Traduction structurelle des Réseaux de Petri Temporels vers les Automates Temporisés

Franck Cassez et Olivier H. Roux

IRCCyN/CNRS UMR 6597
BP 92101
1 rue de la Noë
44321 Nantes Cedex 3 France,
email : Prénom.Nom@irccyn.ec-nantes.fr

RÉSUMÉ. *Dans cette article, nous considérons les réseaux de Petri t-temporels (RdPT) c'est à dire pour lesquels le temps est associé aux transitions sous la forme d'un interval. Nous en donnons une sémantique formelle en terme de systèmes de transitions temporisés. Nous proposons ensuite une traduction structurelle des RdPTs en un produit synchronisé d'automates temporisés (ATs) qui préserve la sémantique comportementale (bisimilarité temporelle) des RdPTs. Les conséquences théoriques de ce résultat sont : i) les problémes d'accessibilité et plus généralement de model-checking de TCTL sont décidables pour les RdPTs bornés ; ii) il est possible d'étendre les RdPTs en considérant des contraintes temporelles strictes sur les transitions en préservant les résultats décrits plus haut en i). D'un point de vue pratique, les conséquences sont doubles : i) on peut spécifier un système à l'aide de RdPTs et d'automates temporisés et en donner une sémantique facilement ; ii) les outils disponibles pour l'analyse des automates temporisés (comme KRONOS ou UPPAAL ou CMC) peuvent être utilisés pour la vérification de RdPTs.*

ABSTRACT. *In this paper, we consider Time Petri Nets (TPN) where time is associated with transitions. We give a formal semantics for TPNs in terms of Timed Transition Systems. Then, we propose a translation from TPNs to Timed Automata (TA) that preserves the behavioural semantics (timed bisimilarity) of the TPNs. For the theory of TPNs this result is two-fold: i) reachability problems and more generally TCTL model-checking are decidable for bounded TPNs; ii) allowing strict time constraints on transitions for TPNs preserves the results described in i). The practical applications of the translation are: i) one can specify a system using both TPNs and Timed Automata and a precise semantics is given to the composition; ii) one can use existing tools for analysing timed automata (like KRONOS or UPPAAL or CMC) to analyse TPNs.*

MOTS-CLÉS : *Réseaux de Petri temporels, automates temporisés, vérification, bisimulation, logique temporelle.*

KEYWORDS: *Time Petri nets, timed automata, timed-bisimulation, model-checking, temporal logics.*

1re soumission à *MSR 03*, le 26 février 2003.

1. Introduction

Classes de Réseaux de Petri avec du temps.

Il existe deux principales familles d'extension temporelle des réseaux de Petri : les réseaux de Petri temporisés introduits par Ramchandani [RAM 74] et les réseaux de Petri temporels introduits par Merlin [MER 74]. Pour les réseaux de Petri temporisés, les temporisations ont d'abord été associées aux transitions (t-temporisés), puis aux places (p-temporisés). La temporisation représente alors la durée minimale de tir ou le temps de séjour minimum d'un jeton dans une place (ou durée exacte avec une règle de fonctionnement au plus tôt). Les RdP t-temporisés et p-temporisés sont équivalents et sont une sous-classe des réseaux de Petri temporels.

Concernant les réseaux de Petri temporels, l'extension temporelle s'exprime sous la forme d'un intervalle associé principalement aux transitions (t-temporel) [MER 74], ou aux places (p-temporel). En ce qui concerne l'expressivité des réseaux de Petri p-temporels et t-temporels, Khansa a montré [KHA 96] que ces deux modèles sont incomparables. Enfin, les réseaux de Petri t-temporels forment une sous classe des *Time Stream Petri Nets* [DIA 94] qui ont été introduits pour modéliser des applications multimédia.

Avec une sémantique classique, et en particulier sur les réseaux de Petri t-temporels que nous considérons et que nous noterons RdPT, le problème de la *bornitude* est indécidable et les travaux sur ce modèle reportent les résultats de décidabilité (comme l'accessibilité [POP 91]) sur l'hypothèse de bornitude du réseau. La bornitude (et les autres problèmes) sont alors résolus par le calcul de l'espace d'états (quand celui-ci termine). Des travaux récents de Parosh et al [ABD 01] considèrent des RdPTs à *Arcs temporisés* pour lesquels chaque jeton possède une horloge représentant son "âge". Avec un algorithme d'exploration en arrière, ils prouvent que la couverture et la bornitude sont décidables pour cette classe de réseaux de Petri. Cependant, ils supposent un comportement *non-urgent* du réseau : le tir d'une transition peut être retardé, même lorsqu'il dépasse l'échéance de tir, ce qui désensibilise la transition (ceci peut être rapproché des machines *lossy* dans les automates à files [ABD 93], pour lesquels certains problèmes deviennent décidables avec une hypothèse *lossy*).

Accessibilité dans les RdPTs.

Comme pour les automates temporisés, le comportement d'un RdPT peut être défini par des séquences de tir (de transitions) temporisées qui sont des séquences de paires (t, d) où $t \in T$ est une transition du RdPT et $d \in \mathbb{R}_{\geq 0}$. Une séquence $\omega = (t_1, d_1)(t_2, d_2) \cdots (t_n, d_n)$ signifie que t_1 est tirée après d_1 unités de temps, puis t_2 est tirée après d_2 unités de temps et ainsi de suite, de sorte qu'une transition t_i est tirée à la date absolue (depuis l'instant initial) $\sum_{k=1}^{i} d_k$. Un *marquage* M est *accessible* dans un RdPT ssi il existe une séquence temporisée ω conduisant du marquage initial M_0 à M. Il est courant pour les automates temporisés de définir la séquence *non temporisée* à partir de la séquence temporisée : si $\omega = (t_1, d_1)(t_2, d_2) \cdots (t_n, d_n)$ alors $Untimed(\omega) = t_1 t_2 \cdots t_n$. L'analyse de l'accessibilité d'un marquage dans un

RdPT se fait classiquement en construisant le *graphe des classes d'états* (GCE) introduit dans [BER 83], raffiné plus tard dans [BER 91] et plus récemment amélioré dans [LIL 99] par l'utilisation de méthode de réduction d'ordre partiel. Les sommets du GCE représentent des ensembles d'*états* (un état est une paire constituée d'un marquage et d'une contrainte de tir) du RdPT et les arcs sont étiquetés par les noms des transitions. C_0 est la classe d'états contenant le marquage initial M_0. Le graphe des classes d'états défini dans [BER 91] possède les propriétés suivantes : il existe un chemin $C_0 \xrightarrow{\sigma} C$ dans le GCE avec $\sigma \in T^*$ ssi il existe un chemin temporisé $\omega \in (T \times \mathbb{R}_{\geq 0})^*$ tel que $M_0 \xrightarrow{\omega} M$ avec $M \in C$ et *Untimed*$(\omega) = \sigma$. Cela peut être reformulé par : si $L \subseteq T^*$ est le langage accepté par le GCE et L' est le langage temporisé accepté par le RdPT, alors $L =$ *Untimed*(L'). En conséquence le GCE ne peut être utilisé que pour vérifier des propriétés d'accessibilité de marquages, qui sont des propriétés non temporelles[1], ce qui est insuffisant pour vérifier des propriétés temps-réel *quantitatives* telles que : "Il n'est pas possible de rester dans le marquage M plus de n unités de temps" ou "à partir du marquage M le marquage M' est toujours atteint avant n unités de temps".

Automates temporisés.

Les *automates temporisés* (ATs) ont été introduits par Alur et Dill [ALU 94] et ont généré par la suite de nombreux travaux [Fix 94, LAR 95, ALU 99, BOU 00]. Ce modèle est une extension des automates d'état finis avec du temps continu permettant de spécifier des systèmes temps-réel. Il a été montré que la vérification (model-checking) de propriétés exprimées dans la logique temporelle TCTL est décidable [ALU 93, HEN 94] pour les automates temporisés et certaines de leurs extensions [BOU 00]. Cette logique TCTl permet d'exprimer des propriétés avec du temps quantitatif. De plus il existe plusieurs outils efficaces de model-checking tels que UPPAAL [LAR 97, PET 00], KRONOS [YOV 97] et CMC [LAR 98]. Des applications temps-réel industrielles ont été spécifiées et vérifiées avec succès en utilisant ces outils [BEN 96, MAL 96, LIN 98].

Travaux apparentés.

Les relations entre RdPTs et ATs n'ont pas été beaucoup étudiées. Dans [SIF 96], J. Sifakis et S. Yovine se sont principalement intéressés au problème de la *composition*. Ils montrent que pour une sous-classe des réseaux de Petri à flux temporels *saufs*[2], la notion de composition utilisée sur les ATs n'est pas appropriée pour définir ce type de réseaux de Petri comme la composition d'ATs. Ils proposent donc des *automates temporisés à échéance* et une notion plus flexible de composition. Dans [BOR 98] Bornot, Sifakis et Tripakis considèrent des réseaux de Petri à échéance (RdPE) qui sont des réseaux de Petri saufs étendus avec des horloges. Cela implique que les RdPEs sont

1. L'utilisation d'*observateurs* [TOU 97] permet l'expression de propriétés temporelles sous la forme d'un RdPT ; cependant des propriétés sur les marquages sont très difficiles à spécifier avec des observateurs.
2. Un réseau est *sauf* ssi il y au plus un jeton par place dans tous les marquages accessibles.

équivalents à des automates temporisés à échéances dont la structure discrète est le graphe des marquages du réseau de Petri sous-jacent. Ils proposent une traduction des RdPTs vers les RdPEs ce qui définit par transitivité une traduction des RdPTs vers les ATs, ceci pour des réseaux saufs.

