Estimation d'état pour les réseaux de Petri synchronisés temporisés labellisés *

Mouna Gaouar[1], Rabah Ammour[1], Isabel Demongodin[1], and Dimitri Lefebvre[2]

[1] Aix-Marseille Université, CNRS, LIS, Marseille, France.
{mouna.gaouar, rabah.ammour, isabel.demongodin} @lis-lab.fr
[2] Université Le Havre Normandie, GREAH, Le Havre, France.
dimitri.lefebvre@univ-lehavre.fr

Abstract

Cet article traite de l'estimation de l'état courant dans les Réseaux de Petri Synchronisés Temporisés Labellisés, une sous-classe des Réseaux de Petri Synchronisés Temporisés avec Sorties [6]. Pour représenter l'espace d'états, un Graphe de Classes Synchronisé est introduit, puis transformé en un Automate de Classes d'états à Intervalles, un automate fini dans lequel le temps continu est abstrait au moyen d'événements discrets appelés ticks. Un algorithme de construction d'un observateur pour cette classe d'automates est proposé, permettant de déterminer l'état courant du système à partir de séquences d'observations temporisées. Cette approche offre une base formelle pour des applications allant de la commande à la vérification et à la détection de fautes dans les systèmes à événements discrets temporisés.

1 Introduction

Les Systèmes Cyber-Physiques (SCP) sont des systèmes dans lesquels des composants logiciels interagissent avec des processus physiques à travers un réseau de communication. De plus en plus présents dans les environnements industriels, ces systèmes sont fréquemment modélisés à l'aide de formalismes de Systèmes à Événements Discrets (SED), notamment pour des applications telles que le diagnostic de pannes [16] et la détection de cyberattaques [12].

Dans le cadre des SED, les réseaux de Petri constituent l'un des formalismes les plus couramment utilisés pour la modélisation. Les techniques d'estimation d'état pour les réseaux de Petri sont essentielles dans les SCP pour des applications variées, telles que la vérification d'opacité [13], la validation de systèmes [3], ou encore le contrôle supervisé en présence d'attaques [15].

Ce travail s'intéresse à l'estimation d'état dans les modèles de SCP qui intègrent à la fois des contraintes temporelles et une synchronisation avec l'environnement externe. Une méthode est proposée pour une classe spécifique de réseaux de Petri intégrant ces deux aspects.

L'estimation d'état dans les Réseaux de Petri Temporisés et les Réseaux de Petri Temporels (RdPT) a fait l'objet de nombreux travaux, les premières approches reposant sur la construction du Graphe de Classes d'États (GCE) [4]. Le lien établi dans [11] entre les RdPT et les automates temporisés permet d'exploiter des techniques d'estimation développées pour ces derniers [1, 5].

Dans [8], un estimateur d'état est proposé en tenant compte à la fois des observations et des contraintes temporelles. Par ailleurs, plusieurs extensions du GCE ont été développées pour améliorer les capacités d'estimation : le Graphe de Classes d'États Modifié (GCEM) [2], ainsi que ses variantes adaptées aux Réseaux de Petri Temporels Labellisés et aux Réseaux de Petri Interprétés Temporels (RdPIT) [3]. Plus récemment, [10] a introduit un GCEM observé pour

*Ce travail a été partiellement financé par l'Agence Nationale de la Recherche (ANR) dans le cadre du projet ANR-22-CE10-0002.

une estimation d'état en ligne, tandis que [14] a proposé des graphes de classes parallèles afin de construire des observateurs dits critiques exploitant des observations en temps réel.

Cependant, ces approches ne prennent pas en compte de manière explicite la synchronisation avec les événements externes. Les travaux de [11, 10, 14] se limitent aux RdPT, tandis que l'approche proposée dans [3] pour les RdPIT repose sur une actualisation du GCEM à chaque observation. Il s'agit d'une procédure en ligne, où les classes (et non les états) compatibles avec l'observation courante sont identifiées de manière itérative. Cette stratégie, bien qu'efficace pour certaines applications, ne permet pas de générer une structure d'observation réutilisable, comme un observateur, nécessaire pour des analyses hors ligne telles que la vérification d'opacité ou la diagnosticabilité.

Afin de répondre à ces limitations, une méthode d'estimation d'état courant [7] est introduite dans le cadre des Réseaux de Petri Synchronisés Temporisés Labellisés ou Timed Labeled Synchronized Petri Nets (TLSPN), une sous-classe des Réseaux de Petri Synchronisés Temporisés avec Sorties [6], dans laquelle les sorties sont modélisées par des labels associés aux transitions (voir Section 2.1). L'espace d'états d'un TLSPN borné est représenté par un Graphe de Classes d'États Synchronisé ou Synchronized State Class Graph (SynSCG) [6], une extension du GCE tenant compte des événements d'entrée externes (voir Section 2.2). Ce graphe est ensuite transformé en un automate fini, appelé Automate de Classes d'États à Intervalle (ACEI), introduit en Section 3. Cette transformation, inspirée de [5], repose sur une abstraction du temps continu par des événements discrets appelés "ticks". Un algorithme de construction d'un observateur pour le ACEI est proposé (Section 4), permettant l'estimation d'état dans le RdPSTL initial.

2 Réseaux de Petri Synchronisés Temporisés Labellisés et Graphe de Classes d'États Synchronisé

Un Réseau de Petri est une structure $N = \langle P, T, Pre, Post\rangle$ où P est un ensemble de places, T est un ensemble de transitions, $Pre : P \times T \to \mathbb{N}$ et $Post : P \times T \to \mathbb{N}$ sont les matrices d'incidence *pre* et *post* qui spécifient les poids des arcs dirigés des places vers les transitions et inversement.

2.1 Réseaux de Petri Synchronisés Temporisés Labellisés

On note $\mathbb{Q}^+$ (respectivement $\mathbb{R}^+$) l'ensemble des nombres rationnels (respectivement réels) positifs ou nuls, et $\mathbb{Q}_+^*$ (respectivement $\mathbb{R}_+^*$) l'ensemble des nombres rationnels (respectivement réels) strictement positifs.

