Séquentialisation du comportement de réseaux de Petri temporisés

Jan Komenda [1], **Sébastien Lahaye** [2], **Jean-Louis Boimond** [2]

1. *Académie Tchèque des Sciences Žižkova 22 – 616 62 Brno, Rép. Tchèque*
   komenda@ipm.cz
2. *LISA – 62 avenue Notre Dame du Lac – Angers 49000*
   jean-louis.boimond@univ-angers.fr,sebastien.lahaye@univ-angers.fr

RÉSUMÉ. *Ce papier s'intéresse à la séquentialisation des séries formelles à coefficients dans le semi-anneau* $(\mathbb{R} \cup \{-\infty\}, \max, +)$ *qui représentent le comportement de réseaux de Petri temporisés bornés. Des méthodes existent pour modéliser les réseaux de Petri temporisés saufs par des automates (max,+). Les automates résultants sont presque toujours non déterministes et les procédures de déterminisation existantes ne peuvent que très rarement leur être appliquées. Ceci proscrit l'utilisation de certains résultats, notamment pour l'évaluation de performances et la commande. Nous présentons un semi-algorithme pour obtenir des automates (max,+) déterministes en se basant sur la sémantique des réseaux de Petri temporisés. Les automates obtenus peuvent être infinis, mais nous proposons une condition suffisante pour assurer que le semi-algorithme termine et conduise à un automate (max,+) déterministe fini. Lorsque le réseau considéré ne peut être séquentialisé (car la procédure ne termine pas), nous proposons de restreindre le comportement logique du réseau de sorte à assurer la séquentialisation.*

ABSTRACT. *In this paper we are interested in sequentialization of formal power series with coefficients in the semiring* $(\mathbb{R} \cup \{-\infty\}, \max, +)$ *which represent the behavior of timed Petri nets. Several approaches make it possible to derive nondeterministic (max,+) automata modeling safe timed Petri nets. Their nondeterminism is a serious drawback since determinism is a crucial property for numerous results on (max,+) automata (in particular, for applications to performance evaluation and control) and existing procedures for determinization succeed only for restrictive classes of (max,+) automata. We present a natural semi-algorithm for determinization of behaviors based on the semantics of timed Petri nets. The resulting deterministic (max,+)-automata are often infinite, but a sufficient condition is proposed to ensure that the semi-algorithm terminates and leads to a finite state deterministic (max,+)-automaton. Moreover, if the net cannot be sequentialized we propose a restriction of its logical behavior so that the sufficient condition becomes satisfied for the restricted net.*

MOTS-CLÉS : *réseaux de Petri temporisés, automate (max,+), politique de course.*

KEYWORDS: *timed Petri nets, (max,+) automata, race policy.*

DOI:10.3166/JESA.47.139-154

1. Introduction

Les réseaux de Petri temporisés introduits dans (Ramchandani, 1973) sont devenus un outil de référence dans la modélisation des systèmes concurrents dans lesquels la temporisation des événements est déterministe. Toutefois, d'un point de vue mathématique, les modèles basés sur les automates (et leurs extensions temporisées) semblent être mieux adaptés pour le calcul des comportements (séries formelles) et pour des applications à la commande supervisée. Les automates (max,+), c'est-à-dire les automates à multiplicités dans le semi-anneau idempotent $(\mathbb{R} \cup \{-\infty\}, max, +)$ ont été étudiés dans (Gaubert, 1995 ; Gaubert, Mairesse, 1999) où il a notamment été montré que le comportement d'un réseau de Petri temporisé sauf peut être décrit par un automate (max,+) particulier, appelé automate de type tas. Il s'avère que les automates (max,+) ainsi dérivés sont typiquement non déterministes (excepté dans les cas simples sans concurrence tels que les graphes d'états saufs). Ces modèles ont été exploités notamment pour l'évaluation de performances (Gaubert, 1995) et l'ordonnancement (Houssin, 2011) de systèmes. Leur utilisation pour la commande supervisée s'avère délicate du fait que seuls les systèmes décrits par des automates (max,+) déterministes permettent l'obtention de contrôleurs rationnels, voir (Komenda *et al.*, 2009 ; Badouel *et al.*, 2011), et les procédures de déterminisation échouent pour ces modèles. Rappelons que seule une sous-classe des automates (max,+) est déterminisable (c'est-à-dire, pouvant être représenté par un automate (max,+) déterministe équivalent), et que les algorithmes établis jusqu'ici ne s'appliquent qu'à des classes restrictives.

Nous considérons dans ce papier les réseaux de Petri temporisés bornés (le marquage de chaque place est borné par une constante). Nous proposons une procédure qui associe à chacun de ces réseaux un automate (max,+) déterministe. L'automate résultant est parfois infini du fait de la non-terminaison de la procédure (ce constat était prévisible notamment parce que tout automate (max,+) ne peut être déterminisé). La procédure est basée sur un dépliage du graphe d'atteignabilité du réseau logique sous-jacent en utilisant les vecteurs des horloges associées aux transitions, ceci en accord avec la sémantique des réseaux de Petri temporisés. Après avoir expliqué la différence entre la politique dite de la course et celle de la préselection, nous proposons une condition suffisante qui assure la terminaison de la procédure de déterminisation basée sur une notion forte de vivacité des transitions. Finalement, nous présentons une technique basée sur la restriction du comportement logique (langage du réseau de Petri) permettant de restreindre le langage du réseau (typiquement en limitant les choix possibles grâces aux places de conflit) de telle façon que le réseau modifié puisse être séquentialisé.

Le papier est organisé comme suit. Dans la section suivante, des définitions et résultats préliminaires sont brièvement énoncés. Dans la section 3, nous proposons un semi-algorithme de déterminisation. Cette procédure pouvant ne pas terminer, une condition suffisante de terminaison est proposée dans la section 4. La section 5 est dédiée à une technique qui permet d'imposer la condition suffisante en restreignant le comportement logique du réseau (*i.e.*, le support de la série formelle qui décrit le

comportement du réseau). Une conclusion est proposée dans la section 6, ainsi que des extensions et applications futures de notre approche.