Sava et Alla [SAV 01] proposent un algorithme calculant le graphe des marquages d'un RdPT sous la forme d'un automate temporisé. Notre travail se distingue de ce dernier sur les points suivants : i) Leur traduction nécéssite le calcul de l'espace d'état du RdPT alors que la notre est structurelle ; ii) ils ne prouvent pas que l'automate obtenu est *bisimilaire* au réseau initial (ou équivalent dans un sens formellement défini) et ne prouvent aucun critère de correction de leur traduction ; iii) ils ne considèrent que les réseaux de Petri bornés ; iv) ils ne donnent pas de condition d'arrêt de leur algorithme lorsque le réseau de Petri n'est pas borné ; et dans le cas où il est borné ils ne prouvent pas que l'algorithme s'arrête.

Dans cet article nous nous intéressons plus particulièrement à la sémantique des RdPTs, à leurs relations avec les automates temporisés et à la vérification de propriétés de logique temporelle sur les RdPTs. En prenant des critères d'équivalence sémantiques (bisimilarité temporelle) nous donnons à nos algorithmes des fondements rigoureux.

Notre Contribution.

Nous donnons d'abord une *sémantique formelle* des RdPTs [MER 74] en terme de *systèmes de transitions temporisés* [LAR 95]. Nous présentons ensuite une traduction structurelle des RdPTs en un produit synchronisé d'automates temporisés qui préserve la sémantique du réseau de Petri (c'est à dire que le RdPT initial et l'AT résultant de la traduction sont temporellement bisimilaires). Cette opération implique certaines conséquences parmi lesquelles : la logique temporelle TCTL [ALU 93, HEN 94] est décidable pour un RdPT borné, et des propriétés exprimées en TCTL peuvent en pratique être vérifiées sur les RdPTs par l'utilisation d'outils efficaces de model-checking sur les automates temporisés (nous avons implémenté cette traduction qui fournit les automates au format d'entrée d'UPPAAL [LAR 97, PET 00]). C'est à notre connaissance la première preuve *formelle* concernant la décidabilité de TCTL dans les RdPTs bornés. De plus, la traduction étant structurelle, il est possible d'utiliser des outils de test de non bornitude afin de détecter des comportements rendant le RdPT non borné (en général seulement des conditions suffisantes existent pour la non bornitude).

Plan de l'article.

Dans la section 2, nous introduisons la sémantique des RdPTs en terme de systèmes de transitions temporisés et nous donnons les définitions de base concernant les automates temporisés. Dans la section 3 nous montrons comment traduire structurellement un RdPT en un produit synchronisé d'automates temporisés ; nous prouvons alors que les sémantiques du RdPT et du produit d'ATs qui lui est associé sont temporellement bisimilaires. Cela nous permet de vérifier des propriétés temps-réel exprimés en TCTL dans la section 4. Nous présentons notre traducteur et un exemple

de model-checking avec UPPAAL. Enfin nous concluons avec nos travaux en cours et les perspectives dans la section 5.

2. Réseaux de Petri temporels et automates temporisés

Notations.

Nous notons B^A l'ensemble des applications de A dans B. Si A est fini et $|A| = n$, un élément de B^A est aussi un vecteur dans B^n. Les opérateurs usuels $\{+, -, <, =\}$ sur les vecteurs de A^n avec $A = \mathbb{N}, \mathbb{Q}, \mathbb{R}$ sont les extensions vectorielles de leur définition scalaire dans A. On appelle *valuation* un vecteur ν de A^n, et on note ν_i la ième composante de ν. $\overline{0}$ est la valuation telle que $\forall i \in [1..n], \overline{0}_i = 0$. $\mathbb{B} = \{\mathsf{tt}, \mathsf{ff}\}$ représente les valeurs boolénnes *vrai* et *faux*.

Pour une valuation $\nu \in A^n$ et $d \in A$, $\nu + d$ est la valuation (vecteur) telle que $\forall 1 \leq i \leq n, (\nu + d)_i = \nu_i + d$, et pour $A' \subseteq A$, $\nu[A' \mapsto 0]$ est la valuation ν' telle que $\nu'(x) = 0$ pour $x \in A'$ et $\nu'(x) = \nu(x)$ sinon.

$\mathcal{C}(V)$ représente les *contraintes simples* sur un ensemble de variables V. Il est constitué de l'ensemble des combinaisons booléennes (avec les connecteurs $\{\wedge, \vee, \neg\}$) des termes de la forme $v - v' \bowtie c$ ou $v \bowtie c$ avec $v, v' \in V$, $c \in \mathbb{N}$ et $\bowtie \in \{<, \leq, =, \geq, >\}$.

Pour un *système de transitions*, nous notons $s \xrightarrow{a} s'$ pour une transition et $s_0 \stackrel{w}{\Longrightarrow} s_n$ pour la séquence de transitions $s_0 \xrightarrow{a_1} s_1 \rightarrow \cdots \xrightarrow{a_n} s_n$ avec $w = a_1 a_2 \cdots a_n$.

2.1. *Réseaux de Petri temporels*

Les réseaux de Petri temporels, introduit dans [MER 74], étendent les réseaux de Petri autonomes avec des contraintes temporelles sur les tirs des transitions.

Définition 1 (Réseau de Petri temporel) *Un* Réseau de Petri temporel $\mathcal{T}$ *est un n-uplet* $(P, T, {}^\bullet(.), (.)^\bullet, M_0, (\alpha, \beta))$ *avec :*

– $P = \{p_1, p_2, \cdots, p_m\}$ *est un ensemble fini de* places*,*

– $T = \{t_1, t_2, \cdots, t_n\}$ *est un ensemble fini de* transitions*,*

– ${}^\bullet(.) \in (\mathbb{N}^P)^T$ *est la fonction d'incidence* arrière*,*

– $(.)^\bullet \in (\mathbb{N}^P)^T$ *est la fonction d'incidence* avant*,*

– $M_0 \in \mathbb{N}^P$ *est le marquage* initial*,*

– $\alpha \in (\mathbb{Q}_{\geq 0})^T$ *et* $\beta \in (\mathbb{Q}_{\geq 0} \cup \{\infty\})^T$ *sont les fonctions désignant respectivement les dates de tir* au plus tôt *et* au plus tard*.* □

Remarque 1 *Nous considérons ici la définition classique des réseaux de Petri temporels avec des intervals de tir fermés sur les éléments de* $\mathbb{Q}_{\geq 0}$*. Cependant, il est très*

simple d'étendre cette classe de RdPTs en autorisant les contraintes où les intervals sont ouverts à gauche ou à droite. La sémantique que nous donnons en définition 2 ci-après s'étend de façon directe à ce type d'intervals. Il en est de même de la traduction vers les ATs décrite en section 3.

Un *marquage* M du RdPT est une fonction dans $\mathbb{N}^P$ et si $M \in \mathbb{N}^P$, $M(p_i)$ est le nombre de jetons dans la place p_i. Une transition t est *nouvellement sensibilisée* par un marquage M ssi $M \geq {}^\bullet t. \uparrow enabled(t_k, M, t_i) \in \mathbb{B}$ est vrai si t_k est sensibilisée par le tir de la transition t_i à partir du marquage M, et faux dans le cas contraire. La définition de la sensibilisation est basée sur celle de [BER 91, AUR 00] qui est la plus courante. Dans ce cadre, une transition t_k est *nouvellement sensibilisée* par le tir de t_i à partir du marquage M si "elle n'est pas sensibilisée par $M - {}^\bullet t_i$ et elle est sensibilisée par $M' = M - {}^\bullet t_i + t_i{}^\bullet$" [BER 91]. De plus la transition t_k est également *nouvellement sensibilisée* par son propre tir.

Formellement, cela donne :

$$\uparrow enabled(t_k, M, t_i) = \big(M - {}^\bullet t_i + t_i{}^\bullet \geq {}^\bullet t_k\big) \wedge \big((M - {}^\bullet t_i < {}^\bullet t_k) \vee (t_k = t_i)\big) \quad (1)$$

La sémantique des RdPTs peut être donnée en terme de systèmes de transitions temporisés (TTS) [LAR 95] qui sont des systèmes de transitions classiques avec deux types d'étiquettes : des étiquettes discrètes pour les événements et des étiquettes réelles positives pour l'écoulement du temps.

Définition 2 (Sémantique des RdPTs) *La sémantique d'un RdPT* $\mathcal{T} = (P, T, {}^\bullet(.), (.)^\bullet, M_0, (\alpha, \beta))$ *est un système de transitions temporisé* $S_\mathcal{T} = (Q, q_0, \rightarrow)$ *avec :*

– $Q = \mathbb{N}^P \times (\mathbb{R}_{\geq 0})^n$*, avec* $n =\mid T \mid$*;* $\nu \in (\mathbb{R}_{\geq 0})^n$ *est une* valuation *telle que la valeur* ν_i *représente le temps écoulé depuis le dernier instant où la transition* t_i *a été sensibilisée.*

– $q_0 = (M_0, \overline{0})$*,*

– $\longrightarrow \in Q \times (T \cup \mathbb{R}_{\geq 0}) \times Q$ *est la relation de transition composée des transitions discrètes et continues suivantes :*

- la relation de transition discrète est définie par :

$$\forall t_i \in T, (M, \nu) \xrightarrow{t_i} (M', \nu') \text{ ssi } \begin{cases} M \geq {}^\bullet t_i \wedge M' = M - {}^\bullet t_i + t_i{}^\bullet \\ \alpha(t_i) \leq \nu_i \leq \beta(t_i) \\ \nu'_k = \begin{cases} 0 & \text{si } \uparrow enabled(t_k, M, t_i), \\ \nu_k & \text{sinon.} \end{cases} \end{cases}$$

