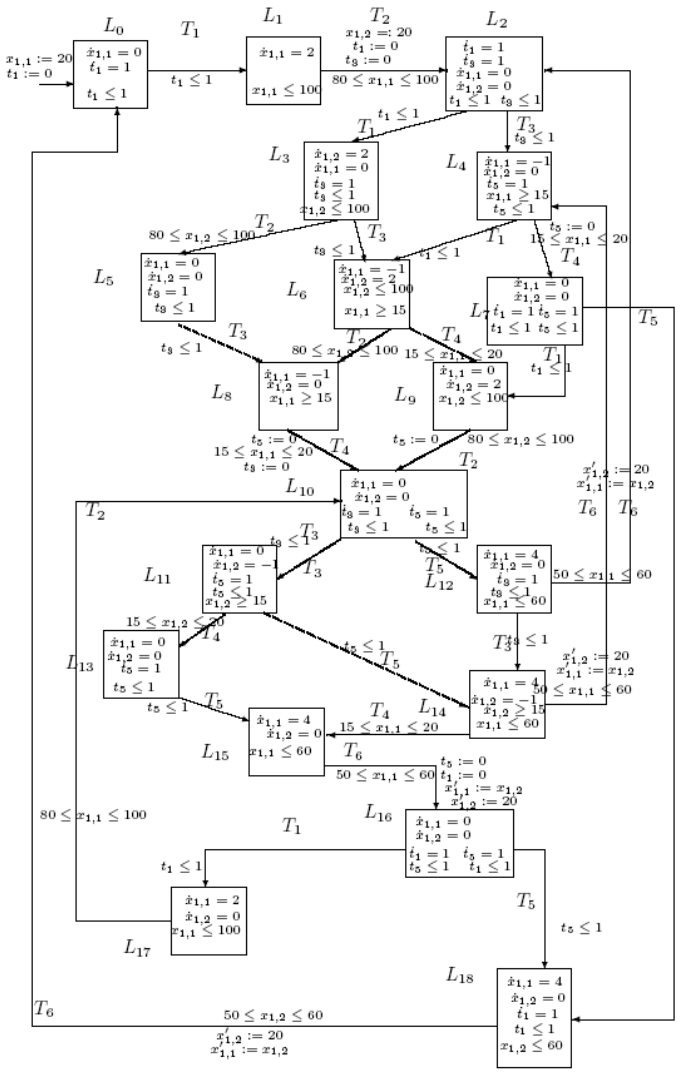

Fig. 5. L'AHL construit à partir du RdP de la figure 3 en appliquant l'algorithme 1.

Nous avons ajouté une horloge artificielle y pour déterminer le temps de séjour dans les différents sommets. Nous allons, par exemple, calculer le temps minimal pour commencer la derniére phase de chauffage. Pour cela, calculons l'espace atteignable pour les sommets L_{12}, L_{14} et L_{15}. Les résultats de calcul donnée par PHAVer sont illustrés dans le tableau II.

Ainsi, nous pouvons déduire que pour les sommets L_{15} et L_{12}, on a $y \geq 90$; Ceci veut dire que 90 u.t sont nécessaires pour entamer la derniére action de chauffage de la premiére piéce. De nombreuses performances du systéme peuvent être déduites de l'automate atteignable. Certaines peuvent être interprétées directement sur le RdP, d'autres sont liées à l'automate. Le fait de disposer du modéle dynamique exact ouvre la voie aux techniques de vérification et de synthése.

VI. Conclusion

Dans le cadre de cet article, nous avons utilisé le RdP-TE qui permet de modéliser une sous-classe des systémes dynamiques hybrides basée sur les réseaux de Petri permettant de modéliser l'évolution des parties continues avec des vitesses constantes (température, etc.) et ainsi la description des processus continus séquentiels.

TABLE II

Espace atteignable pour les sommet L_{12}, L_{14} et L_{15}.

L_{12}	t5 == 0 & -x_{11} ≥ -60 & -t3 ≥ -1 & 2*x_{11} - x_{12} + 2*y ≥ 120 & -2*t3 - x_{12} + 2*y ≥ 80 & -2*t3 + 2*x_{11} - x_{12} + 2*y ≥ 120 & -t3 + y ≥ 90 & -t3 + x_{11} + y ≥ 110 & -x_{11} - 2*x_{12} + 4*y ≥ 140 & -x_{11} + 4*y ≥ 340 & x_{12} ≥ 20 & x_{11} ≥ 15 & x_{11} + y ≥ 110
L_{14}	x_{12} ≥ 15 & -x_{11} - 2*x_{12} + 4*y ≥ 140 & -x_{11} + 4*y ≥ 340 & x_{12} + y ≥ 110 & x_{11} ≥ 15 & x_{11} + y ≥ 110 & x_{11} + x_{12} + y ≥ 130 & -x_{11} ≥ -60 & 2*x_{11} - x_{12} + 2*y ≥ 120
L_{15}	-x_{11} + 4*y ≥ 340 & -x_{12} ≥ -20 & x_{12} ≥ 15 & x_{12} + y ≥ 110 & x_{11} ≥ 15 & x_{11} + x_{12} + y ≥ 130 & -x_{11} ≥ -60