Definition 1. *Un* Réseau de Petri Synchronisé Temporisé Labellisé *(TLSPN) est une structure* $\mathcal{N} = \langle N, E, f, Q, g, D\rangle$ *où :*

- $\langle N\rangle$ *est un réseau de Petri ;*
- E *est un alphabet d'événements d'entrée externes ;*
- $f : T \to E_\lambda = E \cup \{\lambda\}$ *est une fonction qui associe à chaque transition* $t \in T$ *un événement dans* E_λ*, où* λ *est l'événement permanent;*
- Q *est un ensemble de labels, et* $Q_\varepsilon = Q \cup \{\varepsilon\}$ *où* ε *est un symbole utilisé pour indiquer l'absence de label ;*
- $g : T \to Q_\varepsilon$ *est une fonction de labelisation;*

- $D : T \to \mathbb{Q}^+$ *est une fonction qui associe à chaque transition* t_j *une durée* d_j.

Le marquage d'un TLSPN est un vecteur $M \in \mathbb{N}^{|P|}$ qui assigne à chaque place un entier naturel, où $|P|$ désigne le nombre de places du réseau. Le marquage est composé d'un vecteur de *marquage réservé* $M^R \in \mathbb{N}^{|P|}$ et d'un vecteur de *marquage non réservé* $M^{NR} \in \mathbb{N}^{|P|}$ tels que $M = M^R + M^{NR}$. On note $M(p_i)$ le marquage de la place $p_i \in P$ et M_0 le marquage initial. Un TLSPN marqué est noté $\langle \mathcal{N}, M_0 \rangle$.

Pour prendre en compte le temps, une horloge locale notée o_{t_j} est associée à chaque transition t_j, et l'état d'un TLSPN marqué $\langle \mathcal{N}, M_0 \rangle$ est un couple $s = (M, \mathcal{O})$ composé du marquage des places et des valeurs des horloges locales telles que $\mathcal{O} \in \{\mathbb{R}^+ \cup \{\sharp, +\infty\}\}^{|T|}$, où $\sharp$ indique l'état des horloges associées aux transitions non activées, et $|T|$ est le nombre de transitions du réseau. $\mathcal{O}$ est un vecteur dont l'élément o_{t_j} contient la valeur de l'horloge associée à la transition $t_j \in T$. À l'état $s = (M, \mathcal{O})$, la transition t_j est *validée* si le marquage non réservé vérifie $M^{NR} \geq Pre(\cdot, t_j)$. Si t_j est validée, alors $o_{t_j} = +\infty$. La transition t_j peut alors être activée par l'occurrence de l'événement externe qui lui est associé $f(t_j) = e \in E_\lambda$. Autrement dit, si t_j est validée et que l'événement $f(t_j) = e$ se produit, t_j est instantanément activée et son horloge locale est initialisée à d_j et commence à décompter. Lorsque o_{t_j} atteint 0, la transition t_j est franchie et le label $g(t_j)$ est émis. La sémantique de service adoptée dans ce travail est une sémantique mono-serveur pour l'activation et le franchissement des transitions. Lorsqu'une transition t_j est activée, les jetons nécessaires à son franchissement sont réservés. Le marquage global reste inchangé, mais le nouveau marquage réservé M'^R et le marquage non réservé M'^{NR} sont calculés comme suit : $M'^R = M^R + Pre(\cdot, t_j)$ et $M'^{NR} = M^{NR} - Pre(\cdot, t_j)$.

Si plusieurs horloges locales atteignent 0 à l'état $s = (M, \mathcal{O})$, alors toutes leurs transitions associées sont franchies en une seule étape appelée une Séquence Élémentaire de Franchissement $\sigma^f \in 2^T$, menant à l'état s', noté $s[\sigma^f\rangle s'$. Il est également possible que plusieurs transitions validées associées au même événement externe $e \in E_\lambda$ soient activées lors de l'occurrence de e à l'état s, menant à l'état s'. Cela est appelé Séquence Élémentaire d'Activation $\sigma^a \in 2^T$, notée $s[\sigma^a\rangle s'$. À l'état s, l'écoulement du temps $\kappa \in \mathbb{R}^+$, sans franchissement ou activation de transition, mène à l'état s' noté $s[\kappa\ \rangle\ s'$.

Un *conflit synchronisé* se produit lorsque deux transitions ou plus sont validées mais que le marquage non réservé n'est pas suffisant pour les activer toutes à l'occurrence de leur événement externe associé. Autrement dit, si $t, t' \in T$ sont deux transitions en conflit structurel et que $f(t) = f(t') = e$, un *conflit synchronisé* survient si à l'état $s = (M, \mathcal{O})$, t' et t sont toutes deux validées et que l'événement e se produit alors que $(M^{NR} \geq Pre(., t)) \wedge (M^{NR} \geq Pre(., t')) \wedge (M^{NR} < Pre(., t) + Pre(., t'))$. Nous adoptons dans ce cas une politique de résolution de conflits basée sur des priorités, en définissant des priorités entre les transitions en conflit structurel. On note $t \gg t'$ lorsque t a la priorité sur t' ; à l'occurrence de leur événement d'entrée externe, t est activée et les jetons sont réservés pour son franchissement tandis que t' devient non validée.