- la relation de transition continue est définie par :

$$\forall d \in \mathbb{R}_{\geq 0}, (M, \nu) \xrightarrow{\epsilon(d)} (M, \nu') \text{ ssi } \begin{cases} \nu' = \nu + d \\ \forall k \in [1..n], \big(M \geq {}^\bullet t_k \Longrightarrow \nu'_k \leq \beta(t_k)\big) \end{cases}$$

Une exécution *d'un RdPT $\mathcal{T}$ est un chemin dans $S_{\mathcal{T}}$ à partir de q_0. L'ensemble des exécutions de $\mathcal{T}$ est noté $[\![\mathcal{T}]\!]$. Un marquage M est* accessible *dans $\mathcal{T}$ ssi il existe une exécution $(M_0, \overline{0}) \stackrel{\sigma}{\Longrightarrow} (M, \nu)$ dans $[\![\mathcal{T}]\!]$. L'ensemble des* marquages accessibles *de $\mathcal{T}$ est noté $Reach(\mathcal{T})$. Si l'ensemble $Reach(\mathcal{T})$ est fini, le réseau $\mathcal{T}$ est* borné. *Nous notons $(M, \nu) \longrightarrow_e^d (M', \nu')$ pour désigner un pas discret e précédé par un écoulement de temps d'une durée d : $(M, \nu) \stackrel{\epsilon(d)}{\longrightarrow} (M'', \nu'') \stackrel{e}{\rightarrow} (M', \nu')$. Pour une exécution (finie ou infinie) $\sigma = (M_0, \overline{0}) \longrightarrow_{e_0}^{d_0} (M_1, \nu_1) \cdots (M_n, \nu_n) \longrightarrow_{e_n}^{d_n} \cdots, n \in I$, la* durée *de σ est définie par $\sum_{k \in I} d_k$.* □

Cette définition appelle quelques commentaires. Notre sémantique est basée sur la définition classique de [BER 91, AUR 00] pour les RdPTs saufs.

Tout d'abord, les précédentes sémantiques formelles [BER 91, LIL 99, PEZ 99, AUR 00] des RdPTs concernent des RdPTs *saufs*. Notre sémantique est définie pour une classe plus générale des RdPTs et reste cohérente avec la précédente lorsqu'on se restreint aux RdPTs saufs[3]. Nous avons donc donnée une sémantique pour la *multi-sensibilisation* des transitions (dans le cadre *mono-serveur*) qui nous semble la plus simple et la plus appropriée aux applications que nous visons. En effet, plusieurs interprétations peuvent être données à la multisensibilisation des transitions [BER 91, BER 01]. Ceci est important dans la mesure où il n'est pas encore prouvé qu'un RdPT quelconque est équivalent à un RdPT sauf (ce résultat existe pour les RdPs autonomes mais pas pour les RdPTs).

De plus, quelques variations peuvent être trouvées dans la littérature sur les RdPTs concernant le tir des transitions. [PEZ 99] considère deux sémantiques distinctes : i) la *sémantique faible* (Weak Time Semantics : WTS) et ii) la *sémantique forte* (Strong Time Semantics : STS). Concernant la WTS, une transition *peut* être tirée dans son interval de tir alors que dans le cadre STS, une transition *doit* être tirée dans son interval de tir à moins qu'elle ne soit désensibilisée par le tir d'une autre transition. La sémantique la plus classique est STS comme dans [MER 74, BER 91, PEZ 99, AUR 00] et c'est celle que nous avons formellement définie (cependant il est facile d'adapter la définition 2 au cas WTS).

A l'aide de notre sémantique il est facile de définir le caractère *zenon* d'un RdPT. En effet, un système de transitions temporisé est zenon ssi il existe une exécution contenant un nombre infini de transitions discrètes et de durée finie. Un RdPT $\mathcal{T}$ est zenon ssi $S_{\mathcal{T}}$ est zenon. On peut facilement assurer l'absence d'exécutions zenon dans un RdPT : si $\forall i, \alpha(t_i) \neq 0$ alors le RdPT est *non-zenon* et l'exigence que le temps diverge pour toute exécution est vérifiée. De façon générale, si un RdPT est borné, le caractère *zenon* ou *non-zenon* peut être décidée en utilisant l'automate temporisé que nous allons construire dans la section 3 et la procédure décrite dans [HEN 94].

3. Exception faite de la différence avec [LIL 99] concernant la définition des instants de *reset* des transitions nouvellement sensibilisées.

2.2. *Automates temporisés et produit d'automates temporisés*

Les automates temporisés [ALU 94] sont utilisés pour modéliser des systèmes qui combinent des évolutions *discrètes* et *continues*.

Définition 3 (Automate temporisé) *Un* automate temporisé H *est un 6-uplet* (N, l_0, C, A, E, Inv) *avec :*

– N *est un ensemble fini de* localités*,*

– $l_0 \in N$ *est la* localité initiale*,*

– C *est un ensemble fini d'*horloges *(de valeur réelle positive),*

– A *est un ensemble fini d'*actions*,*

– $E \subseteq N \times \mathcal{C}(V) \times A \times 2^C \times N$ *est un ensemble fini d'*arcs*; $e = \langle l, \gamma, a, R, l' \rangle \in E$ représente un arc orienté de la localité l vers la localité l' avec la garde γ, l'étiquette a et l'ensemble de* reset R*.*

– $Inv \in \mathcal{C}(V)^N$ *associe un* invariant *à chaque localité. Nous restreignons les invariants à des conjonctions de termes de la forme* $c \leq r$ *pour* $c \in C$ *et* $r \in \mathbb{N}$*.* □

La sémantique d'un automate temporisé est aussi un système de transitions temporisé.

Définition 4 (Sémantique d'un automate temporisé) *La* sémantique *d'un automate temporisé* $H = (N, l_0, C, A, E, Act, Inv)$ *est un système de transitions temporisé* $S_H = (Q, q_0, \rightarrow)$ *avec* $Q = N \times (\mathbb{R}_{\leq 0})^C$*,* $q_0 = (l_0, \overline{0})$ *est l'état initial et* $\rightarrow$ *est défini par :*

$$(l, v) \xrightarrow{a} (l', v') \quad \text{ssi} \quad \exists\, (l, \gamma, a, R, l') \in E \,/\, \begin{cases} \gamma(v) = \mathit{tt},\ v' = v[R \mapsto 0] \text{ et} \\ Inv(l')(v') = \mathit{tt} \end{cases}$$

$$(l, v) \xrightarrow{\epsilon(t)} (l', v') \quad \text{ssi} \quad \begin{cases} l = l' \qquad v' = v + t \quad \text{et} \\ \forall\, 0 \leq t' \leq t,\ Inv(l)(v + t') = \mathit{tt} \end{cases}$$

Une exécution *d'un automate temporisé* H *est un chemin dans* S_H *à partir de* q_0*. L'ensemble des chemins de* H *est noté* $[\![H]\!]$*.* □

Il est courant de décrire un système comme une composition parallèle d'automates temporisés. Dans ce but, on utilise la notion classique de composition basée sur une *fonction de synchronisation* à la Arnold-Nivat [ARN 94]. Soient H_1, …, H_n n automates temporisés avec $H_i = (N_i, l_{i,0}, C_i, A, E_i, Inv_i)$. Une *fonction de synchronisation* f est une fonction partielle de $(A \cup \{\bullet\})^n \hookrightarrow A$ pour laquelle $\bullet$ est un symbole qui indique qu'un automate n'est pas impliqué dans un pas du système global. Nous notons $(H_1 | \ldots | H_n)_f$ la composition parallèle des H_i synchronisés par f. Les états de $(H_1 | \ldots | H_n)_f$ sont des paires $(\overline{l}, v)$ avec $\overline{l} = (l_1, \ldots, l_n) \in N_1 \times \ldots \times N_n$ et $v = v_1 \cdots v_n$ avec[4] $v_i \in (\mathbb{R}_{\geq 0})^{C_i}$ (Nous supposons que les ensembles d'horloges

4. v_i est la restriction de v sur C_i.

C_i sont disjoints.) La sémantique d'un produit synchronisé d'automates temporisés est un système de transitions temporisé : le produit synchronisé peut évoluer par une transition discrète ou par l'écoulement du temps si tous les composants du produit l'autorisent. Cela est formalisé par la définition suivante :

Définition 5 (Sémantique d'un produit d'automates temporisés) *Soient* $H_1, \ldots, H_n$, n *automates temporisés avec* $H_i = (N_i, l_{i,0}, C_i, A, E_i, Inv_i)$, *et* f *une fonction de synchronisation partielle de* $(A \cup \{\bullet\})^n \hookrightarrow A$. *La sémantique de* $(H_1|\ldots|H_n)_f$ *est un système de transitions temporisé* $S = (Q, q_0, \rightarrow)$ *avec* $Q = N_1 \times \ldots \times N_n \times (\mathbb{R}_{\geq 0})^C$, q_0 *est l'état initial* $((l_{1,0}, \ldots, l_{n,0}), \overline{0})$ *et* $\rightarrow$ *est défini par :*

– $(\overline{l}, v) \xrightarrow{b} (\overline{l'}, v')$ *ssi il existe* $(a_1, \ldots, a_n) \in (A \cup \{\bullet\})^n$ *s.t.* $f(a_1, \ldots, a_n) = b$ *et pour tout* i *nous avons :*

. *Si* $a_i = \bullet$, *alors* $l'_i = l_i$ *et* $v'_i = v_i$,

. *Si* $a_i \in A$, *alors* $(l_i, v_i) \xrightarrow{a_i} (l'_i, v'_i)$.

– $(\overline{l}, v) \xrightarrow{\epsilon(t)} (\overline{l}, v')$ *ssi* $\forall\, i \in [1..n]$, *nous avons* $(l_i, v_i) \xrightarrow{\epsilon(t)} (l_i, v'_i)$ □

Nous pouvons à partir du produit de n automates temporisés construire (syntaxiquement) un nouvel automate temporisé dont la sémantique est celle décrite ci-dessus [LAR 95]. Pour la suite, nous considérerons un produit $(H_1|\ldots|H_n)_f$ comme un nouvel automate temporisé dont la sémantique est temporellement bisimilaire[5] à la sémantique du produit (Cf. définition 5).