2. Préliminaires

Les concepts et résultats sur les dioïdes, les automates (max, +) et les réseaux de Petri utilisés par la suite sont brièvement rappelés dans cette section. Le lecteur est invité à consulter les références (Baccelli *et al.*, 1992), (Gaubert, 1995) et (David, Alla, 2010) pour disposer de plus de détails.

DÉFINITION 1 (dioïde). — *Un* dioïde *est un* semi-anneau $(\mathcal{D}, \oplus, \otimes)$ *dans lequel l'addition* $\oplus$ *est idempotente. L'addition (resp., la multiplication* $\otimes$*) a un élément nul* ε *(resp., identité* e*).*

Un exemple classique de dioïde est l'ensemble $\mathbb{R} \cup \{-\infty\}$ doté du maximum pour loi additive et de l'addition conventionnelle comme multiplication, noté $\mathbb{R}_{\max}$, avec $\varepsilon = -\infty$ et $e = 0$.
L'ensemble des matrices $n \times n$ à coefficients dans $\mathbb{R}_{\max}$ doté de l'addition et de la multiplication matricielles définies de façon usuelle est également un dioïde, noté $\mathbb{R}_{\max}^{n \times n}$. L'élément nul (neutre pour l'addition) est la matrice, notée ε_n et exclusivement composée de ε $(= -\infty)$. La matrice identité (neutre pour la multiplication) est la matrice, notée I_n, composée de e $(= 0)$ sur la diagonale et de ε $(= -\infty)$ partout ailleurs. Pour nombre réel a, on note $a^+ = \max\{a, 0\}$ et $a^- = -\min\{a, 0\}$. Etant donné un vecteur $\vec{c} \in \mathbb{R}_{\max}^{1 \times n}$ et un scalaire $d \in \mathbb{R}_{\max}$, on note $\vec{c} + d$ la somme (usuelle) point à point de $\vec{c}$ et de $d = (d, d, \ldots, d)$.

Si A est un ensemble (*alphabet*) fini, le monoïde libre sur A est défini par l'ensemble A^* des mots finis composés de lettres appartenant à A. Un *mot* $w \in A^*$ peut être écrit comme une séquence $w = a_1 a_2 \ldots a_p$ où a_1, a_2, …, $a_p \in A$ et p est un entier naturel. Une relation d'ordre, notée $\preceq$, peut être définie : pour $v, w \in A^*$, on a $v \preceq w$ s'il existe $u, u' \in A^*$ tel que $w = uvu'$. On appelle alors v un *sous mot* de w. Enfin, pour un mot $v \in A^*$, la longueur de v est notée $\mid v \mid$.

Un automate dont les transitions sont pondérées (à multiplicités) dans le semi-anneau $\mathbb{R}_{\max}$ est appelé automate (max,+).

DÉFINITION 2 (automate (max,+)). — *Un automate (max,+)* G *est un 4-uplet* (Q, A, α, μ) *où :*

– Q *et* A *sont des ensembles finis respectivement d'états et d'événements ;*

– $\alpha \in \mathbb{R}_{\max}^{1 \times |Q|}$ *est tel que* $\alpha_q \neq \varepsilon$ *si* q *est un état initial ;*

– $\mu : A^* \to \mathbb{R}_{\max}^{|Q| \times |Q|}$ *est un morphisme, spécifié par les matrices* $\mu(a) \in \mathbb{R}_{\max}^{|Q| \times |Q|}$, $a \in A$ *et la relation suivante pour toute chaîne* $w = a_1 \ldots a_n$ *:*

$$\mu(w) = \mu(a_1 \ldots a_n) = \mu(a_1) \ldots \mu(a_n).$$

Un coefficient $[\mu(a)]_{qq'} \neq \varepsilon$ signifie qu'à partir de l'état q, une transition vers l'état q' est possible si l'événement a survient.

Cette définition est légèrement différente de celle donnée dans (Gaubert, 1995) car on ne distingue pas les états finaux (tous les états sont considérés finaux comme pour les modèles de type tas (Gaubert, Mairesse, 1999)). Ainsi, le langage sous-jacent des automates (max,+) [1] sera toujours préfixe clos.

EXEMPLE 3. — La figure 1 est un exemple de représentation graphique associée à un automate (max,+) :

– les nœuds correspondent aux états $q \in Q$;

– une flèche existe d'un état $q \in Q$ vers un état q' s'il existe un événement $a \in A$ tel que $[\mu(a)]_{qq'} \neq \varepsilon$: l'arc est labellisé par $a/[\mu(a)]_{qq'}$ et représente la transition d'états quand l'événement a se produit. La valeur de $[\mu(a)]_{qq'}$ est interprétée comme la durée associée à l'événement a (à savoir, le temps d'activation avant que l'événement a puisse se produire) ;

– une flèche sans état de départ distingue un état initial, et le poids associé est un délai initial.

Dans cet exemple, on a $Q = \{q_1, q_2, q_3, q_4\}$, $A = \{a, b\}$, $\alpha = \begin{pmatrix} 0 & \cdot & 0 & 0 \end{pmatrix}$, et

$$\mu(a) = \begin{pmatrix} \cdot & 2 & \cdot & 2 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & 1 \\ \cdot & 3 & \cdot & 3 \end{pmatrix}, \mu(b) = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ 2 & \cdot & 2 & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 0 \end{pmatrix}.$$

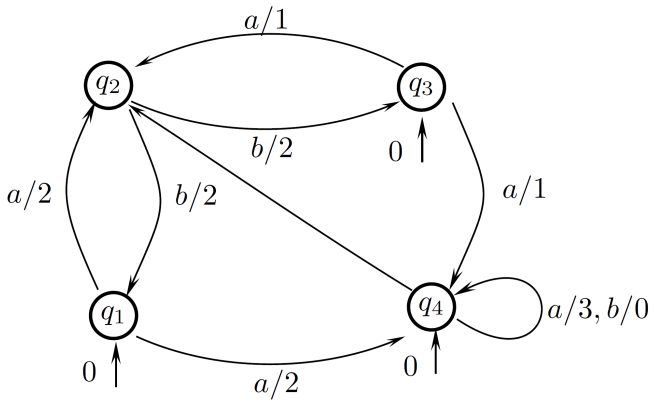

Figure 1. Un automate (max,+)