Example 1. *La Figure 1 présente un exemple de TLSPN marqué* $\langle \mathcal{N}_1, M_0 \rangle$ *où* $M_0 = M^{NR} = [2, 0, 0, 0, 0]$, $E = \{e_1, e_2\}$, $Q = \{a, b\}$ *et* $t_1 \gg t_4$, *i.e., la transition* t_1 *est prioritaire par rapport à* t_4. △

Une *trajectoire temporisée* notée ρ sur un TLSPN $\mathcal{N}$, d'un état initial s_0 vers un état s_n, est une séquence de longueur $n \in \mathbb{N}$ alternant états et événements temporisés définie comme suit :

$$\rho = s_0 \xrightarrow{\omega_1, z_1} s_1 \xrightarrow{\omega_2, z_2} \cdots \xrightarrow{\omega_n, z_n} s_n \tag{1}$$

où :

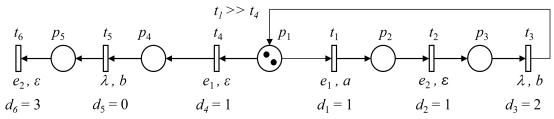

Figure 1: Réseau de Petri Temporisé Synchronisé Labellisé $\mathcal{N}_1$

- s_k est le $k^{\text{ème}}$ état dans la trajectoire ρ avec $k = 1, \ldots, n$;
- ω_k est soit un événement interne, c'est-à-dire un franchissement de transitions avec émission de labels ($q \in 2^{Q_\varepsilon}$), soit un événement externe ($e \in E_\lambda$) ;
- $z_k \in \mathbb{R}^+$ est un horodatage représentant le temps écoulé entre l'état s_{k-1} et l'état s_k.

Ainsi, $s_{k-1} \xrightarrow{\omega_k, z_k} s_k$ désigne l'occurrence de ω_k après z_k unités de temps depuis s_{k-1}, qui vérifie les conditions d'activation et de franchissement de $\mathcal{N}$, et conduit à l'état s_k.

Un état s_n d'un TLSPN marqué est *accessible* s'il existe une trajectoire temporisée $\rho = s_0 \xrightarrow{\omega_1, z_1} s_1 \xrightarrow{\omega_2, z_2} \cdots \xrightarrow{\omega_n, z_n} s_n$, $k = 1, \ldots, n$ telle que s_0 est l'état initial. De plus, un marquage M est *accessible* s'il existe un état accessible $s = (M, \mathcal{O})$. Bien que les états d'un TLSPN soient des couples composés de marquages et de valeurs d'horloges, nous considérons qu'un TLSPN est *borné* si le nombre de ses marquages accessibles est fini.

Proposition 1. *L'ensemble des marquages accessibles d'un TLSPN $\mathcal{N} = \langle N, E, f, Q, g, D \rangle$ est un sous-ensemble de l'ensemble des marquages accessibles de sa structure autonome N ayant le même marquage initial.*

Preuve. Soit $\mathcal{N}$ un TLSPN et N_A sa structure autonome, tous deux ayant un marquage initial M_0. Considérons un marquage M accessible du TLSPN $\mathcal{N}$. Si M est accessible dans $\mathcal{N}os$, alors il existe une trajectoire temporisée ρ de l'état $s_0 = (M_0, \mathcal{O}_0)$ à l'état $s = (M, \mathcal{O})$ telle que $s_0[\rho\rangle s$. Soit T^* l'ensemble de toutes les séquences formées de transitions dans T. On définit ainsi $\varrho \in T^*$, une séquence de transitions composée de toutes les transitions franchies dans la trajectoire ρ, dans le même ordre. ϱ vérifie donc $M = M_0 + C \cdot \tilde{\varrho}$, où $C = Post - Pre$ est la matrice d'incidence du réseau et $\tilde{\varrho}$ est le vecteur de franchissement correspondant à la séquence ϱ. Dans ce cas, il est possible d'appliquer la même séquence de franchissements ϱ dans la structure autonome N_A pour atteindre M à partir de M_0.

Étant donné que la structure autonome du TLSPN permet le franchissement libre des transitions, i.e., sans dépendance vis-à-vis des entrées externes ou des contraintes temporelles, la séquence ϱ peut être appliquée. Cette construction étant valable pour tout marquage M accessible du TLSPN, il en découle que tous les marquages accessibles d'un TLSPN sont inclus dans les marquages accessibles de sa structure autonome. □

2.2 Graphe de Classes d'États Synchronisé

Le Graphe de Classes d'États Synchronisé est une extension du Graphe de Classes d'États qui capture le comportement synchronisé et temporisé des TLSPN bornés. Il repose sur les classes d'états synchronisées, qui constituent une abstraction des états accessibles d'un TLSPN.

Une classe d'états synchronisée est un couple $C = (M, \Theta)$ où M est un marquage et Θ est un domaine d'horloges appelé domaine d'activation et de franchissement. Une classe d'états C est un ensemble d'états d'un TLSPN tel que tout état $s \in C$ est défini par un marquage M, identique à celui de la classe C, et par un vecteur d'horloges locales $\mathcal{O}$ satisfaisant les contraintes

d'activation et de franchissement définies par Θ. La procédure de construction du SynSCG est décrite en détail dans [6].

Definition 2. *Un graphe de classes d'états synchronisé est un graphe orienté $\mathcal{G} = (C_\mathcal{G}, A_\mathcal{G})$ où $C_\mathcal{G}$ est un ensemble de classes d'états (nœuds du graphe) et $A_\mathcal{G}$ est un ensemble d'arcs reliant ces nœuds. Chaque arc $A_{ij} \in A_\mathcal{G}$ est de la forme $(C_i, \omega, \Delta_{ij}, C_j)$, où $C_i, C_j \in C_\mathcal{G}$ sont respectivement les classes source et cible, $\omega \in E_\lambda \cup 2^T \times 2^{Q_\epsilon}$ représente soit un événement externe, soit un événement interne accompagné de ses labels, et Δ_{ij} est un intervalle de temps $[l_{ij}, u_{ij}]$ ou $[l_{ij}, +\infty)$, avec $l_{ij}, u_{ij} \in \mathbb{Q}^+$.*

Le SynSCG fournit une représentation finie de l'espace d'états d'un TLSPN borné dans le cadre de la sémantique mono-serveur.

Soit $\mathcal{N}$ un TLSPN borné avec un état initial s_0, et soit $\mathcal{G}$ son graphe de classes. Un état s est dit accessible s'il existe une classe $C \in C_\mathcal{G}$ telle que $s \in C$. On note $R(\mathcal{N})$ l'ensemble des états accessibles de $\mathcal{N}$.

