CARI'10

Dépliage des réseaux de Petri temporels à modèle sous-jacent non sauf

Médésu Sogbohossou* — David Delfieu**

* Laboratoire d'Electrotechnique, de Télécommunications et d'Informatique Appliquée
Université d'Abomey-Calavi
01 BP 2009 Cotonou, BENIN
medesu.sogbohossou@gmail.com

** Institut de Recherche en Communications et Cybernétique de Nantes
1 rue de la Noé BP 92101 44321 Nantes Cedex 3, FRANCE
david.delfieu@irccyn.ec-nantes.fr

RÉSUMÉ. Pour la vérification formelle des systèmes dynamiques concurrents ou coopérants modélisés à l'aide des réseaux de Petri, la méthode du dépliage est utilisée pour endiguer le phénomène bien connu de l'explosion combinatoire. Une extension de la méthode aux réseaux de Petri temporels à modèle sous-jacent non sauf est présentée. Le dépliage obtenu est simplement un préfixe de celui du réseau de Petri ordinaire sous-jacent au réseau temporel. Pour une certaine classe de réseaux temporels, un préfixe fini capturant l'espace d'état et le langage temporisé découle du calcul d'un ensemble fini de processus finis réalisables. Les contraintes temporelles quantitatives associées à ces processus peuvent servir à valider plus efficacement les spécifications temporelles d'un système temps réel dur.

ABSTRACT. For the formal verification of the concurrent or communicating dynamic systems modeled with Petri nets, the method of the unfolding is used to cope with the well-known problem of the state explosion. An extension of the method to the non safe time Petri nets is presented. The obtained unfolding is simply a prefix of that from the underlying ordinary Petri net to the time Petri net. For a certain class of time Petri nets, a finite prefix capturing the state space and the timed language ensues from the calculation of a finite set of finite processes with valid timings. The quantitative temporal constraints associated with these processes can serve to validate more effectively the temporal specifications of a hard real-time system.

MOTS-CLÉS : réseaux de Petri temporels, caractère non sauf, dépliage temporel, préfixe complet fini, processus temporisés, systèmes temps réel durs

KEYWORDS : time Petri nets, non safeness, timed unfolding, finite complete prefix, time processes, hard real-time systems

1. Introduction

Les réseaux de Petri sont un outil de modélisation de la dynamique de systèmes descriptibles sous forme d'enchaînements d'états et d'événements. Il permet d'exprimer aisément le parallélisme vrai, les mécanismes de synchronisation et de communication inhérents au fonctionnement des systèmes concurrents. Les réseaux de Petri temporels (ou réseaux temporels) [13, 2] en constituent une extension temporisée tel qu'un intervalle statique représente un délai variable associé à chaque événement potentiel du système.

Les systèmes temps réel dur [6] impliquent des échéances temporelles strictes à respecter, et intègrent souvent des actions contrôlées par une alarme (chien de garde). Les réseaux temporels peuvent modéliser ce type de système, mais se restreignent aux systèmes sous un ordonnancement non préemptif des tâches. La vérification et la validation d'une application temps réel portent essentiellement sur l'aspect temporel, c.-à-d. sur le respect des échéances temporelles définies pour ses tâches. Des techniques analytiques [6], qui considèrent chaque tâche suivant un modèle canonique simple, sont souvent utilisées pour vérifier l'ordonnançabilité des tâches. Mais elles impliquent souvent des approximations qui conduisent à des résultats pessimistes et à un surdimensionnement de l'architecture matérielle du système. De plus, ces techniques ne prennent pas en compte l'aspect fonctionnel du système qui est analysable à partir de ses comportements. Enfin, elles ne permettent pas de vérifier des contraintes temporelles concernant les événements internes aux tâches : les seuls événements observables sont le réveil et la fin d'exécution d'une tâche. Les méthodes de vérification basées sur des modèles comportementaux tels que les réseaux temporels permettent de dépasser toutes ces limitations. Malheureusement, leur mise en œuvre à l'aide des techniques courantes de vérification, basées sur la construction d'un graphe d'états [2, 20, 3], pose le problème de l'explosion combinatoire.

L'explosion combinatoire est causée par la sémantique d'entrelacement qui est une technique coûteuse de représentation de la notion de parallélisme. En effet, cette sémantique impose une mise à plat combinatoire sous forme de séquences alternatives de tous les événements concurrents afin d'obtenir, par énumération, l'espace d'état. La taille de cet espace d'état varie alors de façon exponentielle avec le nombre d'événements.

Pour y remédier, plusieurs techniques dites *d'ordre partiel* ont été proposées. Une première catégorie de techniques alternatives sont les réductions d'ordre partiel [18, 21, 19] : elles visent à générer un espace d'états réduit, en permettant d'économiser en entrelacements des événements concurrents au prix d'une énumération non totale des états accessibles. Une seconde catégorie est donnée par la technique du dépliage qui décrit des comportements (sous forme d'une succession d'événements), appelés *processus*, sans entrelacer les événements concurrents [8]. Le calcul d'un préfixe fini du dépliage permet de capturer l'espace d'état (ce préfixe est alors dit *complet*), notamment pour les réseaux de Petri ordinaires saufs [11] et non saufs bornés [9]. La technique du dépliage a prouvé sa meilleure efficacité pour la réduction de l'explosion combinatoire, comparativement aux réductions d'ordre partiel, par exemple pour la vérification des circuits asynchrones [11, 12].

Comparativement au dépliage des réseaux ordinaires, la difficulté du dépliage des réseaux temporels concerne l'impossibilité d'évaluer de manière locale la faisabilité d'un événement sans tenir compte des concurrences et des conflits, un conflit pouvant notamment empêcher une occurrence du fait de restrictions temporelles trop contraignantes.

Ceci oblige à considérer des états globaux à l'instar de la construction d'un graphe d'état. L'extension de la technique aux réseaux temporels s'est limitée aux modèles sous-jacents saufs [16, 5, 17]. Aura et Lilius [1] formalisent la notion de *processus temporisé* pour les réseaux temporels saufs (le modèle sous-jacent pouvant être non sauf), sans définir une méthode de calcul d'un préfixe complet. Dans [17], la méthode de dépliage temporel proposée consiste à calculer les processus du réseau sous-jacent qui sont réalisables dans le modèle temporisé, en associant une caractérisation temporelle quantitative à chaque processus identifié. Elle peut toutefois nécessiter de réintroduire des entrelacements pour certains événements concurrents dans la représentation des processus.

La contribution de ce papier est l'extension de la technique de dépliage aux réseaux temporels à modèle sous-jacents non saufs. Le pouvoir d'expression des réseaux non saufs est utile pour modéliser, par exemple, les systèmes producteurs-consommateurs.

Après quelques rappels sur les réseaux de Petri à la section 2, la technique du dépliage est résumée à la section 3 ; le résultat d'un dépliage est notamment considéré comme une union de processus alternatifs. La section 4 présente le principe de calcul des processus d'un réseau temporel et définit l'algorithme pour obtenir un préfixe complet du dépliage temporel, la finitude d'un dépliage complet étant conditionnée par une restriction sur le modèle. Une illustration de la méthode est exposée à la section 5. La conclusion à la section 6 présente quelques perspectives pour ces travaux.

2. Réseaux de Petri

2.1. Réseaux de Petri ordinaires

Définition 1. *Un* réseau de Petri ordinaire *est un triplet* $\mathcal{N} \stackrel{\text{def}}{=} < \mathcal{P}, \mathcal{T}, \mathcal{W} >$ *tel que :*

– $\mathcal{P}$ *est un ensemble de places ;*

– $\mathcal{T}$ *est un ensemble de transitions (avec* $\mathcal{P} \cap \mathcal{T} = \emptyset$*) ;*

– $\mathcal{W} \subseteq \mathcal{P} \times \mathcal{T} \cup \mathcal{T} \times \mathcal{P}$ *est la relation de flux.*

Dans le contexte d'une construction finie de l'espace d'état d'un réseau de Petri, $\mathcal{P} \cup \mathcal{T}$ est fini. L'ensemble des nœuds précédents (resp. suivants) d'un nœud $x \in \mathcal{P} \cup \mathcal{T}$ est noté ${}^\bullet x \stackrel{\text{def}}{=} \{y \in \mathcal{P} \cup \mathcal{T} \mid (y, x) \in \mathcal{W}\}$ (resp. $x^\bullet \stackrel{\text{def}}{=} \{y \in \mathcal{P} \cup \mathcal{T} \mid (x, y) \in \mathcal{W}\}$).

Un *marquage* est une application $m : \mathcal{P} \rightarrow \mathbb{N}$. Dans le réseau marqué $< \mathcal{N}, m_0 >$, m_0 désigne le marquage initial. Une transition t est *sensibilisée* par m si ${}^\bullet t \subseteq m$ (en considérant m comme un multi-ensemble de places). Le *tir* de t conduit au marquage $m' = m \backslash {}^\bullet t \cup t^\bullet$, ce qui est noté $m \stackrel{t}{\rightarrow} m'$. Soit $m_0 \stackrel{\sigma}{\rightarrow} m$ t.q. $\sigma = t_1 t_2 ... t_n \in \mathcal{T}^*$: σ désigne une séquence de tirs (éventuellement vide) à partir de m_0. $< \mathcal{N}, m_0 >$ est *non sauf* si $\exists (m, p, \sigma)$ t.q. $m_0 \stackrel{\sigma}{\rightarrow} m \wedge m(p) > 1$ (avec $p \in \mathcal{P}$) ; le marquage m est également dit non sauf.

2.2. Réseaux temporels

Les réseaux temporels sont une extension des réseaux de Petri telle qu'un intervalle de délais est associé à chaque transition.