Dans le but de l'analyse des systémes modélisés par les RdP-TE, nous avons proposé un algorithme permettant de transformer systématiquement, de maniére structurelle le RdP-TE en un automate hybride équivalent. Ainsi, nous pouvons bénéficier des nombreux travaux qui existent déjà dans la littérature concernant l'analyse à base d'automate hybride. Des mesures de performances et des vérifications de propriétés de vivacité ainsi que l'analyse de l'atteignabilité ont été réalisés, dans le cadre de notre travail, avec l'outil PHAVer [4], [5]. Les travaux futurs concernent la synthése de la commande des systémes modélisés par le RdP-TE. Ainsi, il serait intéressant de développer des méthodes systématiques de synthése permettant de revenir au modéle initial pour changer les paramétres du modéle. Ainsi, on peut respecter les contraintes imposées pour le bon fonctionnement du systéme.

Références

[1] Y. El Touati, N. Ben Hadj Alouane, et M. Yeddes. Reseaux de petri temporel etendu. *MSR05, RS-JESA*, volume 39, pages 207–222. Hermes, Lavoisier, 2005.

[2] B. Berthomieu et M. Diaz. Modeling and verification of time nets. *IEEE transactions on software Engeneering*, 17(3) :259–273, 1991.

[3] T. Henzinger et H. Wong-Toi. Hytech : A model checker for hybrid systems. *Software Tools for Technology Transfer*, 1 :110–122, 1997.

[4] Manfred Morari et Lothar Thiele. Hybrid systems : Computation and control, 8th international workshop, hscc 2005, zurich, switzerland, march 9-11, 2005, proceedings. *HSCC*, volume 3414 of *Lecture Notes in Computer Science*. Springer, 2005.

[5] Goran Frehse. Phaver : Algorithmic verification of hybrid systems past hytech. *HSCC*, pages 258–273, 2005.

[6] T. Henzinger. The theory of hybrid automata. *proceedings of the 11th annual IEEE Symposium on Logic in Computer Science*.

[7] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, et S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1) :3–34, 1995.

[8] Olivier (H.) Roux et Anne-Marie Déplanche. A t-time Petri net extension for real time-task scheduling modeling. *European Journal of Automation (JESA)*, 36(7) :973–987, 2002.

[9] Giacomo Bucci, Andrea Fedeli, Luigi Sassoli, et Enrico Vicario. Timed state space analysis of real-time preemptive systems. *IEEE Trans. Softw. Eng.*, 30(2) :97–111, 2004.

[10] A. Sava. *Sur la Synthèse de la Commande Des Systèmes à Evenements Discrets Temporisés.* thèse de doctorat, Institut Nationale Polytechnique de Grenoble, 2001.

[11] Alexandru Tiberiu Sava et Hassane Alla. A control synthesis approach for time discrete event systems. *Math. Comput. Simul.*, 70(5) :250–265, 2006.

[12] F. Cassez et O.H. Roux. Traduction structurelle des réseaux de petri temporels vers les automates temporisés. *4 ème Colloque*

sur la Modélisation des Systèmes Réactifs (MSR 03), Metz, France, octobre 2003.

[13] Franck Cassez et Olivier (H.) Roux. Structural translation from time petri nets to timed automata. *Electronic Notes in Theoretical Computer Science : Fourth International Workshop on Automated Verification of Critical Systems (AVoCS'04)*, London (UK), September 2004. Elsevier.

[14] ALLAM M. et ALLA H. From hybrid petri nets to hybrid automata. *Journal Européen des Systèmes Automatisés (JESA)*, volume 32, pages 1165–1185, 1998.

[15] M. Allam. *Sur l'analyse quantitative des réseaux de Petri hybrides : Une approche basée sur les automates hybrides*. Thése d'état de l'inpg, Institut Nationale Polytechnique de Grenoble, 1998.

[16] Thomas A. Henzinger, Peter W. Kopke, Anuj Puri, et Pravin Varaiya. What's decidable about hybrid automata ? pages 373–382, 1995.