3. Traduction structurelle des RdPTs vers les ATs

Dans cette section, nous construisons un produit synchronisé d'automates temporisés à partir d'un RdPT tel que leurs comportements soient temporellement bisimilaires.

3.1. *Traduction structurelle des RdPTs vers les ATs*

Soit un réseau de Petri temporel $\mathcal{T} = (P, T, {}^\bullet(.), (.)^\bullet, M_0, (\alpha, \beta))$ avec $P = \{p_1, \cdots, p_m\}$ et $T = \{t_1, \cdots, t_n\}$.

Automate temporisé associée à une transition du RdPT.

Nous définissons un automate temporisé $\mathcal{A}_i$ pour chaque transition t_i de T (voir Fig. 1). Cet automate temporisé possède une horloge x_i. Les états de l'automate $\mathcal{A}_i$ donne l'état de la transition t_i : dans l'état t la transition est sensibilisée ; dans l'état $\bar{t}$ elle n'est pas sensibilisée ; dans l'état *Firing* elle est en train d'être tirée. L'état initial de chaque $\mathcal{A}_i$ dépend du marquage initial M_0 du RdPT qui est traduit. Si $M_0 \geq {}^\bullet t_i$

5. Cf. section 3 pour la bisimilarité temporelle.

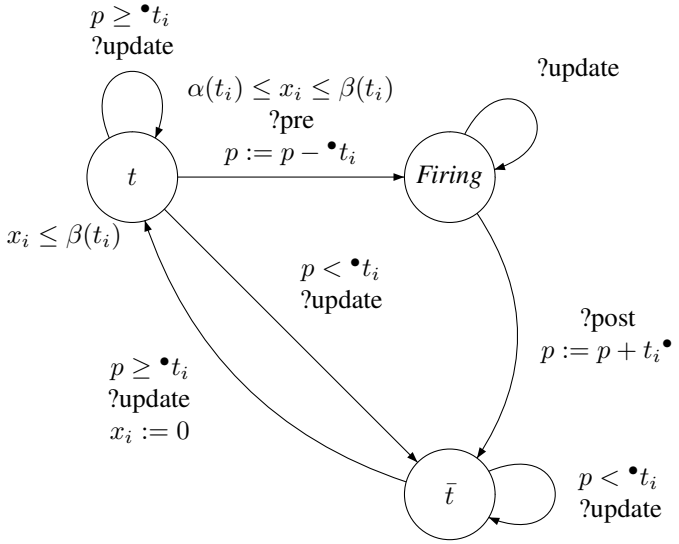

Figure 1. *Automate $\mathcal{A}_i$ correspondant à la transition t_i*

alors l'état initial est t et dans le cas contraire c'est $\bar{t}$. Cet automate met à jour un tableau d'entiers p (tel que $p[i]$ correspond au nombre de jetons dans la place p_i) qui est partagé par tous les $\mathcal{A}_i$'s et le superviseur défini ci-après.

Le superviseur.

Le superviseur SU est décrit Fig. 2. Les localités 1 à 3 indicées par un c sont supposées *urgentes* ou *committed*[6] ce qui signifie que le temps ne peut pas s'écouler lorsque l'on est dans ces localités. L'état initial du superviseur est 0.

Nous définissons la fonction de synchronisation f à $n+1$ paramètres par (n est le nombre de transitions du RdPT à traduire) :

– $f(!pre, \bullet, \cdots, ?pre, \bullet, \cdots) = pre_i$ if $?pre$ est le $(i+1)$ème argument et tous les autres arguments sont $\bullet$,

– $f(!post, \bullet, \cdots, ?post, \bullet, \cdots) = post_i$ if $?post$ st le $(i+1)$ème argument et tous les autres arguments sont $\bullet$,

– $f(!update, ?update, \cdots, ?update) = update$.

Nous notons $\Delta(\mathcal{T}) = (SU \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_n)_f$ l'automate temporisé associé au RdPT $\mathcal{T}$. Nous allons prouver dans la section suivante que la sémantique de $\Delta(\mathcal{T})$ est *équivalente* à la sémantique de $\mathcal{T}$. Pour cela nous devons associer les états de $\mathcal{T}$ aux états de $\Delta(\mathcal{T})$ et nous définissons l'équivalence suivante :

6. dans SU, les localités *committed* ou *urgentes* sont équivalentes ; ce type de localité peut être simulé par l'ajout d'une horloge x qui est mise à 0 en entrant dans la localité et en ajoutant l'invariant $x = 0$ à la localité ; pour notre problème, il suffit de considérer que dans une localité urgente le temps ne peut s'écouler.

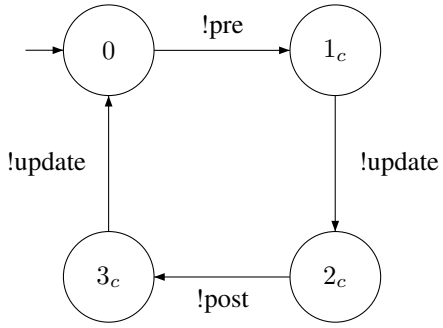

Figure 2. *Automate du superviseur SU*

Définition 6 (Équivalence entre états) *Soient* (M, ν) *un état de* $S_{\mathcal{T}}$ *et* $((s, p), \overline{q}, v)$ *une configuration*[7] *de* $S_{\Delta(\mathcal{T})}$. *Nous avons*

$$(M,\nu) \approx ((s,p),\overline{q},v) \text{ ssi } \begin{cases} s = 0, \\ \forall i \in [1..m], \quad p[i] = M(p_i), \\ \forall k \in [1..n], \quad q_k = \begin{cases} t \text{ si } M \geq {}^\bullet t_k, \\ \bar{t} \text{ sinon} \end{cases} \\ \forall k \in [1..n], \quad \nu_k = v_k \end{cases}$$ □

3.2. *Correction de la traduction*

Nous prouvons maintenant que notre traduction préserve le comportement du RdPT initial dans le sens où la sémantique du RdPT et sa traduction sont temporellement bisimilaires. Soient un RdPT $\mathcal{T}$ et $S_{\mathcal{T}} = (Q, q_0, \rightarrow)$ sa sémantique. Soient $\mathcal{A}_i$ l'automate associé à la transition t_i de $\mathcal{T}$ comme décrit par la Fig. 1, SU l'automate superviseur de la Fig. 2 et f la fonction de synchronisation définie précédemment. La sémantique de $\Delta(\mathcal{T}) = (SU \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_n)_f$ est le système de transitions temporisé $S_{\Delta(\mathcal{T})} = (Q_{\Delta(\mathcal{T})}, q_0^{\Delta(\mathcal{T})}, \rightarrow)$.

7. (s, p) est l'état de SU où p est le vecteur donnant le nombre de jetons par place ; $\overline{q}$ donne l'état discret de $\mathcal{A}_1 \times \cdots \times \mathcal{A}_n$ (donc $\bar{q}_i$ est l'état de $\mathcal{A}_i$) et v les valeurs des horloges $x_i, i \in [1..n]$ de chaque $\mathcal{A}_i$.

Théorème 1 (Bisimilarité temporelle) *Pour tout état* (M,ν) *de* $S_{\mathcal{T}}$ *et* $((0,p),\bar{q},v)$ *de* $S_{\Delta(\mathcal{T})}$ *tel que* $(M,\nu) \approx ((0,p),\bar{q},v)$ *nous avons :*

$$(M,\nu) \xrightarrow{t_i} (M',\nu') \quad \textit{ssi} \quad \begin{cases} ((0,p),\bar{q},v) \stackrel{w_i}{\Longrightarrow} ((0,p'),\bar{q}',v') \textit{ avec} \\ w_i = pre_i.update.post_i.update \textit{ et} \\ (M',\nu') \approx ((0,p'),\bar{q}',v') \end{cases} \tag{2}$$

$$(M,\nu) \xrightarrow{\epsilon(d)} (M',\nu') \quad \textit{ssi} \quad \begin{cases} ((0,p),\bar{q},v) \xrightarrow{\epsilon(d)} ((0,p'),\bar{q}',v') \textit{ et} \\ (M',\nu') \approx ((0,p'),\bar{q}',v') \end{cases} \tag{3}$$

□

La preuve du théorème 1 est donnée en annexe A, page 19.

Nous pouvons maintenant poser un corollaire qui nous permettra dans la section suivante de faire du *model-checking* de propriétés TCTL sur les RdPTs. Nous notons $\Delta((M,\nu)) = ((0,p),\bar{q},v)$ si $(M,\nu) \approx ((0,p),\bar{q},v)$, pour une transition $t_i \in T$, $\Delta(t_i) = pre_i.update.post_i.update$ et $\Delta(\epsilon(d)) = \epsilon(d)$. Remarquons que Δ est une bijection et que nous pouvons donc parler de Δ^{-1} si nécessaire. Nous étendons Δ aux transitions de $\Delta(\mathcal{T})$: $\Delta((M,\nu) \xrightarrow{e} (M',\nu')) = \Delta((M,\nu)) \xrightarrow{\Delta(e)} \Delta((M',\nu'))$ avec $e \in T \cup \mathbb{R}_{\geq 0}$ (étant donné que $\Delta(t_i)$ est un mot, cette transition est en quatre pas). De même, nous pouvons étendre Δ aux exécution : si $\rho \in [\![\mathcal{T}]\!]$ nous notons $\Delta(\rho)$ l'exécution correspondante dans $[\![\Delta(\mathcal{T})]\!]$. Notons que Δ^{-1} est seulement défini pour les exécutions σ de $[\![\Delta(\mathcal{T})]\!]$ dont le dernier état est de la forme $((0,p),\bar{q},v)$ (c'est à dire pour lequel le superviseur est dans l'état 0). Nous notons cette propriété *last*$(\sigma) \models SU.0$.