□

1. Supports des séries de puissance formelle reconnues.

Afin de décrire l'évolution dynamique d'un automate (max,+), le vecteur $x_G(w) \in \mathbb{R}_{\max}^{1\times|Q|}$, avec $w \in A^*$, est défini par :

$$x_G(w) = \alpha\mu(w)\ . \tag{1}$$

Un élément $[x_G(w)]_q$ est interprété comme la date à laquelle l'état q est atteint à l'issue de la séquence w partant d'un état initial (avec pour convention que $[x_G(w)]_q = \varepsilon$ si l'état q n'est pas atteignable à partir d'un état initial à travers la séquence w). Les éléments de x_G sont des *dateurs généralisés*, on a :
$x_G(wa) = x_G(w)\mu(a)$ avec $x_G(\epsilon) = \alpha$. Finalement, le dateur généralisé $y_G(w) = \oplus_q (x_G)_q(w)$ définit comme la somme (maximum) des composantes de x_G est connu comme la sortie ou comportement de G. En effet, y_G est souvent représenté sous forme d'une série formelle sous jacente: $y_G = \bigoplus_{w\in A^*} y_G(w)w$.

DÉFINITION 4 (Réseau de Petri). — *Un réseau de Petri $\mathcal{G}$ est un 4-uplet $(\mathcal{P}, \mathcal{T}, \mathcal{F}, M_0)$, où $\mathcal{P}$ est un ensemble fini de places, $\mathcal{T}$ est un ensemble fini de transitions, $\mathcal{F} \subseteq (\mathcal{P}\times\mathcal{T}) \cup (\mathcal{T}\times\mathcal{P})$ est une relation entre les places et les transitions, $M_0 : \mathcal{P} \to \mathbb{N}$ définit le marquage initial des places.*

Le marquage évolue selon les règles suivantes :

1. La transition a est *sensibilisée* pour un marquage M (propriété notée $M \xrightarrow{a}$) s'il existe au moins un jeton dans chacune de ses places d'entrée. $M \not\xrightarrow{a}$ signifie que la transition a n'est pas sensibilisée pour M.

2. Une transition a sensibilisée peut être franchie/tirée. Le tir de a transforme M en M' en ôtant un jeton dans chacune de ses places d'entrée et en ajoutant un jeton dans chacune de ses places de sortie (transformation décrite par la notation $M \xrightarrow{a} M'$).

Un mot $w = a_1 a_2 \dots a_n \in \mathcal{T}^*$ correspond à une *séquence de franchissement* partant d'un marquage M_0 s'il existe une séquence de marquages $M_1 M_2 \dots M_n$ telle que la transition a_i est sensibilisée pour M_{i-1} et que son franchissement transforme M_{i-1} en M_i, soit la notation $M_0 \xrightarrow{a_1} M_1 \xrightarrow{a_2} \dots M_{i-1} \xrightarrow{a_i} M_i$. Le *langage* $L(\mathcal{G})$ d'un réseau de Petri $\mathcal{G}$ est égale à l'ensemble des séquences de franchissement $w \in \mathcal{T}^*$ possibles depuis le marquage initial M_0.

Un réseau de Petri est dit *sauf* (resp., *m-borné*) si pour tous les marquages accessibles, chaque place contient au plus un jeton (resp., au plus m jetons).

Le graphe d'atteignabilité d'un réseau de Petri borné $\mathcal{G}$ est l'automate $Reach(\mathcal{G}) = (\mathcal{M}, M_0, \mathcal{T}, t_r)$, où $\mathcal{M}$ est l'ensemble des marquages atteignables (fini de par l'hypothèse de bornitude), $M_0 \in \mathcal{M}$ est le marquage initial, l'ensemble des événements est l'ensemble des transitions (typiquement labellisées par des symboles alphabétiques en bijection avec $\mathcal{T}$) et la fonction partielle de transition $t_r : \mathcal{M}\times\mathcal{T} \to \mathcal{M}$ définie pour $M, M' \in \mathcal{M},\ t \in \mathcal{T}$ par $t_r(M,t) = M'$ ssi $M \xrightarrow{t} M'$. L'automate $Reach(\mathcal{G})$ est déterministe puisque chaque transition a un label différent.

Pour une transition $a \in \mathcal{T}$, ${}^\bullet a$ (resp., $a^\bullet$) représente l'ensemble de ses places d'entrée (resp., de sortie). Le réseau de Petri est un *graphe d'état* si ces ensembles sont des singletons pour toutes les transitions. De façon duale, pour une place $p \in \mathcal{P}$,

${}^\bullet p$ (resp., $p^\bullet$) représente l'ensemble des transitions en amont (resp.,en aval) de p. Le réseau de Petri est un *graphe d'événements* si ces ensembles sont des singletons pour toutes les places du réseau. On considère les réseaux de Petri temporisés pour lesquels des durées finies de franchissement sont associées aux transitions.

DÉFINITION 5 (Réseau de Petri temporisé). — *Un réseau de Petri temporisé est un uplet $(\mathcal{G}, \tau)$, où $\tau = (\tau_a)_{a \in \mathcal{T}}$ est un vecteur représentant les temps associés aux transitions, τ_a est interprété comme le temps minimal devant s'écouler entre l'instant où la transition a est sensibilisée et l'instant où elle peut être franchie.*

On appele *graphe d'événements temporisé* (GET) un réseau de Petri temporisé dont le réseau logique est un graphe d'événements. Plusieurs hypothèses sur le fonctionnement des réseaux de Petri temporisés sont faites par la suite :

– un jeton du marquage initial est supposé être arrivé dans le réseau de Petri à l'instant 0 ;

– si une place a plusieurs transitions de sortie (ce qui est une situation potentielle de *conflit*), il est nécessaire, pour chaque jeton dans cette place, de décider quelle transition doit être tirée. Ils existent deux politiques pour arbitrer les conflits. Nous préférons dans ce papier politique de course qui est basée sur les considérations temporelles: la transition dont le franchissement peut se produire le plus tôt est tirée (une définition plus précise de la sémantique est donnée dans la section suivante).