Définition 2. *Un* réseau de Petri temporel $\mathcal{N}$ *est un tuple* $\mathcal{N} \stackrel{\text{def}}{=} < \mathcal{N}_U, Efd, Lfd >$ *tel que :*

– $\mathcal{N}_U \stackrel{\text{def}}{=} < \mathcal{P}, \mathcal{T}, \mathcal{W} >$ *est le réseau de Petri ordinaire sous-jacent à* $\mathcal{N}$*,*

– $Efd : \mathcal{T} \longrightarrow \mathbb{Q}^+$ *est l'application* délais de tir au plus tôt*,*

– $Lfd : \mathcal{T} \longrightarrow \mathbb{Q}^+ \cup \{\infty\}$ *est l'application* délais de tir au plus tard*.*

Ainsi, à chaque transition $t \in \mathcal{T}$ est associée un intervalle statique $[Efd(t), Lfd(t)]$ de bornes rationnelles positives ou nulles tel que $Efd(t) \leq Lfd(t)$.

Dans la suite, un réseau temporel $\mathcal{N}$ est considéré comme sauf ($\mathcal{N}_U$ pouvant être non sauf ou non borné). Une horloge (dite locale) est associée à chaque instance de transition t sensibilisée : elle indique le temps écoulé depuis l'instant où la sensibilisation de t a eu lieue pour la dernière fois ; la transition t ne peut être franchie qu'instantanément et après un délai compris dans l'intervalle $[Efd(t), Lfd(t)]$. Le cas particulier $Efd(t) = 0$ et $Lfd(t) = \infty$ pour tout $t \in \mathcal{T}$ correspond à la sémantique des réseaux de Petri ordinaires : un tir s'effectue sans contrainte de délai.

L'ensemble des horloges actives (c.-à-d. celles des transitions sensibilisées) fonctionnent à la même vitesse et sans interruption. L'écoulement du temps est considéré comme continu et donc à valeurs dans $\mathbb{R}^+$ (réels non négatifs). Dans un réseau temporel, un état est non seulement défini par un marquage, mais également par l'ensemble des valeurs courantes des horloges actives des transitions sensibilisées puisqu'elles déterminent les tirs possibles à un instant. Soient m, un marquage atteint par $< \mathcal{N}, m_0 >$ et $T_m \stackrel{\text{def}}{=} \{t \in \mathcal{T} \mid {}^\bullet t \subseteq m\}$ (ensemble des transitions sensibilisées par m). L'application $\vartheta : T_m \longrightarrow \mathbb{R}^+$ représente les valeurs des horloges actives à un instant donné. Un *état* S du réseau temporel est ainsi défini par le couple (m, ϑ). A l'instant initial, l'état est $S_0 \stackrel{\text{def}}{=} (m_0, \vartheta_0)$ tel que $\forall t \in T_{m_0},\ \vartheta_0(t) = 0$ (toute transition t est nouvellement sensibilisée).

A l'instant où l'horloge associée à une transition t atteint la borne maximale $Lfd(t)$, t doit impérativement être tirée, sauf si le tir immédiat d'une autre transition t' franchissable désensibilise t (cas de *conflit structurel* : ${}^\bullet t \cap {}^\bullet t' \neq \emptyset$). Ce principe de fonctionnement est connu sous l'appellation de *sémantique forte* [4].

Dans un réseau ordinaire, un changement d'état est simplement déterminé par le tir d'une transition sensibilisée. L'adjonction du temps implique une seconde forme de changement d'état : l'écoulement d'un quantum de temps sans transition discrète (c.-à-d. avec uniquement la mise à jour des horloges actives) et réglementé par la sémantique forte. Le principe de l'écoulement dense du temps peut occasionner une infinité d'états possibles.

Pour définir la sémantique formelle d'un réseau temporel, une règle pourrait être exprimée pour une transition discrète (tir instantané de transition) et une autre pour une transition continue (écoulement de temps sans tir). En ne s'intéressant qu'aux possibilités des transitions discrètes, $S \xrightarrow{(\theta,t)} S'$ signifie que $\theta \in \mathbb{R}^+$ unités de temps s'écoulent depuis l'état $S \stackrel{\text{def}}{=} (m, \vartheta)$ pour effectuer un tir $t \in T_m$ conduisant à l'état $S' \stackrel{\text{def}}{=} (m', \vartheta')$.

Définition 3. $S \xrightarrow{(\theta,t)} S'$ *si et seulement si :*

– ${}^\bullet t \subseteq m$*,*

– $\vartheta(t) + \theta \geq Efd(t)$*,*

– $\forall t' \in T_m,\ \vartheta(t') + \theta \leq Lfd(t')$*.*

L'état S' est défini par :

– $m' \stackrel{\text{def}}{=} m \backslash {}^\bullet t \cup t^\bullet$,

– $\vartheta' : T_{m'} \longrightarrow \mathbb{R}^+$ *tel que* $\forall t' \in T_{m'}$ *:*

- si ${}^\bullet t' \subseteq m \backslash {}^\bullet t$*, alors* $\vartheta'(t') = \vartheta(t') + \theta$ *(t' est dite persistante),*

- sinon $\vartheta'(t') = 0$ *(t' est nouvellement sensibilisée).*

Une séquence temporisée $\tau = (\theta_1, t_1), (\theta_2, t_2), .., (\theta_n, t_n)$ (de support $\sigma = t_1 t_2 ... t_n \in \mathcal{T}^*$) est *réalisable* si $S_{i-1} \stackrel{(\theta_i, t_i)}{\longrightarrow} S_i$, avec $0 < i \leq n$.

2.3. Calcul de l'espace d'état

Le calcul de l'espace d'état des réseaux de Petri (et de leurs extensions en général), sous forme d'un *graphe d'état*, repose sur la sémantique séquentielle : les états accessibles sont énumérés en calculant les séquences de transitions possibles à partir de l'état initial.

Plusieurs constructions pour couvrir l'espace d'état ont été définies pour les réseaux temporels [2, 3], notamment celle préservant les marquages accessibles et les séquences temporisées (pour vérifier les propriétés LTL de la logique temporelle) et celle préservant en plus la structure de branchement (pour vérifier les propriétés CTL). Les résultats des travaux sur les vérifications basées sur le dépliage d'un réseau ordinaire se limitent aux propriétés LTL [7]. Pour la première construction, appelée par la suite *graphe des classes*, une classe d'états C_σ est associée à chaque séquence σ de tirs depuis l'état initial S_0 et regroupe l'ensemble des états atteints par toutes les séquences temporisées de support σ. C_σ est définie par le marquage m atteint par σ et par un domaine de tirs D qui est une conjonction de contraintes de délais entre deux instances dans l'ensemble constitué par les transitions sensibilisées par m et le dernier tir de la séquence σ. Un réseau temporel est borné si et seulement si son graphe des classes est fini [2].

3. Dépliage d'un réseau de Petri

Un dépliage est une structure qui représente des comportements d'un système en montrant explicitement les relations de concurrence ou de causalité entre ses événements. Deux comportements s'y différencient par des choix distincts d'événements en relation de conflit. Parce qu'un dépliage n'entrelace pas les événements concurrents, il peut contribuer à un gain exponentiel en nombre de nœuds comparativement à un graphe d'état.

3.1. Réseau d'occurrence, dépliage et processus

La structure de branchement d'un dépliage prend la forme d'un réseau de Petri dénommé *réseau d'occurrence*.

Un réseau d'occurrence est un réseau de Petri $\mathcal{O} \stackrel{\text{def}}{=} < \mathcal{B}, \mathcal{E}, \mathcal{F} >$ tel que :

– $\mathcal{F}^+$ (la fermeture transitive de $\mathcal{F}$) est une relation irréflexive (relation d'ordre strict),

– $\forall b \in \mathcal{B}, |{}^\bullet b| \leq 1$,

– $\forall e \in \mathcal{E}, {}^\bullet e \neq \emptyset$ et $e^\bullet \neq \emptyset$.

$\mathcal{B}$ (resp. $\mathcal{E}$) constitue l'ensemble des *conditions* (resp. l'ensemble des *événements*). Pour un événement $e \in \mathcal{E}$, ${}^\bullet e$ (resp. $e^\bullet$) forme les *pré-conditions* (resp. *post-conditions*) de e. $Min(\mathcal{O}) \stackrel{\text{def}}{=} \{b \in \mathcal{B} \mid {}^\bullet b = \emptyset\}$ désigne les *conditions minimales* et $Max(\mathcal{O}) \stackrel{\text{def}}{=} \{b \in \mathcal{B} \mid b^\bullet = \emptyset\}$ désigne les *conditions maximales*.

Trois types de relations, mutuellement exclusives, sont définis entre deux nœuds quelconques de $\mathcal{O}$:

– la *relation de causalité* notée $\prec$: $\forall x, y \in \mathcal{B} \cup \mathcal{E}, x \prec y$ ssi $(x, y) \in \mathcal{F}^+$; par extension, $x \preceq y$ est équivalent à $x \prec y \vee x = y$;

– la *relation de conflit* notée $\sharp$: $\forall e_1, e_2 \in \mathcal{E}\ (e_1 \neq e_2),\ e_1 \sharp e_2$ si ${}^\bullet e_1 \cap {}^\bullet e_2 \neq \emptyset$. De plus, si $e_1 \sharp e_2$, alors $\forall x, y \in \mathcal{B} \cup \mathcal{E},\ e_1 \prec x \wedge e_2 \prec y \ \Rightarrow\ x \sharp y \wedge x \sharp e_2 \wedge e_1 \sharp y$;

– la *relation de concurrence* notée $\wr$: $\forall x, y \in \mathcal{B} \cup \mathcal{E}$ $(x \neq y)$, $x \wr y$ ssi $\neg((x \prec y) \vee (y \prec x) \vee (x \sharp y))$.

Soit $B \subseteq \mathcal{B}$ tel que $\forall b, b' \in B, b \neq b' \Rightarrow b \wr b'$: B est appelé une *coupe*. Soit une coupe B telle que $\forall b \in B, \nexists b' \in \mathcal{B} \backslash B,\ b \wr b'$: la coupe B est dite *maximale*.