Corollaire 1 $\big(\rho \in [\![\mathcal{T}]\!] \wedge \sigma = \Delta(\rho)\big)$ *ssi* $\big(\sigma \in [\![\Delta(\mathcal{T})]\!] \wedge$ *last*$(\sigma) \models SU.0\big)$. □

Preuve 1 *La preuve est une conséquence directe du théorème 1. Il suffit de remarquer que toutes les exécutions de* $\Delta(\mathcal{T})$ *sont de la forme*

$$\sigma = (s_0,v_0) \Longrightarrow^{t_1}_{w_1} (s_1,v_1) \cdots \Longrightarrow^{t_n}_{w_n} (s_n,v_n)$$

avec $w_i = pre_i.update.post_i.update$ *et par conséquent, en utilisant le théorème 1, si le dernier état satisfait* $last(\sigma) \models SU.0$*, il existe une exécution correspondante* ρ *dans* $\mathcal{T}$ *telle que* $\sigma = \Delta(\rho)$. □

Ainsi toutes les exécutions dans $\Delta(\mathcal{T})$ se terminant avec $SU.0$ ont une exécution correspondante dans $\mathcal{T}$. Cette propriété sera utilisée dans la section 4 lorsque nous nous intéresserons au problème de la vérification de propriétés écrites en TCTL pour les RdPTs.

4. Model-Checking de TCTL pour les RdPTs

Nous pouvons maintenant définir TCTL [ALU 93, HEN 94] pour les RdPTs. La seule différence avec les versions de [ALU 93, HEN 94] pour les automates temporisés est que les propositions atomiques habituellement associées aux états deviennent des propriétés de marquage. Pour les applications pratiques avec des *model-checkers* existant, nous supposerons que les RdPTs sont bornés.

TCTL pour RdPTs.

Définition 7 (TCTL pour RdPTs) *Soit un RdPT avec n places, et m transitions de l'ensemble $T = \{t_1, t_2, \cdots, t_m\}$. La logique temporelle TPN-TCTL est définie par :*

$$\textit{TPN-TCTL} ::= \mathbf{M} \bowtie \bar{V} \mid \mathbf{false} \mid t_k + c \leq t_j + d \mid \neg\varphi \mid \varphi \rightarrow \psi \mid \varphi\, \exists\mathcal{U}_{\bowtie c}\, \psi \mid \varphi\, \forall\mathcal{U}_{\bowtie c}\, \psi$$

où $\mathbf{M}$ *et* $\mathbf{false}$ *sont des* mots-clés*,* $\varphi, \psi \in$ *TPN-TCTL,* $t_k, t_j \in T$*,* $c, d \in \mathbb{Z}$*,* $\bar{V} \in (\mathbb{Q} \cup \{\infty\})^n$ *et*[8] $\bowtie \in \{<, \leq, =, >, \geq\}$. □

Intuitivement, la signification de $\mathbf{M} \bowtie \bar{V}$ est que le vecteur *marquage courant* est en relation $\bowtie$ avec $\bar{V}$. Le sens des autres opérateurs est le sens usuel. Nous utilisons les relations usuelles $\mathbf{true} = \neg\mathbf{false}$, $\Diamond_{\bowtie c}\phi = \mathbf{true}\, \exists\mathcal{U}_{\bowtie c}\, \phi$ and $\Box_{\bowtie c} = \neg\Diamond_{\bowtie c}\neg\phi$.

La sémantique de TPN-TCTL est définie sur les systèmes de transitions temporisés. Soit le RdPT $\mathcal{T} = (P, T, {}^\bullet(.), (.)^\bullet, M_0, (\alpha, \beta))$ et $S_\mathcal{T} = (Q, q_0, \rightarrow)$ la sémantique de $\mathcal{T}$. La satisfaction d'une formule φ de TPN-TCTL pour un état (M, ν) est donnée par le tableau 1.

Le RdPT $\mathcal{T}$ satisfait la formule φ de TPN-TCTL, ce qui est noté $\mathcal{T} \models \varphi$ ssi l'état initial de $S_\mathcal{T}$ satisfait φ c.à.d. $(M_0, \nu_0) \models \varphi$.

Quelques exemples de formules de TPN-TCTL.

On considère le RdPT $\mathcal{T}$ de la Fig. 3.

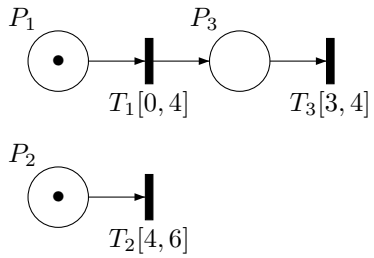

Figure 3. *Le RdPT $\mathcal{T}$*

8. l'utilisation de ∞ dans $\bar{V}$ nous permet des comparaisons du type $M(p_1) \leq 2 \wedge M(p_2) \geq 3$ en écrivant $\mathbf{M} \leq (2, \infty) \wedge \mathbf{M} \geq (0, 3)$.

$$
\begin{array}{lll}
(M,\nu) \models \mathbf{M} \bowtie \bar{V} & \text{ssi} & M \bowtie \bar{V} \\
(M,\nu) \not\models \mathbf{false} & & \\
(M,\nu) \models t_k + c \leq t_j + d & \text{ssi} & \nu_k + c \leq \nu_j + d \\
(M,\nu) \models \neg\varphi & \text{ssi} & (M,\nu) \not\models \varphi \\
(M,\nu) \models \varphi \rightarrow \psi & \text{ssi} & (M,\nu) \models \varphi \text{ implique } (M,\nu) \models \psi \\
(M,\nu) \models \varphi \vee \psi & \text{ssi} & (M,\nu) \models \varphi \text{ ou } (M,\nu) \models \psi \\
(M,\nu) \models \varphi\, \exists\mathcal{U}_{\bowtie c}\, \psi & \text{ssi} & \exists \sigma = (s_0,\nu_0) \stackrel{d_1}{\longrightarrow}_{a_1} (s_1,\nu_1) \cdots \stackrel{d_n}{\longrightarrow}_{a_n} (s_n,\nu_n) \in [\![\mathcal{T}]\!] \text{ t.q.} \\
 & & \begin{cases} (s_0,\nu_0) = (M,\nu) \\ \forall i \in [1..n], \forall d \in [0,d_i), (s_i, \nu_i + d) \models \varphi \textit{ et} \\ \left(\sum_{i=1}^{n} d_i\right) \bowtie c \textit{ et } (s_n,v_n) \models \psi \end{cases} \\
(M,\nu) \models \varphi\, \forall\mathcal{U}_{\bowtie c}\, \psi & \text{ssi} & \forall \sigma = (s_0,\nu_0) \stackrel{d_1}{\longrightarrow}_{a_1} (s_1,\nu_1) \cdots \stackrel{d_n}{\longrightarrow}_{a_n} (s_n,\nu_n) \in [\![\mathcal{T}]\!] \text{ t.q.} \\
 & & \begin{cases} (s_0,\nu_0) = (M,\nu) \\ \forall i \in [1..n], \forall d \in [0,d_i), (s_i, \nu_i + d) \models \varphi \textit{ et} \\ \left(\sum_{i=1}^{n} d_i\right) \bowtie c \textit{ et } (s_n,v_n) \models \psi \end{cases}
\end{array}
$$

Tableau 1. *Sémantique de TPN-TCTL*

La formule $\mathbf{M} \geq (1,2,0)$ est vraie dans tous les états de ce RdPT où le marquage est tel que le nombre de jetons dans P_1 est supérieur ou égal à 1 et le nombre de jetons dans P_2 est supérieur ou égal à 2. Ainsi $\Box_{\leq 3}(\mathbf{M} \geq (1,2,0))$ est vraie dans tous les états pour lesquels on atteint forcément un marquage $\mathbf{M} \geq (1,2,0)$ en moins de trois unités de temps. Si l'on veut indiquer que cela est vrai pour tous les états accessibles on peut écrire $\mathcal{T} \models \Box_{\leq 3}(\mathbf{M} \geq (1,2,0))$. Si l'on souhaite vérifier un propriété P du type "il est possible d'atteindre un état où la transition T_2 est tirable et T_1 ne l'est pas" on peut utiliser les horloges liées aux transitions et procéder comme suit :

1) soit π_1 (resp. π_2) la propriété de marquage indiquant que la transition t_1 (resp. t_2) est sensibilisée : $\pi_1 \equiv \mathbf{M} \geq (0,1,0)$ et $\pi_2 \equiv \mathbf{M} \geq (0,0,1)$

2) t_1 et t_2 désignent les horloges associées aux conditions de tir des transitions ;

3) une formule TPN-TCTL exprimant la propriété P peut être :

$$\mathsf{tt}\, \exists\mathcal{U}_{\geq 0}\big((\neg\pi_1 \vee (t_1 < 4)\big) \wedge \big(\pi_2 \wedge (3 \leq t_2 \leq 4)\big)$$

Nous allons voir que grâce au corollaire 1 vérifier une propriété TPN-TCTL sur un RdPT revient à vérifier une propriété TCTL sur l'automate temporisé correspondant.

Model-checking de TPN-TCTL.