– quand une transition est franchie, elle l'est dès que possible, ce qui correspond à une vitesse maximale (au plus tôt) de fonctionnement.

Un réseau de Petri est représenté par un graphe orienté biparti comme par exemple sur la figure 2. Les deux types de nœuds sont les places dans $\mathcal{P}$ et les transitions dans $\mathcal{T}$, lesquelles sont habituellement représentées par respectivement des cercles et des barres. Un élément de $\mathcal{F}$ est représenté par un arc orienté d'une place vers une transition ou d'une transition vers une place. Le marquage dans une place q est représenté par $M(q)$ jetons dans la place.

3. Procédure de séquentialisation pour les réseaux de Petri temporisés

Nous proposons dans cette section un semi-algorithme permettant la déterminisation des comportements des réseaux de Petri temporisés bornés. Rappelons que la déterminisation est importante pour aborder les problèmes de commande des systèmes temporisés qu'ils soient modélisés par des automates (max,+) ou par des réseaux de Petri temporisés dont le comportement est régi par des séries formelles. En effet, seuls les systèmes dont les séries sont non ambiguës permettent l'existence de contrôleurs rationnels (Komenda *et al.*, 2009 ; Badouel *et al.*, 2011). Notre approche consiste à chercher des conditions assurant l'existence d'un automate (max,+) ayant le même comportement (la même série formelle (max,+)) que le réseau de Petri. La sémantique considérée pour les réseaux temporisés bornés est celle appelée *politique de la course*.

La sémantique des réseaux de Petri temporisés peut être définie en termes de systèmes de transitions temporisés (STT) où les transitions discrètes alternent avec le

passage du temps, lesquels constituent un modèle sémantique de référence pour une large classe de systèmes temporisés, notamment les réseaux de Petri temporels et les automates temporisés. Cette sémantique est basée sur la notion d'état étendu, lequel comprend l'état discret (c'est-à-dire, le marquage du réseau logique sous-jacent) et un vecteur avec les valeurs des horloges associées aux transitions.

L'horloge associée à la transition $t \in \mathcal{T}$ est notée c_t et elle mesure le temp écoulé depuis sa dernière remise à zéro. L'état étendu est alors représenté par la paire $(M, \vec{c})$ composée du marquage $M \in \mathcal{M}$ et du vecteur des horloges $\vec{c} = (c_t)_{t \in \mathcal{T}}$. L'état initial est donné par $(M_0, \vec{0})$, où $\vec{0}$ est un vecteur composé de zéros.

Rappelons qu'une transition $t \in \mathcal{T}$ est dite sensibilisée par un marquage M (ce qui est notée par $M \xrightarrow{t}$) si $\forall p \in {}^\bullet t : M(p) \geq 1$. L'ensemble des transitions sensibilisées par un marquage M est noté $En(M) = \{t \in \mathcal{T} : \ M \xrightarrow{t}\}$.

L'ensemble des horloges remises à zéro à l'issue du franchissement d'une transition sont celles correspondant aux transitions qui sont *nouvellement sensibilisées*. Étant donné un marquage courant $M \in \mathcal{M}$ qui sensibilise la transition $\bar{t} \in \mathcal{T}$ et $M \xrightarrow{\bar{t}} M'$, une transition est nouvellement sensibilisée par le marquage M' si elle est sensibilisée par M' alors qu'elle ne l'était pas par le marquage intermédiaire, c.-à-d., le marquage M' auquel ont été retirés les jetons des places situées en amont ayant contribué au franchissement de la transition, ceci sans ajouter les jetons dans les places situées en aval. Formellement, le marquage intermédiaire est défini par :

$$M''(p) = \begin{cases} M(p) & \text{si } t \notin p^\bullet, \\ M(p) - 1 & \text{sinon, } \textit{i.e.}, \ t \in p^\bullet. \end{cases}$$

La transition $t' \in \mathcal{T}$ est nouvellement sensibilisée par le marquage M', qui résulte du franchissement de la transition précédente $M \xrightarrow{\bar{t}} M'$, ce que nous notons par $t' \in NewEn(M, \bar{t}, M')$, si $M' \xrightarrow{t'}$ et soit $t' = \bar{t}$ ou bien $M'' \not\xrightarrow{t'}$.

Afin de simplifier la présentation, nous introduisons la notation suivante : pour un sous-ensemble des transitions $S \subseteq \mathcal{T}$ et un vecteur de valeurs d'horloges $\vec{c} = (c_t)_{t \in \mathcal{T}}$, $Reset(c, S)$ s'interprète comme de nouvelles valeurs d'horloges, où les composantes correspondant aux transitions de S sont remises à 0 et les autres composantes restent intactes, soit :

$$Reset(\vec{c}, S)_t = \begin{cases} c_t & \text{si } t \notin S \\ 0 & \text{si } t \in S \end{cases}$$

Il y a deux types de transitions dans les STT, cf. (Lime *et al.*, 2012) : les transitions discrètes et les transitions temporelles correspondant simplement à l'écoulement du temps.

Dans le cas particulier des réseaux temporisés, où aucune borne supérieure pour l'intervalle de durée de franchissement n'est spécifié, et où le fonctionnement est à vitesse maximale (les transitions sont franchies dès qu'elles sont sensibilisées), les transitions continues et discrètes sont définies comme suit.

On considère que le temps ne peut plus s'écouler si une transition sensibilisée peut être franchie instantanément : $(M, \vec{c}) \xrightarrow{t} (M', \vec{c}+d)$ pour $d \in \mathbb{R}^+$ si $\forall t \in En(M)$ et $\forall 0 < d' \leq d : c_t + d' \leq \tau_t$. Ceci correspond bien à la politique de la course pour la résolution des conflits quand il y a plusieurs transitions sensibilisées dans le réseau. Au lieu de condition d'écoulement du temps avec des délais non déterministes, nous avons dans le cas des réseaux temporisés, un écoulement du temps bien déterminé qui peut donc être exactement calculé. Notamment, la transition continue de STT est :

$$(M, \vec{c}) \xrightarrow{a} (M, \vec{c}+d)$$

avec $d = \min_{a \in En(M)} \tau_a - c_a$, ce qui s'interprète comme le temps qui peut s'écouler jusqu'au franchissement de la première transition sensibilisée par le marquage courant M, puisque la valeur de son horloge c_a atteint la valeur de temporisation τ_a de la transition. La transition dont l'horloge a atteint en premier la valeur de temporisation est donc retenue pour le franchissement tandis que les autres transitions sensibilisées peuvent attendre. Dans le cas particulier de conflit entre plusieurs transitions en aval d'une place suffisamment marquée : il y a plusieurs transitions qui « gagnent » la course (*i.e.,* et on considère alors tous les franchissements possibles.