Définition 4. Le dépliage $Unf_F \stackrel{\text{def}}{=} < \mathcal{O}_F, \lambda_F >$ *du réseau marqué* $< \mathcal{N}, m_0 >$*, avec le réseau d'occurrence* $\mathcal{O}_F \stackrel{\text{def}}{=} < \mathcal{B}_F, \mathcal{E}_F, \mathcal{F}_F >$ *et la fonction d'étiquetage* $\lambda_F : \mathcal{B}_F \cup \mathcal{E}_F \rightarrow \mathcal{P} \cup \mathcal{T}$ *tel que* $\lambda(\mathcal{B}_F) \subseteq \mathcal{P}$ *et* $\lambda(\mathcal{E}_F) \subseteq \mathcal{T}$*, est donné par :*

1) $\forall p \in \mathcal{P}$*, si* $m_0(p) \neq \emptyset$*, alors* $B_p \stackrel{\text{def}}{=} \{b \in \mathcal{B}_F \mid \lambda_F(b) = p \wedge {}^\bullet b = \emptyset\}$ *et* $m_0(p) = |B_p|$ *;*

2) $\forall B_t \subseteq \mathcal{B}_F$ *t.q.* B_t *est une coupe, si* $\exists t \in \mathcal{T}, \lambda_F(B_t) = {}^\bullet t \wedge |B_t| = |{}^\bullet t|$*, alors :*

- ***a)*** $\exists! e \in \mathcal{E}_F$ *t.q.* ${}^\bullet e = B_t \wedge \lambda_F(e) = t$ *;*
- ***b)*** *si* $t^\bullet \neq \emptyset$*, alors* $B'_t \stackrel{\text{def}}{=} \{b \in \mathcal{B}_F \mid {}^\bullet b = \{e\}\}$ *est tel que* $\lambda_F(B'_t) = t^\bullet \wedge |B'_t| = |t^\bullet|$ *;*
- ***c)*** *si* $t^\bullet = \emptyset$*, alors* $B'_t \stackrel{\text{def}}{=} \{b \in \mathcal{B}_F \mid {}^\bullet b = \{e\}\}$ *est tel que* $\lambda_F(B'_t) = \emptyset \wedge |B'_t| = 1$ *;*

3) $\forall B_t \subseteq \mathcal{B}_F$*, si* B_t *n'est pas une coupe, alors* $\nexists e \in \mathcal{E}_F$ *t.q.* ${}^\bullet e = B_t$*.*

La définition 4 exprime l'algorithme d'un dépliage *exhaustif* de $< \mathcal{N}, m_0 >$: l'item 1 décrit la création des conditions minimales et l'item 2 décrit la création des événements successifs et de leurs post-conditions. L'item 3 impose que tout événement soit une action possible : il n'existe pas de nœud supplémentaire à ceux créés par les items 1 et 2. Bien entendu, *le* dépliage est infini lorsque $< \mathcal{N}, m_0 >$ admet un comportement infini. Les transitions puits (c.-à-d. $t \in \mathcal{T}, t^\bullet = \emptyset$), souvent utilisées pour modéliser une action sans suite dans un système temps réel, sont spécifiquement prises en compte par l'item 2.c).

Soit $\mathcal{E} \subset \mathcal{E}_F$. Le réseau d'occurrence $\mathcal{O} \stackrel{\text{def}}{=} < \mathcal{B}, \mathcal{E}, \mathcal{F} >$ associé à $\mathcal{E}$ tel que $\mathcal{B} \stackrel{\text{def}}{=} \{b \in \mathcal{B}_F \mid \exists e \in \mathcal{E}, b \in {}^\bullet e \cup e^\bullet\}$ et $\mathcal{F} \stackrel{\text{def}}{=} \{(x, y) \in \mathcal{F}_F \mid x \in \mathcal{E} \vee y \in \mathcal{E}\}$ est *un préfixe* de $\mathcal{O}_F$ si $Min(\mathcal{O}) = Min(\mathcal{O}_F)$. Par extension, $Unf \stackrel{\text{def}}{=} < \mathcal{O}, \lambda >$ (avec λ, la restriction de λ_F à $\mathcal{B} \cup \mathcal{E}$) est un préfixe du dépliage Unf_F.

Il faut noter que les noms (les éléments dans $\mathcal{E}$ et $\mathcal{B}$) donnés aux nœuds d'un même dépliage peuvent être différents d'une implémentation à une autre. Lorsque m_0 est sauf, un nom peut être choisi de manière indépendante d'une implémentation en utilisant l'arborescence formée par ses prédécesseurs causaux ainsi que le nom du nœud correspondant dans $\mathcal{N}$ [8].

Définition 5. *Soit* $\mathcal{E}_i \subseteq \mathcal{E}_F$*.* $\mathcal{E}_i$ *est un* processus *de* $< \mathcal{N}, m_0 >$ *ssi :* $\forall e \in \mathcal{E}_i,\ \forall e' \in \mathcal{E}_F$ *si* $e' \prec e$ *alors* $e' \in \mathcal{E}_i$ *(*$\mathcal{E}_i$ *est une fermeture causale), et* $\forall (e, e') \in \mathcal{E}_i \times \mathcal{E}_i, \neg(e \sharp e')$*.*

Le réseau $\mathcal{C}_i \stackrel{\text{def}}{=} < \mathcal{B}_i, \mathcal{E}_i, \mathcal{F}_i >$ associé à un processus fini $\mathcal{E}_i$ est appelé *réseau causal*. Il vérifie : $\forall b \in \mathcal{B}_i, |b^\bullet| \leq 1$. $Max(\mathcal{C}_i)$ est l'*état* de $\mathcal{C}_i$. $Min(\mathcal{C}_i)$ et $Max(\mathcal{C}_i)$ sont des coupes maximales. De manière générale, une coupe maximale $B \subseteq \mathcal{B}_i$ correspond à un marquage accessible m de $< \mathcal{N}, m_0 >$ tel que $\forall p \in \mathcal{P}, m(p) = |B_p|$ avec $B_p = \{b \in B \mid \lambda(b) = p\}$.

La *configuration locale* d'un événement e est définie par l'ensemble $[e] \stackrel{\text{def}}{=} \{e' \mid e' \preceq e\}$ et constitue un processus.

Les conflits dans un dépliage découlent de ce qu'il existe un marquage accessible (une coupe dans un dépliage) tel que deux ou plusieurs transitions du réseau marqué $< \mathcal{N}, m_0 >$ soient sensibilisées et que le tir de l'une désactive une autre. D'où la proposition :

Proposition 1. *Soient $e_1, e_2 \in \mathcal{E}_F$. Si $e_1 \sharp e_2$, alors il existe $(e'_1, e'_2) \in [e_1] \times [e_2]$ tel que ${}^\bullet e'_1 \cap {}^\bullet e'_2 \neq \emptyset$ et ${}^\bullet e'_1 \cup {}^\bullet e'_2$ constitue une coupe.*

Définition 6. *Un préfixe $Unf \stackrel{\text{def}}{=} << \mathcal{B}, \mathcal{E}, \mathcal{F} >, \lambda >$ de Unf_F est* complet *lorsque, pour tout marquage accessible m de $< \mathcal{N}, m_0 >$, il existe un processus $\mathcal{E}_i \subseteq \mathcal{E}$ tel que :*

1) $Max(\mathcal{C}_i)$ correspond à m,

2) si $\exists t \in \mathcal{T}$ t.q. ${}^\bullet t \subseteq m$, alors $\exists e \in \mathcal{E}$ t.q. ${}^\bullet e \subseteq Max(\mathcal{C}_i) \wedge \lambda(e) = t$.

Une extension de la définition pourra être aisément faite pour prendre en compte la représentation dans $\mathcal{E}$ d'une multiplicité d'instances t sensibilisées.

A l'instar d'un graphe d'état, un préfixe complet du dépliage d'un réseau ordinaire contient l'espace d'état et représente l'ensemble des séquences possibles de tirs.

3.2. Un préfixe de dépliage est une union de processus alternatifs

Soient : Unf_F le dépliage exhaustif du réseau marqué $< \mathcal{N}, m_0 >$, $\mathcal{C} \stackrel{\text{def}}{=} < \mathcal{B}, \mathcal{E}, \mathcal{F} >$ un réseau causal préfixe de $\mathcal{O}_F$, et un processus $\mathcal{E}_i \subseteq \mathcal{E}$. L'ensemble des *extensions potentielles* de $\mathcal{E}_i$ est donné par : $PE(\mathcal{E}_i) \stackrel{\text{def}}{=} \{e \in \mathcal{E}_F \mid {}^\bullet e \subseteq Max(\mathcal{C}_i)\}$. Il correspond aux instances sensibilisées par le marquage atteint en réalisant le processus $\mathcal{E}_i$.

Le processus $\mathcal{E}_i$ est *maximal* pour le préfixe $\mathcal{E}$ si $PE(\mathcal{E}_i) \cap \mathcal{E} = \emptyset$: il n'admet pas d'extension potentielle dans le préfixe $\mathcal{E}$. Un préfixe de dépliage peut être considéré comme une union des processus maximaux le composant. Soit $\mathbb{P}(\mathcal{E})$, l'ensemble des parties de $\mathcal{E}$.

Définition 7. *Les* processus maximaux alternatifs[1] *du préfixe $\mathcal{E}$ sont donnés par l'ensemble $\overline{\mathcal{E}} \subseteq \mathbb{P}(\mathcal{E})$ tel que : $\mathcal{E} = \bigcup_{\mathcal{E}_j \in \overline{\mathcal{E}}} \mathcal{E}_j$ et tout $\mathcal{E}_i \in \overline{\mathcal{E}}$ est un processus maximal.*

Proposition 2. *Pour $\mathcal{E}_i \in \overline{\mathcal{E}}$, $\forall e \in \mathcal{E} \backslash \mathcal{E}_i$, il existe $e_i \in \mathcal{E}_i$ tel que $e \sharp e_i$.*

Démonstration. Les relations possibles entre $e \in \mathcal{E} \backslash \mathcal{E}_i$ et $e_i \in \mathcal{E}_i$ sont la causalité, le conflit et la concurrence. Raisonnons par l'absurde :

– $e \prec e_i$: D'après la définition d'un processus, e ferait partie de $\mathcal{E}_i$, ce qui n'est pas le cas.