Example 2. *La figure 2 illustre le SynSCG $\mathcal{G}$ du TLSPN $\mathcal{N}_1$. Initialement, les transitions t_1 et t_4 sont validées et associées à l'événement e_1, qui peut survenir à tout instant. Ainsi, la classe initiale C_0 possède la contrainte d'activation $0 \leq \theta_1^a$. Un arc sortant étiqueté par e_1 mène alors à la classe C_1, où t_1 et t_4 sont activées et les deux jetons de p_1 sont réservés. Ces deux transitions nécessitant 1 unité de temps pour être franchies, C_1 possède les contraintes de franchissement $\theta_1^f = 1$ et $\theta_4^f = 1$. Comme les deux transitions sont franchies simultanément, l'arc sortant est étiqueté par les transitions $[t_1, t_4]$, le label a est émis lors du franchissement de t_1, et l'intervalle de temps considéré est $[1, 1]$. L'arc mène à la classe C_2, dans laquelle t_2 et t_5 sont sensibilisées, ce qui est représenté par les contraintes d'activation associées aux événements λ et e_2. L'événement λ, se produisant toujours, prévaut et est pris en compte immédiatement avec l'intervalle de temps $[0, 0]$, menant à la classe C_3. Le raisonnement se poursuit de manière analogue pour le reste du SynSCG.* △

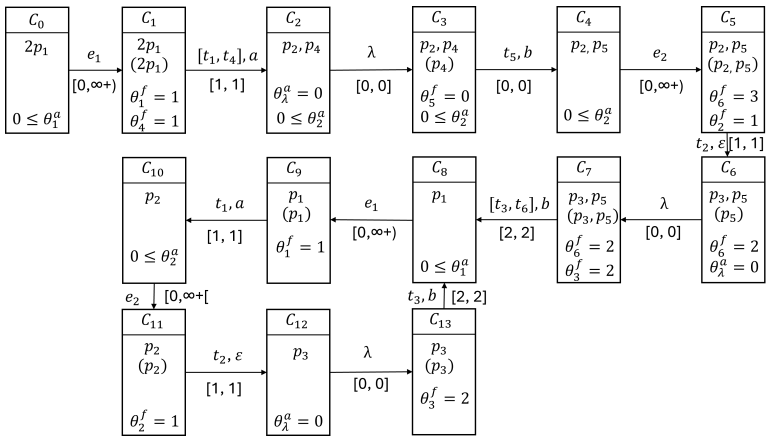

Figure 2: Graphe de Classes d'États Synchronisé $\mathcal{G}$ du TLSPN $\mathcal{N}_1$

3 Automates de Classes d'États à Intervalles

3.1 Abstraction du temps

L'écoulement continu du temps dans les TLSPN peut être représenté de manière discrète comme une suite de "tick" événements. Un tick événement agit comme un marqueur temporel dénotant le passage d'une certaine durée de temps. Cette approche s'inspire des automates à ticks [9], où le temps progresse par incréments fixes d'une unité. Plutôt que des incréments fixes qui discrétisent le temps, nous considérons dans ce travail des évènements qui permettent d'abstraire cet écoulement et de le représenter par deux types d'événements temporisés : ϵ et $(r-\epsilon)$. L'événement ϵ correspond à l'écoulement d'une quantité de temps infinitésimale. En revanche, $(r-\epsilon)$ complète l'écoulement d'un temps $r \in \mathbb{Q}_+^*$ sur le système, puisqu'on a $\epsilon + (r-\epsilon) = r$. Nous appelons *incrément temporel* la durée r.

Dans les TLSPN, chaque transition est associée à une durée constante. Toutefois, lors de la construction du SynSCG, les contraintes temporelles sur le franchissement des transitions sont exprimées sous forme d'inégalités linéaires, et les arcs du graphe sont étiquetés par des intervalles de temps Δ. Cette différence entre la durée constante d'une transition dans la structure du TLSPN et les intervalles de temps dans la dynamique s'explique par la synchronisation avec des événements externes. Contrairement aux RdPT non synchronisés, où les transitions peuvent se déclencher dès que leurs contraintes temporelles sont satisfaites, un TLSPN impose une synchronisation avec les événements d'entrée externes.

Comme ces événements peuvent se produire à des instants arbitraires, le franchissement d'une transition dépend non seulement de sa durée propre, mais aussi du moment où l'événement externe de synchronisation survient. Par conséquent, le système peut rester dans une même classe pendant des durées variables, selon le moment d'activation d'une transition par un événement externe. Nous appelons *temps de séjour* l'intervalle pendant lequel le système demeure dans une classe donnée.

Soit $C_i \in C_\mathcal{G}$ une classe synchronisée ayant k arcs sortants A_{ij}, $j \in 1, \ldots, k$, chacun associé à un intervalle de temps $\Delta_{ij} = [l_{ij}, u_{ij}]$. Le temps de séjour ϕ_i dans la classe C_i est défini par l'intervalle $\phi_i = [0, u_i]$, où $u_i = \max u_{ij}$, ou bien $\phi_i = [0, +\infty)$ si ce maximum est infini. On appelle *temps de séjour maximal* la borne supérieure u_i lorsque ϕ_i est fini.