Supposons que nous ayons à vérifier une formule φ sur le RdPT $\mathcal{T}$ de la Fig. 3. Notre méthode consiste à utiliser l'automate temporisé équivalent $\Delta(\mathcal{T})$ défini section 3. Par exemple, supposons que l'on souhaite vérifier $\mathcal{T} \models \Box_{\leq 3}(\mathbf{M} \geq (1,2,0))$. En fait, cela est équivalent à vérifier $\Box_{\leq 3}(SU.0 \rightarrow (p[1] \geq 1 \wedge p[2] \geq 2))$ sur l'automate temporisé équivalent. Notons que $\Diamond_{\leq 3}(\mathbf{M} \geq (1,2,0))$ se réduit à $\Diamond_{\leq 3}(SU.0 \wedge (p[1] \geq 1 \wedge p[2] \geq 2))$. Nous pouvons maintenant définir la traduction d'une formule TPN-TCTL en une formule TCTL pour automate temporisé.

Définition 8 (Traduction de TPN-TCTL vers TCTL) *Soit φ une formule TPN-TCTL. La traduction $\Delta(\varphi)$ de φ est définie (par induction) par :*

$$\begin{aligned}
\Delta(\mathbf{M} \bowtie \bar{V}) &= \bigwedge_{i=1}^{n} (p[i] \bowtie \bar{V}_i) \\
\Delta(\mathbf{false}) &= \mathbf{false} \\
\Delta(t_k + c \bowtie t_j + d) &= = x_k + c \bowtie x_j + d \\
\Delta(\neg\varphi) &= = \neg\Delta(\varphi) \\
\Delta(\varphi \rightarrow \psi) &= SU.0 \wedge (\Delta(\varphi) \rightarrow \Delta(\psi)) \\
\Delta(\varphi \vee \psi) &= SU.0 \wedge (\Delta(\varphi) \vee \Delta(\psi)) \\
\Delta(\varphi \, \exists\mathcal{U}_{\bowtie c} \, \psi) &= (SU.0 \rightarrow \Delta(\varphi)) \, \exists\mathcal{U}_{\bowtie c} \, (SU.0 \wedge \Delta(\psi)) \\
\Delta(\varphi \, \forall\mathcal{U}_{\bowtie c} \, \psi) &= (SU.0 \rightarrow \Delta(\varphi)) \, \forall\mathcal{U}_{\bowtie c} \, (SU.0 \wedge \Delta(\psi))
\end{aligned}$$

$SU.0$ signifie que le superviseur est dans l'état 0 et les horloges x_k sont les horloges des automates $\mathcal{A}_k$ associés à chaque transition t_k (Cf. Fig 1). □

Nous pouvons maintenant énoncer le théorème suivant :

Théorème 2 *Soient le RdPT $\mathcal{T}$ et $\Delta(\mathcal{T})$ l'automate temporisé équivalent. Soient (M, ν) un état de $S_{\mathcal{T}}$ et $((s, p), \bar{q}, v) = \Delta((M, \nu))$ l'état équivalent de $S_{\Delta(\mathcal{T})}$ (c.à.d. $(M, \nu) \approx ((s, p), \bar{q}, v)$). Alors*

$$\forall \varphi \in \textit{TPN-TCTL} \quad (M, \nu) \models \varphi \textit{ ssi } ((s, p), \bar{q}, v) \models \Delta(\varphi)$$ □

La preuve du théorème 2 est donnée en annexe B, page 20.

Comme corollaire nous obtenons que TCTL est décidable pour les RdPTs bornés :

Corollaire 2 *Le model-checking de TPN-TCTL est décidable pour les RdPTs bornés.* □

Exemple de model-checking.

Nous donnons un exemple d'un RdPT et de sa traduction en automate temporisé. Cette traduction est implémentée dans le logiciel Romeo [ROM 00] qui produit l'automate au format d'entrée d'UPPAAL. Il est alors possible de vérifier des propriétés temps-réel avec UPPAAL sur l'automate équivalent. On prend le RdPT de la Fig. 3

Les automates temporisés correspondant à la transition T_0 et au superviseur sont donnés en Fig. 4 et 5 dans l'annexe C.

Par exemple nous pouvons vérifier la propriété
`A[](( SU.0 and T0.x>4) imply p[1]<1)` ou encore `E<>(T2.Firing and ((not`

`T1.t) or T1.x<4))`. La première signifie que tous les états atteints après 4 unités de temps ont un marquage tel que P_1 n'a pas de jeton. La seconde signifie qu'il est possible de tirer T_2 alors que T_1 n'est pas tirable.

5. Conclusion

Dans cet article, nous avons donné une traduction structurelle d'un réseau de Petri Temporel vers un produit synchronisé d'automates temporisés. Un RdPT $\mathcal{T}$ et sa traduction en automate temporisé $\Delta(\mathcal{T})$ sont temporellement bisimilaires. Cela a pour conséquence que les RdPTs bornés héritent de beaucoup de résultats théoriques sur les automates temporisés. La classe des RdPTs peut être étendue en autorisant des contraintes strictes (intervals ouverts, semi-ouverts ou fermés) pour spécifier les dates de tir des transitions ; pour cette classe étendue, les résultats suivants découlent de notre traduction en AT et du théorème de bisimilarité (théorème 1) :

– l'atteignabilité et plus généralement le model-checking de TCTL sont décidables pour les RdPTs bornés ; de plus les algorithmes efficaces utilisés dans UPPAAL [LAR 97, PET 00] et KRONOS [YOV 97] sont exacts pour les RdPTs (Cf. les récents résultats [BOU 03] de P. Bouyer) ;

– le caractère zénon est décidable pour les RdPTs bornés ;

– la synthèse de contrôleurs (il existe un algorithme [ASA 94] pour les ATs) est possible pour les RdPTs bornés : le problème de l'existence d'un contrôleur (sous forme d'automate temporisé) évitant un ensemble de marquage (ou avec des conditions plus générales comme définies dans [ASA 94]) est décidable et la synthèse est possible dans le cas où un contrôleur existe[9].

– enfin, notre traduction étant structurelle, il est possible d'utiliser un model-checker pour trouver des conditions suffisantes de non-bornitude du réseau.

D'un point de vue pratique, tous les outils disponibles pour l'analyse des automates temporisés s'appliquent aux RdPTs. De plus il est possible de spécifier un système en utilisant conjointement les RdPTs et les ATs. La vérification d'un RdPT en utilisant directement notre traduction risque de ne pas être très efficace : en effet, l'AT équivalent au RdPT a un nombre d'horloges égal au nombre de transitions du RdPT.

Les complexités des différents problèmes (atteignabilité, model-checking de TCTL, synthèse de contrôleur) pour les RdPTs ne peuvent être déduits de la traduction précédente. Cependant, les travaux en cours que nous poursuivons actuellement consistent à traduire un automate temporisé en réseau de Petri temporel (toujours en préservant la bisimilarité temporelle). Si cette traduction inverse est possible, nous obtiendrons de nouveaux résultats concernant la complexité des problèmes de vérification pour les RdPTs (actuellement aucun résultat de ce type n'est connu).

9. pour obtenir le résultat indiquant qu'un contrôleur de type RdPT peut être synthétisé il faudrait prouver qu'il est possible de construire un RdPT à partir de l'automate temporisé.

6. Bibliographie

[ABD 93] ABDULLA P., JONSSON B., « Verifying Programs with Unreliable Channels », *Proceedings, Eighth Annual IEEE Symposium on Logic in Computer Science*, IEEE Computer Society Press, 1993, p. 160–170.

[ABD 01] ABDULLA P. A., NYLÉN A., « Timed Petri Nets and BQOs », *22nd International Conference on Application and Theory of Petri Nets (ICATPN'01)*, vol. 2075 de *Lecture Notes in Computer Science*, Newcastle upon Tyne, United Kingdom, june 2001, Springer-Verlag, p. 53–72.

[ALU 93] ALUR R., COURCOUBETIS C., DILL D., « Model-Checking in Dense Real-time », *Information and Computation*, vol. 104, n° 1, 1993, p. 2–34.

[ALU 94] ALUR R., DILL D. L., « A theory of timed automata », *Theoretical Computer Science*, vol. 126, n° 2, 1994, p. 183–235.

[ALU 99] ALUR, FIX, HENZINGER, « Event-Clock Automata : A Determinizable Class of Timed Automata », *TCS : Theoretical Computer Science*, vol. 211, 1999.

[ARN 94] ARNOLD A., *Finite Transition System*, Prentice Hall, 1994.

[ASA 94] ASARIN E., MALER O., PNUELI A., « Symbolic Controller Synthesis for Discrete and Timed Systems », *Hybrid Systems*, 1994, p. 1-20.

[AUR 00] AURA T., LILIUS J., « A causal semantics for time Petri nets », *Theoretical Computer Science*, vol. 243, 2000.

[BEN 96] BENGTSSON J., GRIFFIOEN W. O. D., KRISTOFFERSEN K. J., LARSEN K. G., LARSSON F., PETTERSSON P., YI W., « Verification of an Audio Protocol with Bus Collision Using UPPAAL », *Proc. of the 8th International Conference on Computer-Aided Verification, LNCS 1102*, 1996, p. 244-256.

[BER 83] BERTHOMIEU B., MENASCHE M., « An enumerative approach for analyzing time Petri nets », MASON R. E. A., Ed., *Information Processing : proceedings of the IFIP congress 1983*, vol. 9 de *IFIP congress series*, Elsevier Science Publishers, Amsterdam, 1983, p. 41–46.

[BER 91] BERTHOMIEU B., DIAZ M., « Modeling and verification of time dependent systems using time Petri nets », *IEEE transactions on software engineering*, vol. 17, n° 3, 1991, p. 259–273.

[BER 01] BERTHOMIEU B., « La méthode des classes d'états pour l'analyse des réseaux temporels », *3e congrès Modélisation des Systèmes Réactifs (MSR'2001)*, Toulouse, France, october 2001, Hermes, p. 275–290.

[BOR 98] BORNOT S., SIFAKIS J., TRIPAKIS S., « Modeling Urgency in Timed Systems », *Lecture Notes in Computer Science*, vol. 1536, 1998, p. 103–129, Springer-Verlag.