1. Deux processus $\mathcal{E}_j, \mathcal{E}_k \subset \mathcal{E}$ sont *alternatifs* s'ils ne contiennent pas les mêmes événements ; plus précisément : $\mathcal{E}_j, \mathcal{E}_k \subset \mathcal{E}_j \cup \mathcal{E}_k$.

– $e_i \prec e$: Puisque $e \notin \mathcal{E}_i$, e ne peut être que successeur d'une (ou plusieurs) post-condition $b_i \in Max(\mathcal{C}_i)$ tel que $e_i \prec b_i \prec e$. Les post-conditions de l'état final $Max(\mathcal{C}_i)$ n'admettant pas d'événement successeur dans $\mathcal{E}$, $e_i \prec e$ est donc impossible.

– $e \wr e_i$: L'ensemble $[e] \cup [e_i]$ constitue alors un processus : en effet, par définition, $[e]$ et $[e_i]$ ne peuvent pas contenir de conflit ; s'il existe un conflit, il ne peut opposer que des événements ancêtres appartenant respectivement à $[e]\backslash\{e\}$ et $[e_i]\backslash\{e_i\}$, ce qui impliquerait par fermeture transitive de la causalité que $e \sharp e_i$. Ceci est contraire à l'hypothèse $e_i \wr e$. Par extension, si e n'est en conflit avec aucun événement dans $\mathcal{E}_i$, alors $\mathcal{E}_i \cup [e]$ constitue également un processus, ce qui est absurde sachant que $\mathcal{E}_i$ est maximal dans $\mathcal{E}$.

La seule autre relation possible est le conflit $e \sharp e_i$. Le processus $\mathcal{E}_i$ contient donc nécessairement un événement en conflit avec e. □

Deux événements $e_1, e_2 \in \mathcal{E}_F$ tels que $e_1 \sharp e_2$ sont des *choix potentiels* lorsque l'ensemble ${}^\bullet e_1 \cup {}^\bullet e_2$ forme une coupe dans $\mathcal{B}_F$. Ceci signifie que $< \mathcal{N}, m_0 >$ admet un marquage accessible tel que les deux événements en conflit sont activés simultanément. Un seul choix d'événement parmi plusieurs mutuellement en conflit peut figurer dans un processus donné. Pour un choix e réalisé dans un processus $\mathcal{E}_i$ (c.-à-d. $e \in \mathcal{E}_i$), l'ensemble $Conf_i(e) \stackrel{\text{def}}{=} \{e' \in \mathcal{E}_F \mid {}^\bullet e \cap {}^\bullet e' \neq \emptyset \wedge {}^\bullet e' \subseteq \mathcal{B}_i\}$ constitue les *choix alternatifs* à e. Un événement est un choix dans un processus s'il admet au moins une alternative ($Conf_i(e) \neq \emptyset$). Les choix alternatifs à e sont irrémédiablement désactivés en produisant cet événement.

Un événement $e \in \mathcal{E}_F$ est *identifié* dans le processus $\mathcal{E}_i$ si la coupe ${}^\bullet e$ est contenue dans $\mathcal{B}_i$. L'identification des choix alternatifs désactivés par un processus permet de connaître les processus qui se différencient de lui en réalisant ces choix.

Considérons deux processus $\mathcal{E}_1, \mathcal{E}_2 \subseteq \mathcal{E}$ ($\mathcal{E}_1 \neq \mathcal{E}_2$) ; l'intersection $\mathcal{E}_1 \cap \mathcal{E}_2$ constitue un processus inclus dans $\mathcal{E}$. En effet, d'après la définition 5 :

– $\mathcal{E}_1$ et $\mathcal{E}_2$ ne contenant pas de couple d'événements en conflit, il en est de même pour tout sous-ensemble tel que $\mathcal{E}_1 \cap \mathcal{E}_2$;

– tout prédécesseur e dans $\mathcal{E}_1 \cap \mathcal{E}_2$ figure dans $\mathcal{E}_1$ et dans $\mathcal{E}_2$, et donc figure également dans l'intersection $\mathcal{E}_1 \cap \mathcal{E}_2$.

Théorème 1. *Si $\mathcal{E}_1$ et $\mathcal{E}_2$ ($\mathcal{E}_1 \neq \mathcal{E}_2$) sont deux processus maximaux inclus dans $\mathcal{E}$, alors tout événement dans l'ensemble $PE(\mathcal{E}_1 \cap \mathcal{E}_2) \cap \mathcal{E}$ est un choix potentiel admettant une alternative dans l'un des processus $\mathcal{E}_1$ ou $\mathcal{E}_2$.*

Démonstration. Soit $\mathcal{E}_3 \stackrel{\text{def}}{=} \mathcal{E}_1 \cap \mathcal{E}_2$. D'après la proposition 2, $\forall e \in PE(\mathcal{E}_3) \cap (\mathcal{E}\backslash\mathcal{E}_1)$, il existe $e_1 \in \mathcal{E}_1$ tel que $e_1 \sharp e$. $e_1 \notin \mathcal{E}_3$: en effet, si $e_1 \in \mathcal{E}_3$ et $e' \in \mathcal{E}_3$ tel que ${e'}^\bullet \cap {}^\bullet e \neq \emptyset$, alors $e_1 \sharp e'$, ce qui est absurde sachant que $\mathcal{E}_3$ contenant e' et e_1 est un processus. D'après la proposition 1, il existe $e'_1 \in [e_1]$ tel que ${}^\bullet e'_1 \cap {}^\bullet e \neq \emptyset$ et ${}^\bullet e'_1 \cup {}^\bullet e$ soit une coupe (pour la même raison que e_1, $e'_1 \in \mathcal{E}_1\backslash\mathcal{E}_3$). Ainsi, e'_1 et e sont des choix alternatifs. Le cas particulier $e \in PE(\mathcal{E}_3) \cap (\mathcal{E}_2\backslash\mathcal{E}_1)$ est implicitement pris en compte.

A l'inverse, tout $e \in PE(\mathcal{E}_3) \cap (\mathcal{E}\backslash\mathcal{E}_2)$ possède un choix alternatif $e'_2 \in PE(\mathcal{E}_3)$ produit dans $\mathcal{E}_2$. □

En somme, deux processus maximaux d'un préfixe de dépliage ne se distinguent que par des choix différents. Par la suite, un dépliage sera obtenu en calculant un ensemble de processus maximaux. Etant calculé un premier processus maximal, tous les processus

alternatifs peuvent être obtenus à partir d'une racine commune et en réalisant un sous-ensemble des choix alternatifs identifiés dans le premier. De proche en proche, l'ensemble des processus alternatifs couvrant l'espace d'état pourra être calculé.

3.3. Dépliage temporel

Soient $< \mathcal{N}, m_0 >$ un réseau temporel marqué tel que $\mathcal{N} \stackrel{\text{def}}{=} < \mathcal{N}_U, Efd, Lfd >$, et $Unf_F \stackrel{\text{def}}{=} << \mathcal{B}_F, \mathcal{E}_F, \mathcal{F}_F >, \lambda_F >$ le dépliage du modèle sous-jacent $< \mathcal{N}_U, m_0 >$.

Pour un processus $\mathcal{E}_n \subseteq \mathcal{E}_F$ tel que $\mathcal{E}_n = \{e_1, \ldots, e_n\}$, $\rho \stackrel{\text{def}}{=} e_1 e_2 \ldots e_n$ est un *entrelacement valide* de $\mathcal{E}_n$ si cette séquence respecte la causalité entre les événements : $\forall (e_i, e_j) \in \mathcal{E}_n \times \mathcal{E}_n$ tel que $1 \leq i < j \leq n$, $\neg(e_j \prec e_i)$ est vérifié. La séquence de tirs correspondante de $< \mathcal{N}_U, m_0 >$ est $\sigma \stackrel{\text{def}}{=} \lambda_F(e_1)\lambda_F(e_2) \ldots \lambda_F(e_n)$.

$< \mathcal{E}_n, d >$ est un *processus temporisé* si $d : \mathcal{E}_n \longrightarrow \mathbb{R}^+$ (fonction de *temporisation*). Pour $e_i \in \mathcal{E}_n$, $d(e_i) \in \mathbb{R}^+$ est l'instant d'occurrence associé à e_i. Un instant $d(e_i)$ est donné en unités de temps absolu, c-à-d. en temps relatif à l'instant 0 de l'état initial de $< \mathcal{N}, m_0 >$. Un processus temporisé est l'équivalent en sémantique d'ordre partiel d'une ou plusieurs séquences temporisées, deux événements concurrents de même date d'occurrence n'étant pas entrelacés.

Un processus temporisé $< \mathcal{E}_n, d >$ est *réalisable* dans le modèle temporisé $< \mathcal{N}, m_0 >$ s'il existe un entrelacement valide $\rho \stackrel{\text{def}}{=} e_1 e_2 \ldots e_n$ de $\mathcal{E}_n$ tel que la séquence temporisée $\tau \stackrel{\text{def}}{=} (d(e_1), \lambda_F(e_1)), (d(e_2) - d(e_1), \lambda_F(e_2)), \ldots, (d(e_n) - d(e_{n-1}), \lambda_F(e_n))$, de support correspondant à ρ, est réalisable dans $< \mathcal{N}, m_0 >$. Plus généralement, un processus $\mathcal{E}_n$ est réalisable s'il admet un processus temporisé réalisable. Tout processus du modèle sous-jacent $\mathcal{N}_U$ n'est pas toujours réalisable dans le modèle temporisé $\mathcal{N}$ à cause de la sémantique forte.

Définition 8. *Soit $TUnf \stackrel{\text{def}}{=} << \mathcal{B}, \mathcal{E}, \mathcal{F} >, \lambda >$, un préfixe de Unf_F. $TUnf$ est un* dépliage temporel *de $< \mathcal{E}, m_0 >$ si pour tout $e \in \mathcal{E}$, il existe un processus réalisable $\mathcal{E}_i \in \overline{\mathcal{E}}$ tel que $e \in \mathcal{E}_i$.*

Bien entendu, pour caractériser entièrement un dépliage temporel, il faut lui associer des contraintes temporelles quantitatives reliant ses événements. Aura et Lilius [1] définissent comment obtenir une caractérisation temporelle quantitative d'un processus support $\mathcal{E}_n$ par un ensemble de contraintes linéaires alternatives entre ses événements. La procédure permet également de vérifier qu'un processus support quelconque admet une temporisation valide. Cette caractérisation, de type ordre partiel, est moins coûteuse (en cas de concurrence) qu'une évaluation séquentielle obtenue sur plusieurs entrelacements des événements du processus support.