Proposition 2. *Soit $C_\mathcal{G}^\phi = \{C_i \in C_\mathcal{G} : \phi_i = [0, u_i], u_i \in \mathbb{Q}^+\}$ l'ensemble des classes d'états synchronisées ayant un temps de séjour fini. Il existe au moins un nombre rationnel strictement positif $r \in \mathbb{Q}_+^*$ qui est un multiple commun des temps de séjour maximaux de toutes les classes $C_i \in C_\mathcal{G}^\phi$, c'est-à-dire : $\exists r \in \mathbb{Q}_+^*, \forall C_i \in C_\mathcal{G}^\phi : u_i = n_i \times r, n_i \in \mathbb{N}$.*

Preuve. Le temps de séjour maximal u_i de chaque classe avec un temps de séjour fini est un nombre rationnel qui peut s'écrire sous la forme $u_i = \frac{v_i}{y_i}$ avec $v_i \in \mathbb{N}$ et $y_i \in \mathbb{N}^*$. Soit $y^* = ppcm(y_0, y_1, \ldots, y_k), k = |C_\mathcal{G}^\phi|$ le plus petit commun multiple des dénominateurs. On peut donc écrire $u_i = \frac{v_i \times y^*}{y_i} \times \frac{1}{y^*}$. Comme $y^* \in \mathbb{N}$, on a $\frac{1}{y^*} \in \mathbb{Q}_+^*$. De plus, y^* étant un multiple de y_i, le quotient $\frac{y^*}{y_i} \in \mathbb{N}$, donc $\frac{v_i \cdot y^*}{y_i} \in \mathbb{N}$. Ainsi, tout temps de séjour maximal peut s'écrire $u_i = n_i \times r, r \in \mathbb{Q}_+^*, n_i \in \mathbb{N}$ avec $r = \frac{1}{y^*}$ et $n_i = \frac{v_i \times y^*}{y_i}$.

□

En s'appuyant sur la Proposition 2, si une classe C a un temps de séjour fini $\phi = [0, u]$, avec $u \in \mathbb{Q}^+$, cet intervalle peut être décomposé en une partition de régions temporelles selon un certain $r \in \mathbb{Q}_+^*$:

$$\phi = [0,0], \dot{\cup}, (0,r), \dot{\cup}, [r,r], \dot{\cup}, (r,2r), \dot{\cup}, \cdots, \dot{\cup}, (u-r,u), \dot{\cup}, [u,u] \tag{2}$$

où $\dot{\cup}$ dénote l'opérateur d'union disjointe.

Cette décomposition est motivée par les deux événements ticks adoptés, ϵ et $(r-\epsilon)$, et permet de découper chaque classe à temps de séjour fini en $2 \cdot \frac{u}{r} + 1$ sous-classes.

3.2 Construction d'un ACEI

Soit I l'ensemble des intervalles ouverts et fermés, de formes respectives $(u, u+r)$, $u \in \mathbb{Q}^+$ et $[u,u], u \in \mathbb{Q}^+$. Pour un temps de séjour fini donné $\phi = [0,u], u \in \mathbb{Q}^+$, on note $I_\phi = \{[0,0],(0,r),[r,r],(r,r+r),\cdots,(u-r,u),[u,u]\}$ l'ensemble des intervalles dans la décomposition de ϕ selon l'Équation 2.

Definition 3. *Un* état étendu *est une paire $x = (C,\tau)$ où C est une classe d'état synchronisée et $\tau \in I_\phi$ est un sous-intervalle du temps de séjour ϕ.*

Un état étendu est une "sous-classe" ou un sous-ensemble des états $s \in C$ dans lesquels le TLSPN peut se trouver après un écoulement de temps k appartenant à l'intervalle τ dans la classe C. Soit $A(C)$ l'ensemble des transitions activées dans la classe C, c'est-à-dire toutes les transitions associées à une contrainte de tir. Un état s d'un TLSPN appartient à un état étendu x, c'est-à-dire $s \in x = (C,\tau)$, si et seulement si $s \in C$ et $\exists k \in \tau$ tel que $\forall t_j \in A(C)$, le changement de variable $(\theta^f_{t_j} + k)$ vérifie les contraintes de tir associées à la classe C.

Definition 4. *Soit un TLSPN marqué $\langle \mathcal{N}, M_0\rangle$ où $\mathcal{N} = \langle N, E, f, Q, g, D\rangle$, et son graphe de classes d'états synchronisé $\mathcal{G} = (C_\mathcal{G}, A_\mathcal{G})$. L'Automate de Classes d'États à Intervalles (ACEI) associé à $\mathcal{G}$ est un automate fini $\mathcal{A}_{syn} = \langle X, \Sigma, \delta, x_0\rangle$, tel que :*

- *$X = C_\mathcal{G} \times I$ est un ensemble fini d'états étendus ;*
- *$\Sigma = E_\lambda \cup 2^{Q_\varepsilon} \cup \{\epsilon, r-\epsilon\}$ est un ensemble fini d'événements ;*
- *$\delta : X \times \Sigma \to X$ est une fonction de transition ;*
- *$x_0 = (C_0, [0,0]) \in X$ est l'état initial étendu.*

Par la suite, on note $\bar{\omega}$ la projection de ω sur $E_\lambda \cup 2^{Q_\varepsilon}$, c'est-à-dire $\bar{\omega} = \omega \uparrow_{\{E_\lambda \cup 2^{Q_\varepsilon}\}}$. Introduisons maintenant la fonction de transition δ qui dicte les changements d'états étendus en fonction de l'occurrence d'un événement $\mu \in \Sigma$ à l'état étendu $x = (C_i, \tau)$. Cette fonction $\delta(x,\mu)$ est définie de sorte que $\delta(x,\mu) = x'$ si et seulement si :

- $\mu \in E_\lambda \cup 2^{Q_\varepsilon}$ et il existe un arc du SynSCG $(C_i, \omega, \Delta_{ij}, C_j) \in A_\mathcal{G}$ tel que $\mu = \bar{\omega}$ et $\tau \subseteq \Delta_{ij}$. Dans ce cas, $x' = (C_j, [0,0])$. Autrement dit, un événement peut survenir à l'état étendu x seulement s'il peut avoir lieu dans le SynSCG pendant l'intervalle τ. Dans le SynSCG, cet événement mène à une nouvelle classe d'état synchronisée, et ainsi, le nouvel état étendu x' contient cette nouvelle classe avec l'intervalle de temps initial $[0,0]$;
- $\mu = \epsilon$ et $\tau = [k,k], k \in \mathbb{Q}^+$ tel que $k+r \in \phi_i$. Alors, $x' = (C_i, (k, k+r))$, ce qui correspond à un écoulement infinitésimal de temps ;
- $\mu = r-\epsilon$ et $\tau = (k, k+r), k \in \mathbb{Q}^+$. Dans ce cas, si ϕ_i est fini, alors $x' = (C_i, [k+r, k+r])$, sinon, $x' = (C_i, [k,k])$, ce qui correspond à un écoulement de temps de $(r-\epsilon)$ unités.