Dans le cadre des STT, les transitions discrètes, choisies parmi celles qui sont sensibilisées comme décrit ci-dessus, sont définies par :
$(M, \vec{c}) \xrightarrow{a} (M', \vec{c'})$ si :

- il existe une transition $M \xrightarrow{a} M'$ dans le réseau logique ;
- $c_a = \tau_a$;
- $c' = Reset(\vec{c}, NewEn(M', t, M))$.

L'élément intéressant dans le cas des réseaux temporisés est que les transitions discrètes et continues peuvent être combinées sous la forme d'une transition discrète avec une multiplicité qui correspond à l'écoulement du temps entre deux transitions discrètes consécutives, ce qui coïncide exactement avec la sémantique des automates (max,+) en termes de STT. Depuis un état, on peut combiner l'écoulement du temps avec la transition discrète suivante au sein d'une unique transition à multiplicité.

Plus précisément, on définit directement $(M, \vec{c}) \xrightarrow{a} (M', \vec{c}+d) = (M', \vec{c'})$ avec $d = \min_{a \in En(M)}(\tau_a - c_a)$ et

$$\vec{c'} = Reset(\vec{c}+d, NewEn(M', t, M)) \tag{2}$$

comme une transition d'automate (max,+)

$$(M, \vec{c}) \xrightarrow{a/d} (M', \vec{c'}).$$

La durée d'un événement a dans l'automate (max,+) déterministe obtenu est appelée, par la suite, la durée observée, ou séquentialisée, de a. L'état initial de l'automate (max,+) est $(M_0, \vec{0})$. On appelle la serie formelle (max,+) obtenu comme le comportement de cet automate la série formelle du réseau $\mathcal{G}$.

Illustrons cette approche à l'aide d'un exemple.

EXEMPLE 6. — Prenons l'exemple simple du producteur-consommateur modélisé par le GET de la figure 2. L'automate (max,+) déterministe correspondant est représenté sur la figure 3.

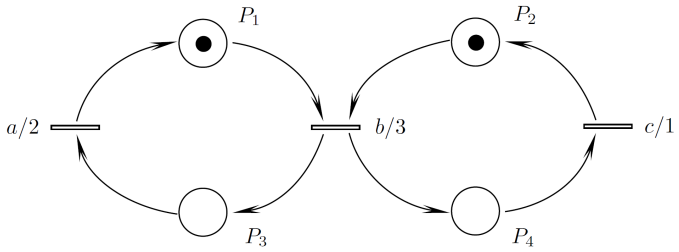

Figure 2. Réseau de Petri temporisé $\mathcal{G}$

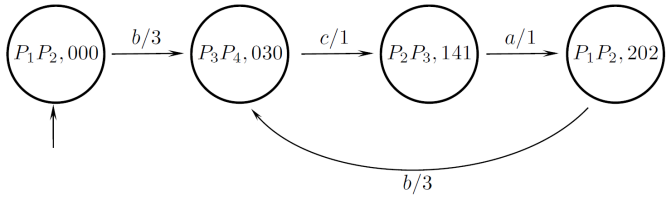

Figure 3. Automate (max,+) déterministe G correspondant à $\mathcal{G}$

□

La procédure de déterminisation de $\mathcal{G}$ est basée sur un dépliage du graphe de marquages du réseau logique sous-jacent et en utilisant les vecteurs d'horloges associées aux transitions, ceci en accord avec la sémantique des réseaux de Petri temporisés.

Procédure de déterminisation du comportement d'un réseau de Petri temporisé borné $\mathcal{G} = (\mathcal{P}, \mathcal{T}, M_0, \mathcal{F}, \tau)$.

1. Construire le graphe de marquages de $\mathcal{G}$, noté $reach(\mathcal{G}) = (\mathcal{M}, A, M_0, t)$.

2. Construire l'automate (max,+) correspondant, noté $G_d = (Q_d, A, q_{0,d}, \delta_d)$:
Initialisation : $A = \mathcal{T}$, $q_0 = (M_0, (0, \ldots, 0))$, $Q_d = \{(M_0, (0, \ldots, 0))\}$, `ProcédureRécursive`$(M_0, (0, \ldots, 0))$.

`ProcédureRécursive`$(M, \vec{c})$
Pour tout $a \in \mathcal{T}$ tel que $M \xrightarrow{a} M'$:
Calculer d'après l'Equation (2) les nouvelles valeurs d'horloges c' pour la transition $(M, \vec{c}) \xrightarrow{a} (M', \vec{c'})$.
S'il existe $(\hat{M}, \hat{\vec{c}}) \in Q_d$ tel que $\hat{M} = M'$ et $\hat{\vec{c}} = \vec{c'}$
posons alors
$\delta_d((M, \vec{c}), a, (\hat{M}, \hat{\vec{c}})) = (\tau_a - c_a)^+$
Sinon

"Créer un nouvel état" $Q_d := Q_d \cup (M', \vec{c'})$ et posons
$\delta_d((M, \vec{c}), a, (M', \vec{c'})) = (\tau_a - c_a)^+$
`ProcédureRécursive`$(M', \vec{c'})$
Fin Pour tout
Fin `ProcédureRécursive`

Soulignons que, d'après la sémantique opérationnelle des réseaux de Petri temporisés, deux états étendus égaux (c.-à-d., avec les mêmes marquages et les mêmes valeurs d'horloges) doivent être bisimilaires. Autrement dit, leurs comportements futurs coïncident, car il est bien connu que, pour les automates (max,+) déterministes, la bisimilarité coïncide avec l'égalité des séries formelles. De tels états peuvent être fusionnés et un nouvel état n'est pas créé s'il existe déjà un état $(\hat{M}, \hat{\vec{c}}) \in Q_d$ avec $\hat{M} = M'$ et $\hat{\vec{c}} = \vec{c'}$.