L'aspect quantitatif d'un dépliage temporel peut être donné par les caractérisations temporelles associées aux processus maximaux réalisables dans $\overline{\mathcal{E}}$. Un dépliage temporel est *complet* si ses processus maximaux réalisables décrivent l'espace d'état et l'ensemble des séquences temporisées du réseau temporel $\mathcal{N}$.

4. Calcul des processus réalisables d'un dépliage complet

Dans la suite, l'objectif est d'obtenir un préfixe de dépliage de réseau temporel en calculant un ensemble de processus maximaux réalisables rendant ce préfixe complet.

Dans le graphe des classes d'un réseau temporel borné admettant un comportement infini, un ensemble fini de séquences finies représente l'espace d'état : la procédure de calcul termine le développement d'une séquence potentiellement infinie lorsqu'une classe d'état atteinte par une séquence précédemment obtenue est à nouveau atteinte. Dans un dépliage temporel, un concept similaire appliqué aux calculs des processus permettra de couvrir l'espace d'état.

4.1. Spécificités du calcul d'un processus de réseau temporel

Le calcul d'un processus quelconque de réseau temporel peut s'effectuer itérativement : si $\mathcal{E}_i$ est le processus courant réalisable ($i = 0$ à l'état initial, avec $\mathcal{E}_0 = \emptyset$), l'ajout d'une extension potentielle $e_{i+1} \in PE(\mathcal{E}_i)$ est tel que le processus étendu $\mathcal{E}_{i+1} \stackrel{\text{def}}{=} \mathcal{E}_i \cup \{e_{i+1}\}$ admet une temporisation valide ; dans ce contexte temporisé, $e_{i+1} \in PE(\mathcal{E}_i)$ est alors une extension *possible* pour $\mathcal{E}_i$. La faisabilité d'un processus s'évalue à partir de ses événements (internes), de ses choix alternatifs et de ses extensions potentielles, en utilisant les contraintes Efd/Lfd des transitions correspondantes. Selon Aura et Lilius [1], une évaluation ordre partiel quantitative s'effectue généralement en temps exponentiel avec la taille du processus, contrairement à la faisabilité d'une temporisation de processus qui peut toujours s'évaluer en temps polynomial.

Il faut remarquer que deux choix potentiels en conflit dans un dépliage ne sont pas nécessairement dus à un conflit structurel dans $\mathcal{N}$, comme le montre la figure 1 : dans le dépliage du réseau sous-jacent (b), les choix potentiels e_3 et e_4 en conflit portent sur la même transition t_3. Il est évident que e_3 est impossible d'après les spécifications temporelles en (a). Un tel conflit est dû au caractère non sauf de $< \mathcal{N}_U, m_0 >$: la coupe ${}^\bullet e_3 \cup {}^\bullet e_4$ (figure 1(b)) n'est pas un marquage sauf. Ainsi, un choix désactivé par un processus donné (ici, e_3 est désactivé par le processus $\{e_1, e_2, e_4\}$) peut ne pas constituer un choix réalisable dans un processus alternatif (ici, le processus $\{e_1, e_2, e_3\}$ n'est pas réalisable).

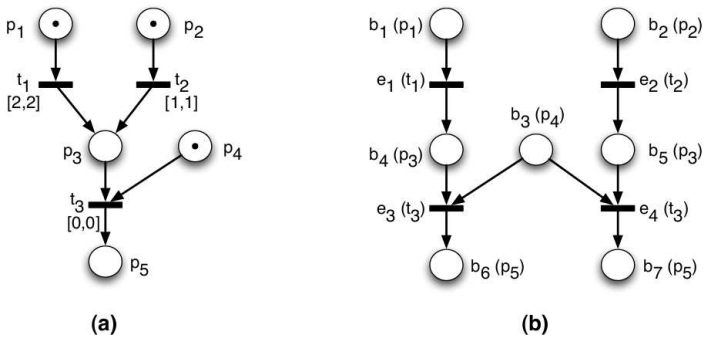

Figure 1. *Dépliage du modèle sous-jacent non sauf d'un réseau temporel sauf*

4.2. Etat charnière

L'événement fictif e_0 créant m_0 ($e_0^\bullet \stackrel{\text{def}}{=} Min(\mathcal{O}_F)$) constitue l'origine temporelle de date d'occurrence d_0 supposée nulle. Soient un processus $\mathcal{E} \subseteq \mathcal{E}_F$ et $e_i \in \mathcal{E}$: la variable d_i représente une date d'occurrence de e_i. Pour $e_j \in \mathcal{E}$, $d_{i,j} \stackrel{\text{def}}{=} d_j - d_i$ (délai d'occurrence de e_j relativement à e_i). Soit $\mathcal{C}$, le réseau causal associé à $\mathcal{E}$.

Définition 9. $Max(\mathcal{C})$ *est un* état charnière *ssi* $\exists e_j \in \mathcal{E} \cup \{e_0\}$ *t.q. :* $\forall e_k \in PE(\mathcal{E}), e_j^\bullet \cap {}^\bullet e_k \neq \emptyset$ *et* $\forall e_i \in \mathcal{E}\backslash\{e_j\}$, *si* $e_i \wr e_j$ *alors* $d_{i,j} > 0$ *dans toute temporisation valide de* $\mathcal{E}$.

$d_{i,j} > 0$ signifie que l'événement concurrent e_i précède e_j dans le processus $\mathcal{E}$. Dans un état charnière, l'événement e_j est toujours le *dernier* à se produire dans le processus $\mathcal{E}$ et constitue un parent[2] de toute extension potentielle de $\mathcal{E}$. En particulier, $Min(\mathcal{C})$ est un état charnière.

Un dépliage peut être obtenu sans entrelacer des événements concurrents. La conséquence est que les états accessibles ne sont généralement pas tous visités. Soit $\rho_i \stackrel{\text{def}}{=} e_1 e_2 ... e_i$, la séquence adoptée pour le calcul du processus $\mathcal{E}_i$. Un état charnière est toujours emprunté quelque soit la séquence adoptée :

Proposition 3. *Soit une coupe maximale* $B \subseteq \mathcal{B}_i$. *Si* B *est un état charnière, alors il existe* ρ_j, *un préfixe de* ρ_i ($j \leq i$), *tel que le réseau causal* $\mathcal{C}_j$ *associé vérifie* $Max(\mathcal{C}_j) = B$.

Démonstration. Le cas $B = Max(\mathcal{C}_i)$ est trivial puisque $i = j \Rightarrow \rho_i = \rho_j \wedge \mathcal{C}_i = \mathcal{C}_j$.

Considérons le cas $B \neq Max(\mathcal{C}_i)$. Soit $E \stackrel{\text{def}}{=} \{e \in \mathcal{E}_i \mid e^\bullet \subseteq B\}$, l'ensemble des événements pour un processus dans l'état B tel que pour tout $e_x \in \mathcal{E}_i\backslash E \wedge e_x{}^\bullet \cap B \neq \emptyset$, $\exists e_y \in E, e_x \prec e_y$. Les événements de E sont évidemment deux à deux concurrents.

Cas $|E| = 1$ **:**

L'unique événement $e \in E$ n'admet pas d'événement concurrent dans le processus $\mathcal{E}_i$: puisqu'un processus ne contient pas de conflit, tout événement $e' \in \mathcal{E}_i\backslash\{e\}$ est tel que $e' \prec e \vee e \prec e'$.

Soit le processus $\mathcal{E}_{j-1} \stackrel{\text{def}}{=} \{e' \in \mathcal{E}_i \mid e' \prec e\}$ de réseau causal associé $\mathcal{C}_{j-1}$. $\mathcal{C}_{j-1}$ n'admet que l'extension potentielle $e_j = e$ puisque tout autre événement $e'' \notin \mathcal{E}_{j-1}$ vérifie $e \prec e''$.

Soit ρ_{j-1}, la séquence adoptée pour produire les événements dans $\mathcal{E}_{j-1}$. Toute séquence ρ_i est telle que $\rho_i = \rho_{j-1} \cdot e_j \cdot \rho_k$, avec ρ_k constitué des événements descendant de e_j (avec $j + k = i$). La séquence ρ_j dont le dernier événement est sans concurrent est toujours préfixe de ρ_i et $Max(\mathcal{C}_j) = B$ (avec $\mathcal{E}_j \stackrel{\text{def}}{=} \mathcal{E}_{j-1} \cup \{e_j\}$).

Cas général où $|E| > 1$ **:**

B étant un état charnière, $\exists e_j \in E$ tel que $\forall e_k \in E\backslash\{e_j\}$, on a $d_{k,j} > 0$ dans toute temporisation valide du processus $\mathcal{E}_j \stackrel{\text{def}}{=} \{e \in \mathcal{E}_i \mid \exists b \in B, e \prec b\}$. Les événements dans $\mathcal{E}_j\backslash\{e_j\}$ précèdent toujours e_j (définition 9) ; tout autre événement $e_k \in \mathcal{E}_i\backslash\mathcal{E}_j$ vérifie $e_j \prec e_k$. Toute séquence ρ_i a donc la forme $\rho_i = \rho_{j-1} \cdot e_j \cdot \rho_k$, avec : ρ_{j-1} la séquence empruntée pour produire les événements dans $\mathcal{E}_j\backslash\{e_j\}$ et ρ_k constitué des événements descendant de e_j ($j + k = i$). Un préfixe $\rho_j \stackrel{\text{def}}{=} \rho_{j-1} \cdot e_j$ définit le réseau causal $\mathcal{C}_j$ associé à $\mathcal{E}_j$ tel que $Max(\mathcal{C}_j) = B$. □

2. Un événement e est *parent* d'un événement e' si $e^\bullet \cap {}^\bullet e' \neq \emptyset$ avec $e \in \{e_0\} \cup [e']\backslash\{e'\}$.