[BOU 00] BOUYER P., DUFOURD C., FLEURY E., PETIT A., « Are Timed Automata Updatable ? », *Proc. 12th Int. Conf. Computer Aided Verification (CAV'2000), Chicago, IL, USA, July 2000*, vol. 1855 de *Lecture Notes in Computer Science*, Springer, 2000, p. 464–479.

[BOU 03] BOUYER P., « Untameable timed automata ! », *Proc. of the 20*th *Annual Symposium on Theoretical Aspects of Computer Science (STACS'2003)*, n° 2607 Lecture Notes in Computer Science, Springer–Verlag, février 2003.

[DIA 94] DIAZ M., SENAC P., « Time stream Petri nets : a model for timed multimedia information », *Lecture Notes in Computer Science*, vol. 815, 1994, p. 219– ? ?

[Fix 94] FIX, L., ALUR, R., HENZINGER, T.A., « A Determinizable Class of Timed Automata », DAVID L. DILL, Ed., *sixth International Conference on Computer-Aided Verification CAV*, vol. 818 de *Lecture Notes in Computer Science*, Standford, California, USA, juin 1994, Springer-Verlag, p. 1–13.

[HEN 94] HENZINGER T. A., NICOLLIN X., SIFAKIS J., YOVINE S., « Symbolic model checking for real-time systems », *Information and Computation*, vol. 111, n° 2, 1994, p. 193–244.

[KHA 96] KHASA W., DENAT J.-P., COLLART-DUTILLEUL S., « P-Time Petri Nets for manufacturing systems », *International Workshop on Discrete Event Systems, WODES'96*, Edinburgh (U.K.), august 1996, p. 94–102.

[LAR 95] LARSEN K. G., PETTERSSON P., YI W., « Model-Checking for Real-Time Systems », *Fundamentals of Computation Theory*, 1995, p. 62-88.

[LAR 97] LARSEN K. G., PETTERSSON P., YI W., « UPPAAL in a Nutshell », *Journal of Software Tools for Technology Transfer*, vol. 1, n° 1/2, 1997, p. 134-152.

[LAR 98] LAROUSSINIE F., LARSEN K. G., « CMC : A tool for compositional model-checking of real-time systems », *Proc. IFIP Joint Int. Conf. Formal Description Techniques & Protocol Specification, Testing, and Verification (FORTE-PSTV'98)*, Kluwer Academic Publishers, 1998, p. 439-456.

[LIL 99] LILIUS J., « Efficient State Space Search for Time Petri Nets », *Electronic Notes in Theoretical Computer Science*, vol. 18, 1999, Elsevier Science B.V.

[LIN 98] LINDAHL M., PETTERSSON P., YI W., « Formal Design and Analysis of a Gear-Box Controller », *Proc. of the 4*th *Workshop on Tools and Algorithms for the Construction and Analysis of Systems*, n° 1384 Lecture Notes in Computer Science, Springer–Verlag, mars 1998, p. 281–297.

[MAL 96] MALER O., YOVINE S., « Hardware timing verification using KRONOS », *Proc. 7th Israeli Conference on Computer Systems and Software Engineering*, Herzliya, Israel, juin 1996.

[MER 74] MERLIN P. M., « A study of the recoverability of computing systems », PhD thesis, Department of Information and Computer Science, University of California, Irvine, CA, 1974.

[PET 00] PETTERSSON P., LARSEN. K. G., « UPPAAL2k », *Bulletin of the European Association for Theoretical Computer Science*, vol. 70, 2000, p. 40–44.

[PEZ 99] PEZZÉ M., YOUNG M., « Time Petri Nets : A Primer Introduction », Tutorial presented at the Multi-Workshop on Formal Methods in Performance Evaluation and Applications, Zaragoza, Spain, september 1999, Avaiblable at www.elet.polimi.it/people/pezze.

[POP 91] POPOVA L., « On time Petri nets », *Journal Information Processing and Cybernetics, EIK*, vol. 27, n° 4, 1991, p. 227–244.

[RAM 74] RAMCHANDANI C., « Analysis of asynchronous concurrent systems by timed Petri nets », PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1974, Project MAC Report MAC-TR-120.

[ROM 00] ROMEO, « $http : //www.irccyn.ec - nantes.fr/irccyn/d/fr/equipes/TempsReel/logs\%$ guilf, *A tool for Time Petri Nets Analysis*, , 2000.

[SAV 01] SAVA A. T., ALLA H., « Commande par supervision des systèmes à évènements discrets temporisés », *Modélisation des systèmes réactifs (MSR 2001)*, , 2001, p. 71–86.

[SIF 96] SIFAKIS J., YOVINE S., « Compositional Specification of Timed Systems (Extended Abstract) », *13th Annual Symposium on Theoretical Aspects of Computer Science*, vol. 1046 de *lncs*, Grenoble, France, 22–24 février 1996, Springer, p. 347–359.

[TOU 97] TOUSSAINT J., SIMONOT-LION F., THOMESSE J.-P., « Time constraint verifications methods based time Petri nets », *6th Workshop on Future Trends in Distributed Computing Systems (FTDCS'97)*, Tunis, Tunisia, 1997, p. 262–267.

[YOV 97] YOVINE S., « Kronos : A Verification Tool for real-Time Systems », *Journal of Software Tools for Technology Transfer*, vol. 1, n° 1/2, 1997, p. 123-133.

A. Preuve du théorème 1

Preuve 2 *Nous prouvons d'abord la formule 2. Supposons* $(M,\nu) \approx ((0,p),\bar{q},v)$. *Alors, étant donné que* t_i *peut être tirée à partir de* (M,ν) *nous avons : (i)* $M \geq {}^\bullet t_i$ *(ii)* $\alpha(t_i) \leq \nu_i \leq \beta(t_i)$ *(iii)* $M' = M - {}^\bullet t_i + t_i{}^\bullet$ *(iv)* $\nu'_k = 0$ *si* $\uparrow$*enabled*(t_k, M, t_i) *et* $\nu'_k = \nu_k$ *sinon. A partir de (i), de (ii) et de l'équivalence entre états, nous en déduisons que* $\bar{q}_i = t$ *et* $\alpha(t_i) \leq v_i \leq \beta(t_i)$. *Par conséquent,* $?pre$ *peut être franchie dans* $\mathcal{A}_i$. *Dans l'état* 0 *du superviseur,* $!pre$ *est la seule transition possible. Étant donné que la fonction de synchronisation* f *permet* $(!pre, \bullet, \cdots, ?pre, \cdots, \bullet)$ *l'action globale* pre_i *est possible. Après cette action* $\Delta(\mathcal{T})$ *atteint l'état* $((1,p_1),\bar{q}_1,v_1)$ *avec* $\bar{q}_{1k} = \bar{q}_k, \forall k \neq i$ *et* $\bar{q}_{1i}$ = *Firing. Ainsi,* $p_1 = p - {}^\bullet t_i$ *et* $v_1 = v$.

Maintenant, la seule transition possible pour le superviseur qui est dans l'état 1 *est la transition* update *pour laquelle tous les* $\mathcal{A}_i$ *se synchronisent (voir* f*). A partir de* $((1,p_1),\bar{q}_1,v_1)$ *nous atteignons* $((2,p_2),\bar{q}_2,v_2)$ *avec* $p_2 = p_1$, $v_2 = v_1$. *Pour* $k \neq i$, $\bar{q}_{2k}$ = *if* $p_1 \geq {}^\bullet t_k$ *et* $\bar{q}_{2k} = \bar{t}$ *sinon et* $\bar{q}_{2i}$ = *Firing.*

La transition globale suivante est nécessairement la transition $post_i$ *qui conduit à* $((3,p_3),\bar{q}_3,v_3)$ *avec* $p_3 = p_2 + t_i{}^\bullet$, $v_3 = v_2$ *et* $\bar{q}_{3k} = \bar{q}_{2k}, \forall k \neq i$ *et* $\bar{q}_{3i} = \bar{t}$.

A partir de ce dernier état, seule la transition update *est possible et conduit à* $((0,p_4),\bar{q}_4,v_4)$ *avec* $p_4 = p_3$, v_4 *et* q_4 *donné par :* $\bar{q}_{4k} = t$ *si* $p_3 \geq {}^\bullet t_k$ *et* $\bar{t}$ *sinon.* $v_{4k} = 0$ *si* $\bar{q}_{3k} = \bar{t}$ *et* $\bar{q}_{4k} = t$ *et* $v_{4k} = v_{1k}$ *sinon. Notons simplement que* $\bar{q}_{3k} = \bar{t}$ *ssi* $p - {}^\bullet t_i < {}^\bullet t_k$ *et* $\bar{q}_{4k} = t$ *ssi* $p - {}^\bullet t_i + t_i{}^\bullet \geq {}^\bullet t_k$. *Cela implique que* $v_{4k} = 0$ *ssi* $\uparrow$ *enabled*(t_k, p, t_i) *et avec (iv) cela donne* $\nu'_k = v_{4k}$. *Étant donné que* $p_4 = p_3 = p_2 + t_i{}^\bullet = p_1 - {}^\bullet t_i + t_i{}^\bullet = p - {}^\bullet t_i + t_i{}^\bullet$ *et que (iii) nous avons* $\forall i \in [1..m], M'(p_i) = p_4[i]$. *Nous concluons alors que* $((0,p_4),\bar{q}_4,v_4) \approx (M',\nu')$.

La réciproque de la formule 2 est évidente en suivant le même cheminement que précédemment .