Par contre, il se peut qu'il existe deux (ou plusieurs) états étendus ayant les mêmes comportements futurs (donnant lieu aux mêmes séries formelles), mais avec des valeurs d'horloges différentes. Autrement dit, l'automate (max,+) déterministe que l'on obtient n'est pas minimal. Ceci ne constitue pas un problème majeur dans le cas où l'automate est fini (ce qui est toujours le cas pour un réseau sans conflit), car il existe des algorithmes de minimisation des automate (max,+) déterministes calculables en temps polynomial (cf. (Mohri, 1997)). Pour les réseaux avec des places de conflits (un graphe d'état suffit pour voir ce problème !), il arrive typiquement que la Procédure 3 ne se termine pas, car il y a un nombre infini d'états étendus dès que le vecteur des horloges, calculé à l'aide de l'équation (2) peut être non borné.

REMARQUE 7. — Nous avons montré comment obtenir un automate (max,+) ayant la même série formelle (max,+) qu'un réseau de Petri temporisé en considérant la politique de la course lors de résolution de conflits. Une autre politique de résolution des conflits entre les transitions sensibilisées par un marquage est celle dite de *présélection*. Dans ce cas, toutes les séquences de tirs possibles d'un point de vue logique sont considérées (même si certaines d'entre-elles contredisent l'ordre compatible avec les temporisations du réseau). Autrement dit, la politique de présélection consiste à considérer toutes les séquences de franchissement du graphe de marquages, ce qui revient à chercher les temporisations des transitions dans le graphe de marquages compatibles avec le comportement concurrent du réseau temporisé. Il s'ensuit que les automates (max,+) construits pour la politique de la présélection englobent les automates proposés dans la section précédente pour la politique de la course.

La politique de la présélection permet donc de considérer une classe plus large de comportements. Elle peut donner du sens à des modèles qui n'en auraient pas en utilisant la politique de la course. Rappelons que la modélisation des réseaux de Petri temporisés saufs par des automates (max,+) (Gaubert, Mairesse, 1999 ; Lahaye *et al.*, s. d.) considère systématiquement la politique de la présélection. Pour cette politique, il faut ajuster les règles de transitions discrètes et continues des STT : si l'on choisit de tirer une transition qui ne gagnerait pas la course, alors la valeur d'horloge

de la transition suivante, celle qui devait être tirée, dépasse sa temporisation. Dans ce cas, la transition sera instantanée (car les durées séquentialisées ne peuvent pas être négatives), mais il y aurait une perte d'information si l'horloge des transitions nouvellement sensibilisées était remise à zéro : à ces horloges devront être associées des valeurs positives correspondant à la différence entre les valeurs d'horloges et les temporisations. □

4. Condition suffisante pour la séquentialisation

Bien souvent il n'existe pas un automate (max,+) fini déterministe ayant le même comportement qu'un réseau de Petri temporisé borné donné. Pour s'en convaincre, il suffit de considérer un réseau de Petri temporisé sauf de la figure 4. Il est clair que seule transition c peut etre tiré, car b n'est gagne jamais la course avec c. Intuitivement, le fait que certains réseaux de Petri temporisés bornés ne puissent pas être séquentialisés vient de la nécessité de disposer d'une mémoire infinie pour décrire leurs comportements sous la forme d'un automate (max,+) déterministe. Plus précisément, il existe une transition sensibilisée qui ne peut ne pas être tirée sur une période de temps non bornée du fait que d'autres transitions sensibilisées (typiquement des séquence correspondant à des circuits du graphe de marquages) sont tirées. Afin de permettre la séquentialisation de tels systèmes, il faudrait un nombre infini de compteurs (sous forme de nouveaux états) pour calculer la durée de cette période non bornée pendant laquelle la transition n'est pas tirée. En termes de vecteur d'horloges, c'est ce type de situation qui se traduit par une augmentation non bornée de l'horloge de la transition "oubliée", ce qui fait que la procédure de déterminisation ne termine pas faute d'avoir un nombre infini de valeurs d'horloges différentes atteignables.

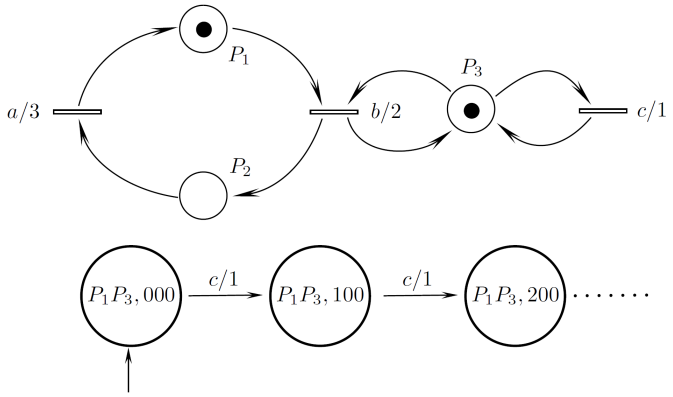

Figure 4. Réseau de Petri pour lequel la procédure ne termine pas

Ceci arrive quand on peut attribuer la ressource (jeton) dans une place de conflit indéfiniment (pendant une période de temps non bornée) seulement pour un sous ensemble de transitions situées en aval de cette place, alors que les autres transitions en aval ne sont plus tirées. Dans ce cas, nous devons mémoriser le temps de séjour des

jetons pour les autres branches afin de pouvoir déterminer la durée observée d'une transition une fois choisie pour être franchie.

Une façon de pallier ce problème est de restreindre le comportement logique du réseau en arbitrant tous les conflits de façon à ce que chaque délai entre deux franchissements consécutifs de chaque transition soit fini. Formellement, nous voudrions qu'il existe une borne sur la longueur maximale des sous mots qui ne contienne pas une transition donnée. Une telle condition formulée ci-dessous sera alors suffisante pour que le comportement d'un réseau de Petri temporisé borné puisse être séquentialisé, *i.e.,* qu'il existe un automate (max,+) fini déterministe avec le même comportement.