Un état charnière constitue un état où les horloges des extensions potentielles du processus sont initialisées en même temps (transitions toutes nouvellement sensibilisées). De plus, tout entrelacement réalisable sur l'ensemble des événements du processus se termine par ce même état du réseau temporel.

Proposition 4. *Soit $\mathcal{E}_k$, un processus constituant une alternative pour un processus réalisable $\mathcal{E}_i$ dans un état charnière, et soient $e_j \in \mathcal{E}_i$ et $e_l \in \mathcal{E}_k$ des choix en conflit. Si e_l n'est pas identifié dans $\mathcal{C}_i$, alors le processus alternatif $\mathcal{E}_k$ n'est pas réalisable.*

Démonstration. Autrement dit, est-ce possible qu'une pré-condition d'un choix alternatif à un événement $e_j \in \mathcal{E}_i \backslash \{e_i\}$ soit produite tardivement après le dernier événement e_i du processus $\mathcal{E}_i$? D'après la définition 9, e_j ne peut se produire après e_i. Au moment où e_i est produit, tout choix alternatif à e_j et non encore identifié possède au moins une pré-condition créée par un certain événement concurrent avec e_i et réalisable après e_i. Comme un tel événement n'existe pas pour le processus $\mathcal{E}_i$ dans un état charnière, une telle pré-condition et un tel choix alternatif n'existent pas non plus.

De plus, comme tout événement produit après e_i en constitue un descendant causal, un choix alternatif à e_i n'admet pas de pré-condition créée par un événement concurrent produit après e_i.

En conclusion, un choix non identifié par un processus dans un état charnière n'est pas réalisable dans un processus qui lui est alternatif. □

4.3. Obtention des processus alternatifs

Deux états charnières *équivalents* correspondent à un même état du réseau temporel. Le dépliage étant obtenu par calcul de différents processus, le développement d'un processus se termine (processus maximal) soit lorsque ce processus est dans un état charnière équivalent à un autre précédemment atteint (par un préfixe de ce même processus ou d'un processus alternatif), soit lorsque l'état atteint est sans extension possible (marquage mort). Cette restriction, qui impose que les processus maximaux d'un dépliage complet soit dans un état charnière, est justifiée par deux raisons :

– d'une part, l'identification des choix alternatifs permettant de produire les processus alternatifs en dépend (proposition 4). En effet, pour rendre complet un dépliage (définition 6), tout événement sensibilisé par un état intermédiaire d'un processus mais réalisable dans un processus alternatif doit être pris en compte, à l'instar du calcul des séquences composant un graphe d'état ;

– d'autre part, un état non charnière n'a pas une représentation simple ni économique dans une sémantique non séquentielle : tout comme la caractérisation d'un processus (cf. sous-sections 3.3 et 4.1), un système qui exprime l'ensemble des contraintes temporelles entre des événements, à savoir les extensions potentielles et leurs événements parents, est de nature disjonctive, et la mémorisation des différentes classes d'états visitées lors du développement des processus peut alors être explosive.

Un premier processus réalisable maximal ($\mathcal{E}_1$) peut être obtenu arbitrairement en produisant à chaque itération une nouvelle extension possible. Soit $\mathcal{E}_i$ un processus réalisable : $Conf(\mathcal{E}_i) \stackrel{\text{def}}{=} \bigcup_{e_k \in \mathcal{E}_i} Conf_i(e_k)$ est l'ensemble des choix alternatifs désactivés en réalisant $\mathcal{E}_i$. Si $\mathcal{E}_i$ est un processus maximal, tout processus alternatif $\mathcal{E}_j$ se distingue de $\mathcal{E}_i$ en réalisant des choix propres (cf. proposition 2) incluses dans $Conf(\mathcal{E}_i)$. D'après le

théorème 1, ses choix sont activés par leur préfixe commun $\mathcal{E}_i \cap \mathcal{E}_j$ et sont contenus dans l'ensemble $PE(\mathcal{E}_i \cap \mathcal{E}_j)$. Comme un processus alternatif $\mathcal{E}_j$ n'est pas connu *a priori*, il devra être généré à partir d'un sous-ensemble X_j de choix identifiés dans $Conf(\mathcal{E}_i)$. Ce sous-ensemble $X_j \subseteq \mathcal{E}_j$ constitue la *racine de choix* à partir de laquelle le processus $\mathcal{E}_j$ se spécialise vis-à-vis de $\mathcal{E}_i$: $\forall e_j \in \mathcal{E}_j \backslash \mathcal{E}_i, \exists x_j \in X_j \; t.q. \; x_j \preceq e_j$. Ainsi, tout nouveau processus alternatif d'un dépliage sera calculé à partir d'un ancien processus $\mathcal{E}_i$ en réalisant précisément un sous-ensemble X_j de choix non encore expérimentés précédemment.

Soit $X_j \in \mathbb{P}(Conf(\mathcal{E}_i))$ tel que pour tout $x_a, x_b \in X_j$ (avec $x_a \neq x_b$), $\neg(x_a \sharp x_b)$ est vérifié : $[X_j]$ est un processus. Si X_j n'est pas un sous-ensemble de choix d'un processus maximal obtenu précédemment au cours du dépliage, alors il peut déterminer un nouveau processus $\mathcal{E}_j$. Un nouveau processus maximal $\mathcal{E}_j$ est réalisable si tous les choix X_j peuvent être produits au cours du développement de ce processus. Pour *forcer* les choix X_j, il suffit d'éviter que toute pré-condition dans ${}^\bullet X_j \stackrel{\text{def}}{=} \bigcup_{x_j \in X_j} {}^\bullet x_j$ ne soit consommée par un événement non inclus dans X_j au cours de l'extension du processus $\mathcal{E}_j$. Bien entendu, pour le premier processus $\mathcal{E}_1$ développé au cours d'un dépliage, cet ensemble de choix à forcer est vide.

Définition 10. *Soient un préfixe de dépliage $TUnf = << \mathcal{B}, \mathcal{E}, \mathcal{F} >, \lambda >$ et $\mathcal{E}_i \in \overline{\mathcal{E}}$.* $Alt(\mathcal{E}_i) \stackrel{\text{def}}{=} \{X_j \in \mathbb{P}(Conf(\mathcal{E}_i)) \mid (\nexists \mathcal{E}_k \in \overline{\mathcal{E}} \; t.q. \; X_j \subseteq \mathcal{E}_k) \wedge ([X_j]$ *est un processus*$)\}$. *Un* nouveau *processus alternatif $\mathcal{E}_j$ à $\mathcal{E}_i \in \overline{\mathcal{E}}$ est déterminé par la réalisation d'un sous-ensemble de choix $X_j \in Alt(\mathcal{E}_i)$.*

La fonction `Processus_maximal` permet de produire un nouveau processus maximal alternatif à $\mathcal{E}_i$ en ne produisant que deux catégories d'événements : soit les événements déjà contenus dans $\mathcal{E}_i$ s'ils ne sont pas en conflit avec les choix X_j, soit les événements X_j et leurs descendants causaux. Ainsi, en rejetant les choix alternatifs $Conf(\mathcal{E}_i) \backslash X_j$, une conclusion déterministe pourra être tirée sur la faisabilité d'un processus alternatif contenant X_j. Lorsque le processus alternatif n'est pas réalisable sous ces contraintes, la fonction retourne un ensemble vide.

Dans la fonction `Processus_maximal`, $\mathcal{H}$ désigne l'ensemble des états charnières identifiés au cours du calcul des différents processus ; $\mathcal{H}$ et $Min(\mathcal{O})$ sont des variables globales. Le cas particulier du premier processus réalisable $\mathcal{E}_1$ est pris en compte si les extensions potentielles sont produites librement, c.-à-d. lorsque le paramètre $\mathcal{E}_i = \emptyset$.

Proposition 5. *La fonction `Processus_maximal` retourne un ensemble non vide si et seulement si cet ensemble est un processus maximal réalisable contenant X_j.*

Démonstration. La fonction `Processus_maximal` ne peut retourner un ensemble non vide que si l'une des conditions des lignes 5 ou 7 est vérifiée.

Les pré-conditions ${}^\bullet[X_j]$ (incluses dans $\mathcal{B}_i$) sont préservées pour ne produire que les événements $[X_j]$. Par conséquent, un événement dans $[X_j]$ restera toujours sensibilisé tant que tous les événements $[X_j]$ ne seront pas produits.

Quand le retour s'effectue par la ligne 5, le processus $\mathcal{E}_j$ obtenu n'admet plus d'extension potentielle ($\mathcal{E}_j$ est maximal), et donc, tous les événements X_j sont nécessairement produits dans $\mathcal{E}_j$.

Le retour s'effectue par la ligne 7 lorsque $\mathcal{E}_j$ est dans un état charnière équivalent : $\mathcal{E}_j$ est maximal. Admettons qu'il existe $x_j \in [X_j]$ sensibilisé et non produit dans $\mathcal{E}_j$. Selon la

```
1  début
2      E_j := ∅ ; B_j := Min(O);
3      tant que ∃e_j ∈ PE(E_j) t.q.
       (E_i = ∅ ∨ (e_j ∈ E_i, •e_j ∩ •X_j = ∅) ∨ (∃x ∈ X_j, x ≼ e_j)) ∧ (E_j ∪ {e_j} est réalisable)
       faire
4          E_j := E_j ∪ {e_j};
5          si PE(E_j) = ∅ alors retourner E_j;
6          sinon si Max(C_j) est un état charnière alors
7              si Max(C_j) admet un équivalent dans H alors retourner E_j;
8              sinon H := H ∪ {Max(C_j)} ;
9          fin
10     fin
11     retourner ∅;
12 fin
```

Fonction `Processus_maximal` (*événements* X_j, *processus* $\mathcal{E}_i$)

définition 9 d'un état charnière, il existe un parent $e_j \in \mathcal{E}_j$ qui est commun à tout élément de $PE(\mathcal{E}_j)$ tel que tout événement concurrent avec e_j le précède obligatoirement dans le temps. Or, évidemment $e_j \wr x_j$, et donc x_j précède obligatoirement e_j dans le temps. L'événement x_j n'étant pas encore produit, il s'agit d'une contradiction. En conclusion, tout $x_j \in [X_j]$ est nécessairement produit avant d'atteindre un état charnière. □

En pratique, au lieu d'obtenir $\mathcal{E}_j$ à partir de $Min(\mathcal{O})$, le calcul pourra se faire plus rapidement en partant du plus grand préfixe réalisable de $\mathcal{E}_i$ tel qu'aucune pré-condition dans $^\bullet X_j$ ne soit encore consommée.