Remarquons que pour une classe C avec un temps de séjour infini, c'est-à-dire $\phi = [0, +\infty)$, il existe deux états étendus associés : $x = (C, [0,0])$ et $x' = (C, (0,r))$. Dans une classe à temps de séjour infini, peu importe la durée écoulée, le système reste dans la même classe. Cette

décomposition en états étendus permet de représenter le temps infini de séjour d'une classe en construisant un cycle composé d'événements alternés ϵ et $r-\epsilon$, comme défini dans la fonction de transition δ. Ainsi, l'écoulement infini du temps est représenté dans l'ACEI par une séquence infinie d'événements d'horloge $\epsilon, r-\epsilon, \epsilon, r-\epsilon, \ldots$ alternant entre x et x'.

On dit qu'un événement $\mu \in \Sigma$ en x mène à x' si $\delta(x,\mu) = x'$, ce que l'on note $x[\mu\rangle x'$. L'ensemble des événements possibles à l'état étendu $x_i \in X$ est $F(x_i) = \{\mu \in \Sigma : \delta(x_i,\mu) \in X\}$, c'est-à-dire l'ensemble des événements pour lesquels la fonction de transition δ est définie. Soit $\sigma = \mu_0 \cdots \mu_i \cdots \mu_n, n \in \mathbb{N}, \mu_i \in \Sigma$ une séquence ordonnée d'événements. σ est une séquence réalisable depuis un état étendu $x_i \in X$ si $\delta(\cdots\delta(\delta(x_i,\mu_0),\mu_1)\cdots,\mu_n) = x_j \in X$, et on note $x_i[\sigma\rangle x_j$, aussi noté $\delta^*(x_i,\sigma) = x_j$. Un état étendu $x_j \in X$ d'un ACEI est *atteignable* depuis $x_i \in X$ s'il existe une séquence $\sigma \in \Sigma^*$ telle que $x_i[\sigma\rangle x_j$. L'Algorithme 1 présente les étapes de construction d'un ACEI à partir d'un SynSCG donné.

Algorithm 1 Construction du ACEI à partir du SynSCG

Entrée: Un SynSCG $\mathcal{G} = (C_\mathcal{G}, A_\mathcal{G})$, un incrément de temps $r \in \mathbb{Q}_+^*$
Sortie: L'ACEI associé $\mathcal{A}_{syn} = \langle X, \Sigma, \delta, x_0\rangle$

Initialiser $x_0 \leftarrow (C_0, [0,0])$, $X \leftarrow \{x_0\}$
pour chaque classe C_i dans $C_\mathcal{G}$ **faire**
  $X \leftarrow X \cup \{x_0^i = (C_i, [0,0])\}$
  marquer tous les états étendus de X comme "nouveaux"
fin pour
tant que $\exists x \in X$ marqué "nouveau" **faire**
  sélectionner $x_j^i = (C_i, \tau_j) \in X$ marqué "nouveau"
  si $\tau_j = [k,k], k \in \mathbb{Q}^+ \wedge k + r \in \phi_i$ **alors**
    $X \leftarrow X \cup \{x_{j+1}^i = (C_i, (k, k+r))\}$
    $\delta(x_j^i, \epsilon) \leftarrow x_{j+1}^i$
  sinon si $\tau_j = (k, k+r), k \in \mathbb{Q}^+$ **alors**
    si $\phi_i \neq [0, +\infty)$ **alors**
      $X \leftarrow X \cup \{x_{j+1}^i = (C_i, [k+r, k+r])\}$
      $\delta(x_j^i, r-\epsilon) \leftarrow x_{j+1}^i$
    sinon
      $\delta(x_j^i, r-\epsilon) \leftarrow (C_i, [0,0])$
    fin si
  fin si
  marquer x_{j+1}^i comme "nouveau"
  pour chaque arc $A_{ih} = (C_i, \omega, \Delta_{ih}, C_h) \in A_\mathcal{G}$ **faire**
    pour chaque état étendu $x_j^i = (C_i, \tau_j) \in X$ **faire**
      si $\tau_j \cap \Delta_{ih} \neq \emptyset$ **alors**
        $\delta(x_j^i, \bar{\omega}) \leftarrow x_0^h = (C_h, [0,0])$
      fin si
    fin pour
  fin pour
  dé-marquer x_j^i
fin tant que

Soit $C_\mathcal{G}^\infty = \{C_i \in C_\mathcal{G} : \phi_i = [0, +\infty)\}$ l'ensemble des classes d'états ayant un temps de séjour infini. Rappelons que $C_\mathcal{G}^\phi$ désigne l'ensemble des classes d'états ayant un temps de séjour fini. Considérons un SynSCG tel que $c_\phi = |C_\mathcal{G}^\phi|$ (respectivement $c_\infty = |C_\mathcal{G}^\infty|$), le nombre de classes

d'états avec un temps de séjour fini (respectivement infini) et un incrément de temps $r \in \mathbb{Q}_+^*$. Le nombre d'états étendus du ACEI associé au SynSCG est donné par $|X| = \sum_{C_i \in C_{\mathcal{G}}^{\phi}} \frac{2u_i}{r} + 1 + 2c_\infty$. Par conséquent, la complexité en espace du ACEI est $O(\frac{2c_\phi \bar{u}}{r} + 2c_\infty)$, où $\bar{u} = \max_{C_i \in C_{\mathcal{G}}^{\phi}} \{u_i\}$.