Nous soulignons ici que pour les réseaux de Petri temporisés avec des temporisations rationnelles (qui peuvent être transformées en nombres naturels de façon standard), et donc si les horloges de toutes les transitions sont bornées, alors l'espace des horloges est fini.

DÉFINITION 8 (Vivacité forte). — *Soit $\mathcal{G}$ un réseau de Petri temporisé borné, les transitions de $\mathcal{G}$ sont dites fortement vivantes si* $(\forall t \in \mathcal{T},\ \forall w \in L(\mathcal{G}), \forall v \preceq w,$ $\exists N \in \mathbb{N}$ *:* $(\mid v \mid \geq N \quad \Rightarrow \quad t \in v))$.

La vivacité forte signifie qu'il existe pour chaque transition une borne sur la longueur de sous mots du langage v telle que si v est plus long que cette borne, alors v contient la transition.

PROPOSITION 9. — *Soit $\mathcal{G}$ un réseau de Petri temporisé qui satisfait la vivacité forte. Son comportement est alors séquentialisable,* i.e., *il existe un automate (max,+) déterministe fini ayant la même série formelle.*

PREUVE 10. — Supposons que le langage du réseau de Petri soit rationnel. Construisons l'automate (max,+) déterministe décrit dans la procédure de déterminisation. Il est facile de voir que la vivacité forte implique qu'un nombre fini des états étendus suffit pour décrire le comportement temporel du réseau. Comme les temporisations des transitions sont rationnelles, (*i.e.,* des entiers en changeant l'échelle), le seul cas où le nombre d'états étendus est infini, est qu'il existe une horloge dont les valeurs ne sont pas bornées. Une horloge peut seulement croître au-delà de toute borne si cette horloge n'est pas périodiquement remise à zéro (depuis un certain temps, elle n'est plus remise à zéro). D'après la sémantique du réseau, ceci arrive si la transition n'est pas nouvellement sensibilisée. Mais la vivacité forte implique que, quelque soit le comportement logique (qui seul décide des transitions nouvellement sensibilisée), toute transition est franchie (et donc nouvellement sensibilisée avant son franchissement) de façon périodique. ■

Ce résultat admet comme conséquence intéressante que le comportement de tout graphe d'événements temporisé est séquentiel.

COROLLAIRE 11. — *Pour tout graphe d'événements temporisé $\mathcal{G}$ borné, il existe un automate (max,+) déterministe fini ayant la même série formelle que $\mathcal{G}$.*

PREUVE 12. — Ce résultat découle de la propriété, bien connue, des graphes d'événements temporisés (GET), notamment que ceux-ci admettent toujours un régime périodique après un régime transitoire (Baccelli *et al.*, 1992), autrement dit, dans un GET vivant, toute transition est tirée périodiquement, ce qui est exigée par la vivacité forte. ■

Notons que n'importe quel graphe d'état sauf admet un automate (max,+) déterministe fini ayant la même série formelle : comme il n'y a pas de concurrence effective (le comportement des graphes d'états temporisés est purement séquentiel), il suffit d'ajouter au graphe de marquage les temporisations des transitions, car les durées observées des transitions coïncident avec leurs temporisations, cf. (Lahaye *et al.*, s. d.).

La vivacité d'un réseau est une condition standard imposant qu'à partir de tout marquage du réseau, toutes les transitions peuvent être tirées, *i.e.*, pour tout M et t, il existe une séquence de tirs $s \in \mathcal{T}^*$ telle que t peut être tirée après s : $M \xrightarrow{s} M' \xrightarrow{t}$. Il est alors évident que la vivacité forte implique la vivacité alors que l'inverse n'est pas vrai en général.

Une question naturelle est de savoir si la condition de vivacité forte peut être vérifiée, car la définition impose l'existence d'une borne difficile à établir. Néanmoins, nous avons la proposition suivante.

PROPOSITION 13. — *La vivacité forte d'un réseau de Petri temporisé borné $\mathcal{G}$ équivaut à ce que chaque circuit de son graphe de marquages contient toutes les transitions du réseau.*

PREUVE 14. — La preuve est simple, car si tous les circuits contiennent tous les événements, alors, pour tout mot du langage du réseau et pour chaque sous mot suffisamment long, on devra trouver toutes les transitions. Inversement, s'il existe un circuit qui ne contient pas une transition, il est clair qu'en tirant répétitivement les transitions correspondant à ce circuit de graphe de marquages, il n'existera pas la borne finie N exigée par la vivacité forte. ■

La condition de vivacité forte équivaut donc à une condition structurelle restrictive. Si cette dernière n'est pas satisfaite, nous allons montrer dans la section suivante comment imposer la vivacité forte en restreignant le langage du réseau. Nous rappelons que les circuits du graphe de marquages correspondent exactement aux T-semiflots : les séquence de transitions qui ramènent le réseau au même marquage et qui sont des solutions $S \in \mathbb{N}^{\mathcal{T}}$ de l'équation $W.S = \vec{0}$, où W est la matrice d'incidence déterminée par la relation d'incidence F. Nous obtenons comme solutions des vecteurs de naturels S, dont les composantes correspondent au nombre d'occurrences de chaque transition dans le circuit. A partir de ce vecteur il est aisée d'obtenir la séquence elle-même (en faisant le produit synchrone du langage qui consiste à considérer tous les ordres d'occurrences avec le langage du réseau entier). Ceci n'est cependant pas nécessaire, car il suffit que les solutions S soient toutes telles que chaque composante est strictement positive.

5. Séquentialisation des réseaux temporisés en restreignant leurs langages

Comme il n'est souvent pas possible de trouver un automate (max,+) déterministe fini ayant le même comportement qu'un réseau de Petri temporisé, une question naturelle est de savoir si l'on peut modifier le comportement du réseau afin que le réseau modifié puisse être séquentialisé.

Le problème se pose quand une compétition se produit pour une ressource (place de conflit) et que le jeton est monopolisé par une transition, située en aval, durant un temps infini (empêchant ainsi les autres transitions en compétition d'être considérées).