Nous nous intéressons maintenant à la formule 3. D'après la sémantique des RdPTs, une transition continue $(M,\nu) \xrightarrow{\epsilon(d)} (M',\nu')$ *est possible ssi* $\nu = \nu' + d$ *et* $\forall k \in [1..n], (M \geq {}^\bullet t_k \implies \nu'_k \leq \beta(t_k))$. *D'après l'équivalence* $(M,\nu) \approx ((0,p),\bar{q},v)$, *si* $M \geq {}^\bullet t_k$ *alors* $\bar{q}_k = t$ *et l'évolution continue pour* $\mathcal{A}_k$ *est contrainte par l'invariant* $x_k \leq \beta(t_k)$ *; dans le cas contraire,* $\bar{q}_k = \bar{t}$ *et l'évolution continue*

est non contrainte pour $\mathcal{A}_k$. Le superviseur n'étant soumis à aucune contrainte (invariant) dans l'état 0*, le résultat est prouvé.* □

B. Preuve du théorème 2

Preuve 3 *La preuve est faite par induction structurelle sur la formule TPN-TCTL. Les cas* $\mathbf{M} \bowtie \bar{V}$, **false**, $t_k + c \leq t_j + d$, $\neg\varphi$ *et* $\varphi \rightarrow \psi$, $\varphi \vee \psi$ *sont évidents. Nous donnons la preuve complète pour* $\varphi\, \exists\mathcal{U}_{\bowtie c}\, \psi$ *(la même preuve peut être établie pour* $\varphi\, \forall\mathcal{U}_{\bowtie c}\, \psi$*).*

Condition nécessaire.

Supposons $(M, \nu) \models \varphi\, \exists\mathcal{U}_{\bowtie c}\, \psi$*. Alors par définition, il existe une exécution* ρ *dans* $[\![\mathcal{T}]\!]$ *telle que :*

$$\rho = (s_0, \nu_0) \longrightarrow_{a_1}^{d_1} (s_1, \nu_1) \cdots \longrightarrow_{a_n}^{d_n} (s_n, \nu_n)$$

et $(s_0, \nu_0) = (M, \nu)$, $\sum_{i=1}^{n} d_i \bowtie c$, $\forall i \in [1..n], \forall d \in [0, d_i), (s_i, \nu_i + d) \models \varphi$ *et* $(s_n, \nu_n) \models \psi$*. D'après le corollaire 1, nous en déduisons qu'il existe une exécution* $\sigma = \Delta(\rho)$ *dans* $[\![S_{\Delta(\mathcal{T})}]\!]$ *telle que*

$$\sigma = ((l_0, p_0), \bar{q}_0, v_0)) \Longrightarrow_{w_1}^{d_1} ((l_1, p_1), \bar{q}_1, v_1)) \cdots \Longrightarrow_{w_n}^{d_n} ((l_n, p_n), \bar{q}_n, v_n))$$

et $\forall i \in [1..n], ((l_i, p_i), \bar{q}_i, v_i)) \approx (s_i, \nu_i)$ *(cela impose que* $l_i = 0$*.)*

Étant donné que $(s_n, \nu_n) \approx ((l_n, p_n), \bar{q}_n, v_n))$*, avec l'hypothèse d'induction sur* ψ*, nous pouvons admettre que* $(s_n, \nu_n) \models \psi$ *ssi* $((l_n, p_n), \bar{q}_n, v_n)) \models \Delta(\psi)$ *et conclure que* $((l_n, p_n), \bar{q}_n, v_n)) \models \Delta(\psi)$*. De plus comme* $l_n = 0$ *cela* $((l_n, p_n), \bar{q}_n, v_n)) \models SU.0 \wedge \Delta(\psi)$*. Cela revient à prouver que tous les états intermédiaires satisfont* $SU.0 \rightarrow \Delta(\varphi)$*. Notons que tous les états intermédiaires dans* σ *ne vérifiant pas* $SU.0$ *entre* $((l_i, p_i), \bar{q}_i, v_i)$ *et* $(((l_{i+1}, p_{i+1}), q_{i+1}^{-}, v_{i+1}))$ *vérifient* $SU.0 \rightarrow \Delta(\psi)$*. Nous avons alors juste à prouver que les états intermédiaires qui vérifient* $SU.0$*, vérifient aussi* $\Delta(\varphi)$*. Comme* $\forall i \in [1..n], ((l_i, p_i), \bar{q}_i, v_i)) \approx (s_i, \nu_i)$*, avec l'hypothèse d'induction sur* φ*, nous avons* $\forall i \in [1..n], ((l_i, p_i), \bar{q}_i, v_i)) \models \Delta(\varphi)$*. De plus, en appliquant à nouveau le théorème 1,* $\forall d \in [0, d_i), ((l_i, p_i), \bar{q}_i, v_i + d)) \approx (s_i, \nu_i + d)$ *et en appliquant à nouveau hypothèse d'induction, on obtient* $\forall d \in [0, d_i), ((l_i, p_i), \bar{q}_i, v_i + d)) \models \Delta(\varphi)$*. Par conséquent* $((l_0, p_0), \bar{q}_0, v_0)) \models (SU.0 \rightarrow \varphi)\, \exists\mathcal{U}_{\bowtie c}\, (SU.0 \wedge \psi)$*.*

Condition suffisante.

Supposons $((l_0, p_0), \bar{q}_0, v_0)) \models (SU.0 \rightarrow \Delta(\varphi))\, \exists\mathcal{U}_{\bowtie c}\, (SU.0 \wedge \Delta(\psi))$*. Alors il existe une exécution*

$$\sigma = ((l_0, p_0), \bar{q}_0, v_0)) \Longrightarrow_{w_1}^{d_1} ((l_1, p_1), \bar{q}_1, v_1)) \cdots \Longrightarrow_{w_n}^{d_n} ((l_n, p_n), \bar{q}_n, v_n))$$

avec $((l_n, p_n), \bar{q}_n, v_n)) \models SU.0 \wedge \Delta(\psi)$ *et* $\forall i \in [1..n], \forall d \in [0, d_i), ((l_i, p_i), \bar{q}_i, v_i)) \models (SU.0 \rightarrow \Delta(\varphi))$*. Etant donné que* $((l_n, p_n), \bar{q}_n, v_n)) \models SU.0$*, nous pouvons utiliser le corollaire 1 et nous savons qu'il existe une exécution dans* $[\![\mathcal{T}]\!]$

$$\rho = \Delta^{-1}(\sigma) = (s_0, \nu_0) \rightarrow_{a_1}^{d_1} (s_1, \nu_1) \cdots \rightarrow_{a_n}^{d_n} (s_n, \nu_n)$$

avec $\forall i \in [1..n], ((l_i, p_i), \bar{q}_i, v_i)) \approx (s_i, \nu_i)$. *L'hypothèse d'induction sur* $SU.0 \wedge \Delta(\psi)$ *combinée à* $((l_n, p_n), \bar{q}_n, v_n)) \models \mathcal{S}.0 \wedge \Delta(\psi)$ *implique* $(s_n, \nu_n) \models \psi$. *Pour tous les états intermédiaires de* ρ *nous appliquons aussi l'hypothèse d'induction : chaque* $((l_i, p_i), \bar{q}_i, v_i))$ *est équivalent à* (s_i, ν_i) *et tous les états* $(s_i, \nu_i + d), d \in [0, d_i)$ *satisfont* φ. *Par conséquent* $(s_0, \nu_0) \models \varphi \exists \mathcal{U}_{\bowtie c} \psi$. □

C. Automates UPPAAL

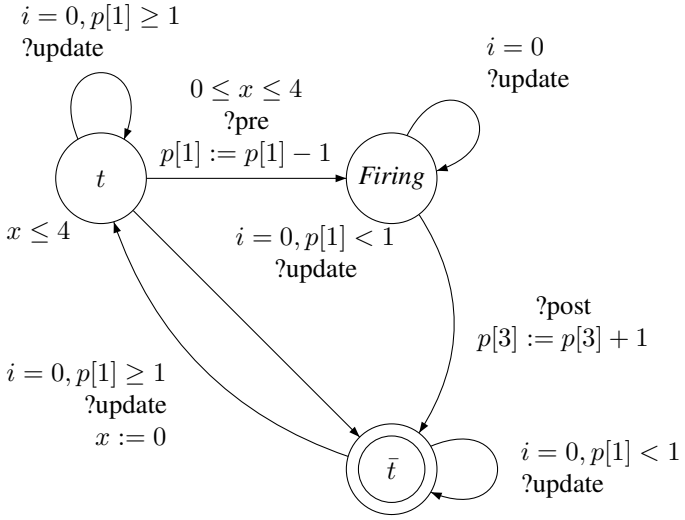

Figure 4. *Automate* $\mathcal{A}_1$ *associé à la transition* T_1

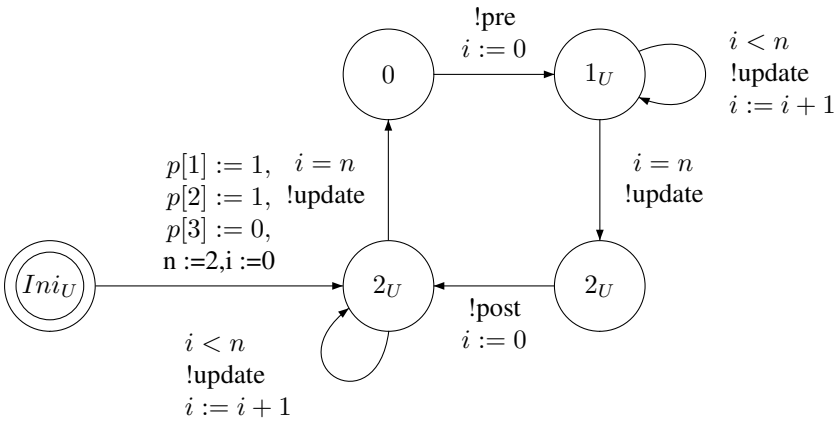

Figure 5. *Automate du superviseur SU pour l'exemple de la Fig. 3*