La procédure `Prefixe_complet` permet d'obtenir le préfixe complet du dépliage temporel. Elle commence d'abord par calculer $\mathcal{E}_1$ après création de l'état initial $Min(\mathcal{O})$. Ensuite, à partir des ensembles de choix alternatifs $Alt(\mathcal{E}_1)$, tous les processus alternatifs maximaux de $\mathcal{E}_1$ sont produits. Il en est de même pour chaque processus maximal ajouté au dépliage $\overline{\mathcal{E}}$ jusqu'à ce qu'il n'y en ait plus de nouveau.

Théorème 2. *Si la procédure* `Prefixe_complet` *se termine, alors elle fournit un résultat* $\overline{\mathcal{E}}$ *qui couvre l'espace d'état du réseau temporel.*

Démonstration. Si tout processus réalisable du réseau temporel peut être décrit par les processus dans $\overline{\mathcal{E}}$, alors le préfixe de dépliage correspondant à $\overline{\mathcal{E}}$ est complet. Soit $\mathcal{E}$, un processus réalisable quelconque du réseau temporel (en considérant que les événements sont nommés indépendamment d'une implémentation). Il existe trois possibilités en comparant $\mathcal{E}$ avec un processus maximal quelconque $\mathcal{E}_i \in \overline{\mathcal{E}}$:

1) $\mathcal{E} \subseteq \mathcal{E}_i$: $\mathcal{E}$ est donc décrit par un processus dans $\overline{\mathcal{E}}$;

2) $\mathcal{E}_i \subset \mathcal{E}$: un préfixe de $\mathcal{E}$ est décrit par $\mathcal{E}_i$. Les événements du suffixe $\mathcal{E} \backslash \mathcal{E}_i$ sont produits à partir d'un état charnière équivalent à un état charnière intermédiaire $Max(\mathcal{C}'_j)$ d'un processus $\mathcal{E}_j \in \overline{\mathcal{E}}$. Puisque ces états sont identiques dans le réseau temporel, les comportements qui en dérivent sont également identiques. Par extrapolation, les événements $\mathcal{E} \backslash \mathcal{E}_i$ peuvent être assimilés totalement ou partiellement à $\mathcal{E}_j \backslash \mathcal{E}'_j$, ce qui ramène aux items 1 à 3 ;

3) $\mathcal{E}$ et $\mathcal{E}_i$ ne sont pas en relation d'inclusion : $\mathcal{E}$ constitue alors un processus alternatif à $\mathcal{E}_i$. Puisque la procédure `Prefixe_complet` identifie toute partie intiale de

processus alternatif à $\mathcal{E}_i$, il existe donc un processus $\mathcal{E}_j \in \overline{\mathcal{E}}$ tel que $\mathcal{E} \subseteq \mathcal{E}_i$ (item 1) ou $\mathcal{E}_i \subset \mathcal{E}$ (item 2).

Ainsi, tout processus réalisable du réseau temporel décrit des états empruntés par les processus $\overline{\mathcal{E}}$. □

```
1  début
2      Etablir Min(O) à partir de m0 ; H := {Min(O)};
3      E1 := Processus_maximal(∅, ∅);
4      Ē := {E1} ; i := 1 ; n := 1;
5      répéter
6          pour chaque Xj ∈ Alt(Ei) faire
7              Ej := Processus_maximal(Xj, Ei);
8              si Ej ≠ ∅ alors
9                  n := n + 1;
10                 Ē := Ē ∪ {Ej};
11             fin
12         fin
13         i := i + 1;
14     jusqu'à i > n ;
15 fin
```

Procédure `Prefixe_complet`

Un processus réalisable du réseau temporel peut contenir une infinité d'événements sans pour autant qu'un état intermédiaire soit charnière. La procédure `Prefixe_complet` s'exécutera alors indéfiniment dans ce cas.

Théorème 3. *Pour un réseau temporel borné, la procédure `Prefixe_complet` se termine en produisant un préfixe complet fini si un nombre fini d'événements est toujours produit entre deux états charnières consécutifs* [3] *dans toute exécution infinie.*

Démonstration. Un réseau temporel borné admet un nombre fini de classes d'état. Il existe donc un nombre fini de classes d'équivalence d'états charnières et d'états sans extension potentielle.

Puisqu'un nombre fini d'événements séparent deux états charnières consécutifs et qu'il existe un nombre fini de classes d'états charnières possibles, toute exécution infinie du réseau admet un préfixe fini (c.-à-d. composé d'un nombre fini d'événements) de processus dans un état charnière équivalent à un état précédemment emprunté ; tout état charnière est obligatoirement emprunté d'après la proposition 3. Ainsi, tout processus maximal dans $\overline{\mathcal{E}}$ qui est préfixe d'une exécution infinie est fini.

Toute exécution infinie du réseau admettant un préfixe de processus dans un état charnière, le nombre de préfixes de processus terminés dans un état charnière est borné par le nombre fini de classes d'équivalence d'état possibles. De même, il existe un nombre fini d'états sans extension potentielle qui borne le nombre possible de processus terminés par un marquage mort.

3. *consécutif* signifie que tout état intermédiaire entre deux états charnières est non charnière.

En somme, $\overline{\mathcal{E}}$ est composé d'un nombre fini de processus maximaux, chaque processus étant fini. La procédure `Prefixe_complet` se termine donc en fournissant un résultat $\overline{\mathcal{E}}$ couvrant l'espace d'état. □

Une implémentation de l'algorithme de dépliage pourrait intégrer un test pour détecter tout marquage non sauf, pour un réseau temporel supposé normalement sauf. Ce test devra exploiter la structure de réseau d'occurrence obtenu parallèlement aux calculs des processus afin d'établir les relations de concurrence, ainsi que les contraintes temporelles entre les événements du processus en cours de développement.

5. Un exemple de dépliage

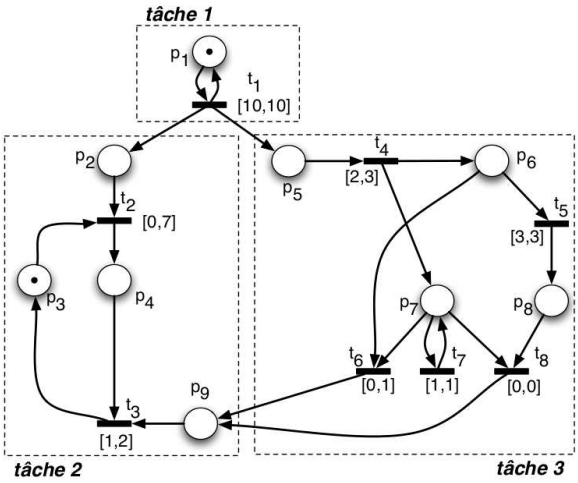

Figure 2. *Exemple de réseau temporel*

Le réseau temporel de la figure 2 modélise un système temps réel constitué de trois parties : une tâche périodique de réveil (tâche 1) et deux tâches d'application (tâches 2 et 3). Les deux tâches d'application, qui sont de durée d'exécution variable, ont la même période de réveil. La fin de la tâche 2 dépend de celle de la tâche 3.

Bien que le réseau temporel soit sauf, son modèle sous-jacent est non sauf : sans la contrainte de délais $[10, 10]$ associée à la transition t_1, la tâche 1 (tâche productrice) peut s'exécuter indéfiniment avant les autres, rendant les places p_2 et p_6 non bornées.

Le dépliage temporel est donné à la figure 3. L'ensemble $\overline{\mathcal{E}}$ est composé des six processus maximaux réalisables obtenus dans cet ordre (selon la mise en œuvre spécifique) :

– $\mathcal{E}_1 = \{e_1, e_2, e_3, e_4, e_6, e_8\}$ avec : $Max(\mathcal{C}_1) = \{b_{10}, b_{11}, b_{12}, b_{13}\}$ et $Conf(\mathcal{E}_1) = \{e_5, e_7\}$. $Alt(\mathcal{E}_1) = \{\{e_5\}, \{e_7\}, \{e_5, e_7\}\}$ détermine $\mathcal{E}_2$ par $X_2 = \{e_7\}$ et $\mathcal{E}_3$ par $X_3 = \{e_5, e_7\}$; $\{e_5\}$ ne détermine pas de processus réalisable ;

– $\mathcal{E}_2 = \{e_1, e_2, e_3, e_4, e_7, e_9, e_{11}\}$ avec : $Max(\mathcal{C}_2) = \{b_{16}, b_{11}, b_{12}, b_{13}\}$ et $Conf(\mathcal{E}_2) = \{e_5, e_6, e_{10}\}$. $Alt(\mathcal{E}_2) = \{\{e_5\}\}$; $\{e_5\}$ ne détermine pas de processus