Example 3. *Considérons l'application de l'Algorithme 1 au SynSCG $\mathcal{G}$ de la Figure 2 pour construire son ACEI $\mathcal{A}$, comme illustré dans la Figure 3. Comme toutes les durées associées aux transitions sont des entiers, un incrément de temps naturel est $r = 1$. L'algorithme commence par créer tous les états étendus $(C_i, [0, 0])$ pour toutes les classes $C_i \in C_{\mathcal{G}}$. La classe d'état C_0 a un temps de séjour infini, ainsi il existe une boucle entre $x_0 = (C_0, [0, 0])$ et $x_1 = (C_0, (0, 1))$. L'événement externe e_1 survient à C_0 dans un intervalle de temps $[0, +\infty)$. Ainsi, les intervalles τ_0 et τ_1 de x_1 et x_0 permettent tous deux l'occurrence de e_1, c'est-à-dire $\delta(x_0, e_1) = \delta(x_1, e_1) = x_2 = (C_1, [0, 0])$. Après les événements successifs de type tick ϵ et $(1 - \epsilon)$, correspondant à l'écoulement d'une unité de temps, on atteint $x_4 = (C_1, [1, 1])$, où le tir des transitions t_1 et t_4 émet le label a et conduit à $x_5 = (C_2, [0, 0])$.* △

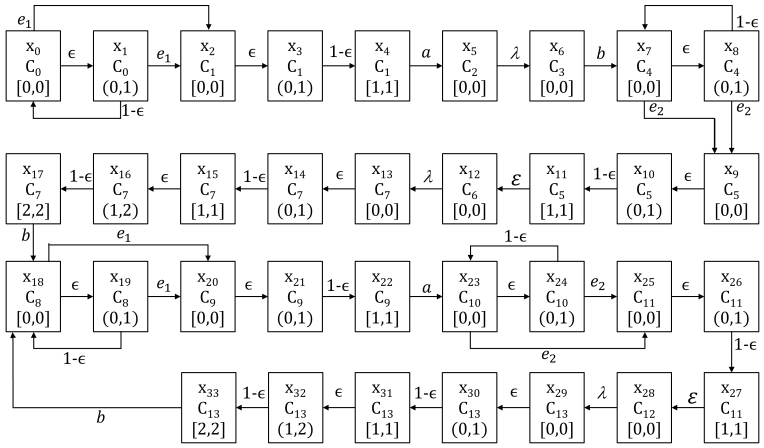

Figure 3: Automate de Classes d'États à Intervalles $\mathcal{A}$ du TLSPN $\mathcal{N}_1$

Proposition 3. *Dans un TLSPN borné marqué $\langle \mathcal{N}, M_0 \rangle$, $\forall s, s' \in R(\mathcal{N})$ tels que $s \xrightarrow{\omega', 0} s', \omega' \in E_\lambda \cup 2^T \times 2^{Q_\varepsilon}$, si $s \in x$, alors $\exists \mu \in \Sigma$ tel que $\mu = \bar{\omega}'$ et $\delta(x, \mu) = x'$ avec $s' \in x'$.*

Preuve. Considérons $x \in X$ tel que $s \in x = (C, \tau)$. Un changement d'état $s \xrightarrow{\omega', 0} s'$ correspond, dans le SynSCG, à $C \xrightarrow{\omega', \Delta} C', \tau \subseteq \Delta$. En considérant la fonction de transition δ du ACEI, si $(C, \omega', \Delta, C') \in A_{\mathcal{G}}$, alors une transition $x = (C, \tau) \xrightarrow{\mu} x' = (C', [0, 0])$ est définie avec $\mu = \bar{\omega}'$. Comme s' est l'état atteint instantanément à partir de la classe C, il vérifie alors les contraintes de tir de C' dans l'intervalle $[0, 0]$. Par conséquent, $s' \in x'$. □

Considérons deux états atteignables s et s' d'un TLSPN, tels que l'occurrence d'un événement ω' en s mène à s' sans écoulement de temps. Si s est inclus dans un état étendu x du ACEI, il existe un événement du ACEI $\mu \in \Sigma$, correspondant à l'occurrence de l'événement $\bar{\omega}'$ dans le TLSPN, qui mène de x à x', où x' contient l'état s'.

Proposition 4. *Dans un TLSPN borné, soit l'état s_n atteint à partir de l'état s_{n-1} par une trajectoire temporisée $s_{n-1} \xrightarrow{\omega_n, z_n} s_n$. Si $s_{n-1} \in x$, alors $s_n \in x'$ tel que $\delta^*(x, \sigma\bar{\omega}_n) = x'$ et $\sigma = \epsilon(r-\epsilon)\epsilon(r-\epsilon)...$ de taille m, où $m = 2\frac{z_n}{r}$ si $z_n = k \times r$ $(k \in \mathbb{N})$, ou $m = 2\lfloor \frac{z_n}{r} \rfloor + 1$ sinon.*

Preuve. L'occurrence de ω_n dans un TLSPN, menant de s_{n-1} à s_n, correspond à des séquences dans l'ACEI selon la Proposition 3, via un événement $\bar{\omega}_n$. Considérons d'abord l'évolution temporelle associée à z_n à partir de s_{n-1}. L'ACEI est dans $x = (C, [0, 0])$ avec $s_{n-1} \in x$. Ensuite, un événement de tick ϵ est considéré, ce qui mène à $x_1 = (C, (0, r))$ et $\sigma = \epsilon$. Une fois r unités de temps écoulées, la transition $(r - \epsilon)$ est considérée, donc $\sigma = \epsilon(r - \epsilon)$. Ce processus se poursuit de la même manière jusqu'à ce qu'un état étendu $x_m = (C, \tau_m)$ soit atteint, où $z_n \in \tau_m$, $\sigma = \epsilon(r-\epsilon)\epsilon(r-\epsilon)...$ et $\delta^*(x, \sigma) = x_m$. Si z_n est un multiple de r, alors $\tau_m = [z_n, z_n]$, sinon $\tau_m = (r\lfloor \frac{z_n}{r} \rfloor, (r\lfloor \frac{z_n}{r} \rfloor + 1))$. Ensuite, $\bar{\omega}_n$ est faisable en x_m, donc $\delta(x_m, \bar{\omega}_n) = x'$. Par conséquent, $\delta^*(x, \sigma\bar{\omega}_n) = x'$. □

4 Observateur SCIA et estimation d'état

4.1 Conception de l'observateur

L'ensemble fini des événements Σ est partitionné en deux sous-ensembles disjoints : $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$, où Σ_o désigne l'ensemble des événements observables et Σ_{uo} celui des événements non observables.