Une idée naturelle consiste à restreindre le comportement logique du système (le langage du réseau) afin d'empêcher de considérer seulement un sous ensemble des transitions, situées en aval d'une place de conflit. Une telle approche permet de satisfaire la condition suffisante de déterminisation.

Cette approche, qui évite de ne considérer qu'une partie du modèle, semble raisonnable d'un point de vue pratique et permet le calcul du produit du langage du réseau par un automate décrivant la restriction choisie.

Il est possible pour chaque conflit de rattacher une borne à chaque transition, située en aval, qui spécifie le nombre maximal d'attributions consécutives de la ressource pour la transition considérée. Ainsi, nous allons empêcher les valeurs d'horloges de croître indéfiniment, ce qui ne permettait pas de séquentialiser le réseau. Par exemple, pour le conflit entre les transitions a et b du réseau, nous pouvons imposer au maximum 3 tirs consécutifs de la transition a et 2 tirs consécutifs de la transition b, cf. figure 5.

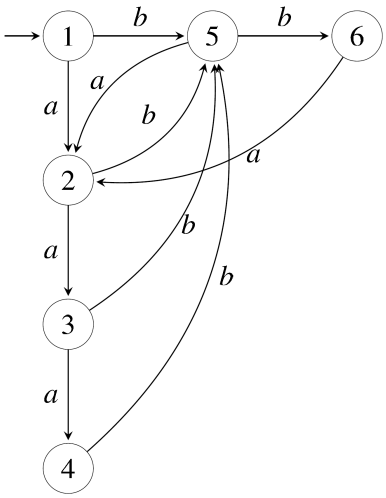

Figure 5. Automate restreignant le langage $(a \oplus b)^*$

Nous adaptons la procédure de déterminisation pour les réseaux dont le comportement n'est pas séquentialisable de la manière suivante : au lieu de considérer le

graphe de marquage original, nous considérons la graphe de marquage modifié avec tous les conflits partiellement arbitrés. Nous l'obtenons en faisant le produit synchrone du graphe de marquage original avec les automates comme celui sur la figure 5 (il est à noter que dans le cas de conflit entre trois ou plus transitions il faut utiliser plusieurs automates pour ces conflits). Puis on continue de la même façon que dans la Procédure : on déplie le graphe de marquage et fusionne des états avec marquage et valeurs d'horloges égaux. Ceci conduit à des larges automates d'attegnabilité, mais on obtient un automate (max,+) déterministe fini, ce qui est important pour des applications au contrôle comme nous l'avons indiqué. Pour réduire la compléxité on peut appliquer la commande décentralisée des automates (max,+) que nous avons récemment étudiée.

Finalement, mentionnons un point important concernant la procédure de déterminisation. Il s'avère que l'automate (max,+) déterministe issu de la procédure n'est pas assuré d'être minimal au regard de la bisimilarité des automates à multiplicité (rappelons que pour les automates (max,+) déterministes la bisimulation coïncide avec l'égalité des séries formelles). Autrement dit, le fait d'avoir un nombre d'états infini ne signifie pas que le comportement du réseau de Petri temporisé $\mathcal{G}$ ne puisse pas être séquentialisé. L'avantage de la condition suffisante sur le langage est qu'elle permet d'être toujours imposée en restreignant le comportement logique du réseau.

Notre objectif futur vise à minimiser les automates (max,+) déterministes issus de la procédure de déterminisation en utilisant des relations d'équivalence sur les valeurs d'horloges préservant la bisimulation. Notons qu'il est toujours possible de minimiser les automates (max,+) déterministes en utilisant l'algorithme de (Mohri, 1997).

6. Conclusion

Nous avons étudié la séquentialisation du comportement des réseaux de Petri temporisés bornés. Il est connu que les séries formelles (max,+) décrivant les comportement des réseaux de Petri temporisés ne peuvent pas toujours être séquentialisées, aussi seulement un semi-algorithme basé sur la sémantique a été proposé et il peut ne pas terminer. Une condition suffisante de terminaison basée sur le langage du réseau logique correspondant a été présentée. Une technique de restriction du comportement logique pour que le réseau satisfasse cette condition est également proposée.

Dans une publication à venir, nous allons proposer un nouvel algorithme de déterminisation pour les automates (max,+) tout en généralisant les conditions suffisantes connues. Une autre direction possible consiste à appliquer la déterminisation proposée dans ce papier au contrôle supervisé des réseaux de Petri temporisés bornés.

Remerciements

Ce travail a été soutenu par GAČR no. P103/11/0517 et par RVO 67985840

Bibliographie

Baccelli F., Cohen G., Olsder G.-J., Quadrat J.-P. (1992). *Synchronization and Linearity*. Wiley.

Badouel E., Bouillard A., Darondeau P., Komenda J. (2011). Residuation of tropical series: rationality issues. In *joint 50th ieee conference on decision and control and european control conference (cdc-ecc'11)*.

David R., Alla H. (2010). *Discrete, continuous, and hybrid petri nets (2nd edition)*. Paris, Springer.

Gaubert S. (1995). Performance Evaluation of (max,+) Automata. *IEEE TAC*, vol. 40, n° 12, p. 2014-2025.

Gaubert S., Mairesse J. (1999). Modeling and Analysis of Timed Petri Nets using Heaps of Pieces. *IEEE TAC*, vol. 44, n° 4, p. 683-698.

Houssin L. (2011). Cyclic jobshop problem and (max,plus) algebra. In *18th ifac wc*. Milan, Italy.

Komenda J., Lahaye S., Boimond J.-L. (2009). Supervisory Control of (max,+) Automata: A Behavioral Approach. *Disc. Event Dyn. Syst.*, vol. 19, n° 4, p. 525-549.

Lahaye S., Komenda J., Boimond J.-L. (s. d.). Compositions of (max,+) automata. *Submitted to Disc. Event Dyn. Syst.*.

Lime D., Roux O. H., Jard C. (2012). Clock transition systems. In *Cs&p*, p. 227-238.

Mohri M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, vol. 23, n° 2, p. 269-311.

Ramchandani C. (1973). *Analysis of asynchronous concurrent systems by timed Petri nets*. Ph.d. thesis, M.I.T.