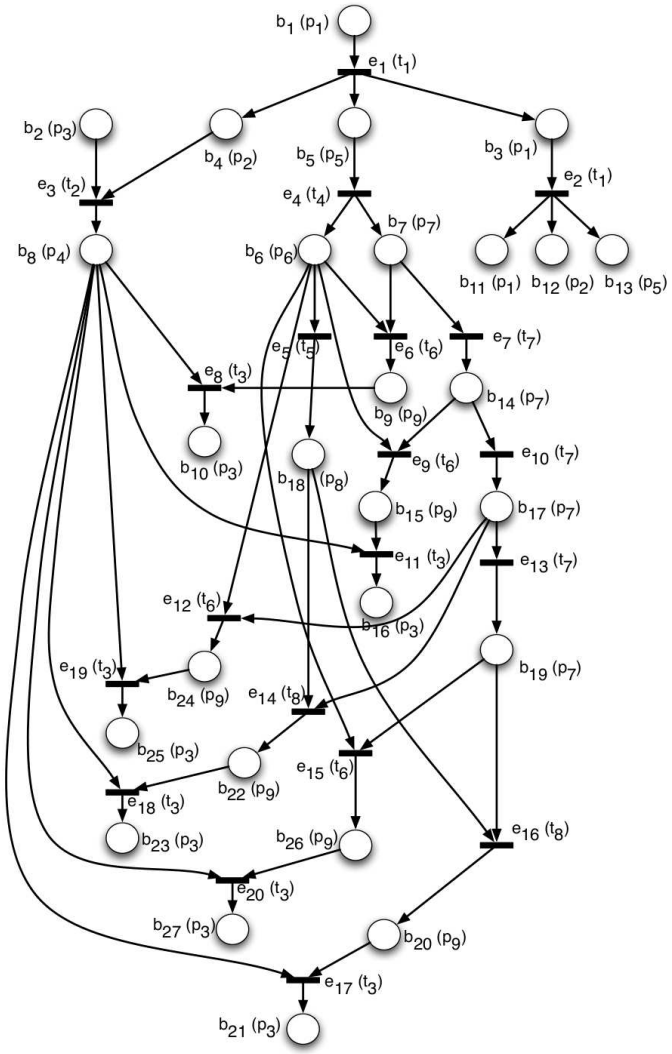

Figure 3. *Préfixe complet de dépliage temporel du réseau de la figure 2*

réalisable ;

– $\mathcal{E}_3 = \{e_1, e_2, e_3, e_4, e_5, e_7, e_{10}, e_{13}, e_{16}, e_{17}\}$ avec : $Max(\mathcal{C}_3) = \{b_{21}, b_{11}, b_{12}, b_{13}\}$ et $Conf(\mathcal{E}_3) = \{e_6, e_9, e_{12}, e_{14}, e_{15}\}$. $Alt(\mathcal{E}_3) = \{\{e_{14}\}, \{e_{12}\}, \{e_{15}\}\}$ détermine $\mathcal{E}_4$ par $X_4 = \{e_{14}\}$, $\mathcal{E}_5$ par $X_5 = \{e_{12}\}$, et $\mathcal{E}_6$ par $X_6 = \{e_{15}\}$;

– $\mathcal{E}_4 = \{e_1, e_2, e_3, e_4, e_5, e_7, e_{10}, e_{14}, e_{18}\}$ avec : $Max(\mathcal{C}_4) = \{b_{23}, b_{11}, b_{12}, b_{13}\}$ et $Conf(\mathcal{E}_4) = \{e_6, e_9, e_{13}\}$. $Alt(\mathcal{E}_4) = \emptyset$;

– $\mathcal{E}_5 = \{e_1, e_2, e_3, e_4, e_7, e_{10}, e_{12}, e_{19}\}$ avec : $Max(\mathcal{C}_5) = \{b_{25}, b_{11}, b_{12}, b_{13}\}$ et $Conf(\mathcal{E}_5) = \{e_5, e_6, e_9, e_{13}\}$. $Alt(\mathcal{E}_5) = \emptyset$;

– $\mathcal{E}_6 = \{e_1, e_2, e_3, e_4, e_7, e_{10}, e_{13}, e_{15}, e_{20}\}$ avec : $Max(\mathcal{C}_6) = \{b_{27}, b_{11}, b_{12}, b_{13}\}$ et $Conf(\mathcal{E}_6) = \{e_5, e_6, e_9, e_{12}\}$. $Alt(\mathcal{E}_6) = \emptyset$.

Chacun des états de ces processus est charnière et équivalent à l'état charnière intermédiaire $\{b_2, b_3, b_4, b_5\}$ généré par le préfixe de processus $\{e_1\}$.

La validation des spécifications temporelles d'un système temps réel se basera sur les contraintes temporelles canoniques [1] obtenues pour chaque processus maximal, en utilisant les délais possibles entre les événements.

6. Conclusion

Un dépliage est constitué d'un ensemble de processus alternatifs qui se distinguent en réalisant des choix différents d'événéments en conflit. Chaque processus maximal du dépliage peut être calculé suivant une approche *ordre partiel* qui évite l'énumération exhaustive des classes d'état accessibles. La représentation d'un préfixe fini couvrant l'espace d'état du réseau temporel repose sur une énumération d'états charnières, un état charnière pouvant être interprété comme un point de synchronisation au niveau global des processus concurrents qui composent le système modélisé.

Bien que la finitude d'un dépliage complet soit conditionnée par une occurrence multiple des états charnières dans toute exécution infinie, cette restriction pourrait ne pas jouer pour certaines applications où la condition de terminaison est autrement spécifiée par le contexte ; par exemple, si le délai ou la taille minimale d'un processus, avant de vérifier une propriété, sont connus. C'est également le cas des systèmes concurrents centralisés où un seul ordonnanceur régit les exécutions des processus.

La sémantique d'ordre partiel des réseaux temporels a été formalisée uniquement pour les modèles saufs [1].Toutefois, la méthode proposée s'applique sans difficulté particulière aux réseaux temporels non saufs lorsque la multi-sensibilisation d'une transition est interprétée suivant la sémantique non déterministe d'un dépliage ordinaire [9] : les instances multiples d'une transition sont considérées indépendantes, et non ordonnées les unes par rapport aux autres. En revanche, la prise en compte d'autres sémantiques (telles que la sémantique dite *standard* et la stratégie *première sensibilisée, première tirée*) est moins évidente et est envisageable dans des travaux futurs.

Les contraintes temporelles quantitatives associées aux processus du préfixe obtenu peuvent servir à produire une analyse d'ordonnançabilité d'un système temps réel. De plus, une extension de la méthode aux politiques préemptives d'ordonnancement, très utilisées, constituerait aussi une autre perspective. Ceci pourrait reposer sur une extension sémantique des réseaux temporels prenant en compte ce type de système [14, 15].

7. Bibliographie

[1] Tuomas Aura and Johan Lilius. Time processes for time Petri nets. In *ICAréseau temporel*, volume 1248 of *LNCS*, pages 136–155, 1997.

[2] Bernard Berthomieu and Michel Diaz. Modeling and verification of time dependent systems using time Petri nets. *IEEE Trans. Software Eng.*, 17(3) :259–273, 1991.

[3] Bernard Berthomieu and François Vernadat. State class constructions for branching analysis of time Petri nets. In *TACAS*, pages 442–457, 2003.

[4] Antonio Cerone and Andrea Maggiolo-Schettini. Time-based expressivity of time Petri nets for system specification. *Theoretical Computer Science*, 216(1-2) :1 – 53, 1999.

[5] T. Chatain and C. Jard. Complete finite prefixes of symbolic unfoldings of safe time Petri nets. *Springer-Verlag*, 4024 :125–145, 2006.

[6] Francis Cottet, Joëlle Delacroix, Claude Kaiser, and Zoubir Mammeri. *Ordonnancement temps réel - Cours et exercices corrigés* . Hermès, janvier 2000.

[7] Javier Esparza and Keijo Heljanko. *Unfoldings : A Partial-Order Approach to Model Checking (Monographs in Theoretical Computer Science. An EATCS Series)*. Springer Publishing Company, Incorporated, 1 edition, 2008.

[8] J. Engelfriet. Branching processes of Petri nets. *Acta Informatica*, 28(6) :575– 591, june 1991.

[9] Javier Esparza, Stefan Römer, and Walter Vogler. An improvement of McMillan's unfolding algorithm. In Tiziana Margaria and Bernhard Steffen, editors, *TACAS*, volume 1055 of *Lecture Notes in Computer Science*, pages 87–106. Springer, 1996.

[10] Johan Lilius. Efficient state space search for time Petri nets. In *Electronic Notes in Theoretical Computer Science*. Springer-Verlag, 1999.

[11] K. L. McMillan. Using unfolding to avoid the state explosion problem in the verification of asynchronous circuits. In *Proc. Fourth Workshop on Computer-Aided Verification*, volume 663, pages 164–177, Montreal, Canada, june 1992. Springer-Verlag.

[12] K. L. McMillan. A technique of state space search based on unfolding. *Form. Methods Syst. Des.*, 6(1) :45–65, 1995.

[13] P. M. Merlin and D. J. Faber. Recoverability of communication protocol - implications of theoretical study. *IEEE Transactions on Communications*, 24 :1036–1043, September 1976.

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

[15] Olivier H. Roux and Didier Lime. Time Petri nets with inhibitor hyperarcs. formal semantics and state space computation. In Jordi Cortadella and Wolfgang Reisig, editors, *ICAréseau temporel*, volume 3099 of *Lecture Notes in Computer Science*, pages 371–390. Springer, 2004.

[16] Alexei Semenov and Alexandre Yakovlev. Verification of asynchronous circuits using time Petri net unfolding. In *DAC '96 : Proceedings of the 33rd annual conference on Design automation*, pages 59–62, New York, NY, USA, 1996. ACM.

[17] Médésu Sogbohossou and David Delfieu. Unfolding of time Petri nets for quantitative time analyses. In *TiSto'09 : International Workshop on Timing and Stochasticity in Petri nets and other models of concurrency*, Paris, France, June 2009.

[18] Antti Valmari. Stubborn sets for reduced state space generation. In *Proceedings of the 10th International Conference on Application and Theory of Petri Nets, 1989, Bonn, Germany ; Supplement*, pages 1–22, 1989.

[19] François Vernadat, Pierre Azéma, and François Michel. Covering step graph. In *In ICAréseau temporel*, pages 516–535. Springer-Verlag, 1996.

[20] Enrico Vicario. Static analysis and dynamic steering of time-dependent systems. *IEEE Trans. Softw. Eng.*, 27(8) :728–748, 2001.

[21] Pierre Wolper and Patrice Godefroid. Partial-order methods for temporal verification. In Eike Best, editor, *CONCUR*, volume 715 of *Lecture Notes in Computer Science*, pages 233–246. Springer, 1993.