L'hypothèse est que l'événement de tick ϵ est non observable. Cette hypothèse est motivée par des considérations pratiques : dans les systèmes réels, les capteurs ne permettent généralement pas de détecter des avancements de temps infiniment petits, ce qui rend ces événements indétectables en pratique. En outre, rendre ces événements observables introduirait une complexité inutile dans la modélisation. En revanche, l'événement de tick $(r - \epsilon)$, qui n'est pas infiniment petit, est considéré comme observable. La valeur de r peut être ajustée en fonction des besoins du modèle et des ressources de calcul disponibles, assurant ainsi un compromis entre réalisme et faisabilité de l'observation. Ainsi, on a $\epsilon \in \Sigma_{uo}$ et $(r - \epsilon) \in \Sigma_o$.

Les labels $q \in 2^Q$ sont supposées observables, tandis que le symbole ε est non observable. Concernant les événements externes, on considère une partition $E = E_o \dot{\cup} E_{uo}$, avec $\lambda \in E_{uo}$. On appelle *séquence observable faisable* toute séquence faisable $\sigma_{obs} \in \Sigma_o^*$.

Definition 5. *L'observateur d'un SCIA $\mathcal{A}_{syn} = \langle X, \Sigma, \delta, x_0 \rangle$ est une structure $\mathcal{A}_{obs} = \langle X_{obs}, \Sigma_o, \delta_{obs}, x_0^{obs} \rangle$, où :*

- *X_{obs} est un ensemble fini d'états $x_{obs} \in 2^X$;*
- *Σ_o est l'ensemble des événements observables ;*
- *$\delta_{obs} : X_{obs} \times (E_o \cup 2^Q \cup \{r - \epsilon\}) \to X_{obs}$ est la fonction de transition ;*
- *$x_0^{obs} \in X_{obs}$ est l'état initial de l'observateur.*

Les états de l'observateur sont des ensembles d'états étendus, reliés entre eux par des événements observables. De plus, pour tout état étendu x appartenant à un état de l'observateur x^{obs}, l'ensemble des états étendus accessibles depuis x via une séquence faisable d'événements non observables est également inclus dans x^{obs}. Ce phénomène est formalisé par l'ensemble $R_{uo}(x_i)$, défini comme l'ensemble des états atteignables à partir de x_i par une séquence d'événements non observables : $R_{uo}(x_i) = \{x \in X \mid \exists \eta \in \Sigma_{uo}^*,\ x_i[\eta\rangle x\}$.

Cet ensemble constitue une brique de base pour l'algorithme 2 de construction de l'observateur $\mathcal{A}_{obs}$ associé à un SCIA donné $\mathcal{A}_{syn}$. L'algorithme initialise l'ensemble des états X_{obs} par son état initial x_0^{obs} défini comme l'union de l'état initial du SCIA et de l'ensemble $R_{uo}(x_0)$, c'est-à-dire $x_0^{obs} = \{x_0\} \cup R_{uo}(x_0)$. L'algorithme étend ensuite itérativement X_{obs} en explorant les états nouveaux, en identifiant leurs successeurs via les transitions observables, et en calculant les états atteignables par une séquence d'évènements non-observables. Chaque nouvel état x_j^{obs} est ajouté à X_{obs} s'il n'est pas déjà présent, et la fonction de transition δ_{obs} est mise à jour. Le processus se poursuit jusqu'à stabilisation de l'ensemble des états.

La complexité en espace de l'observateur est doublement exponentielle en le nombre d'états étendus du SCIA. Elle est donnée par : $O(2^{|X|}) = O\left(2^{\frac{2c_\phi \bar{u}}{r} + 2c_\infty}\right)$ où $\bar{u} = \max_{C_i \in C_\mathcal{G}^\phi} \{u_i\}$.

4.2 Estimation d'état

Dans un TLSPN, une observation temporisée de taille $n \in \mathbb{N}$ est une séquence $\nu = (\nu_1, \kappa_1) \cdots (\nu_n, \kappa_n)$ d'événements temporisés observables $(\nu_i, \kappa_i) \in (E_o \cup 2^Q \cup \bar{\varepsilon}) \times \mathbb{R}^+$, avec $i \in 1, 2, \ldots, n$, où κ_i représente l'instant d'occurrence de l'événement ν_i. Le symbole $\bar{\varepsilon}$ indique qu'aucun label ni aucun événement externe n'a été observé. Plus précisément, l'horodatage κ_i correspond au temps absolu d'occurrence de l'événement ν_i, mesuré par une horloge globale dont la valeur croît de manière continue.

Algorithm 2 Construction de l'observateur SCIA

Entrée: Un ACEI $\mathcal{A}_{syn} = \langle X, \Sigma, \delta, x_0 \rangle$ et une partition des événements en observables et non observables $\Sigma = \Sigma_{uo} \cup \Sigma_o$
Sortie: Son observateur $\mathcal{A}_{obs} = \langle X_{obs}, \Sigma_o, \delta_{obs}, x_0^{obs} \rangle$

Initialiser $x_0^{obs} = \{x_0 \cup R_{uo}(x_0)\}$, $X_{obs} = \{x_0^{obs}\}$
Marquer x_0^{obs} comme "nouveau"
tant que $\exists x_i^{obs} \in X_{obs}$ marqué "nouveau" **faire**
  Sélectionner x_i^{obs} marqué "nouveau"
  pour chaque $x_k \in x_i^{obs}$ tel que $x_k \in X$ **faire**
    pour chaque événement $\mu \in F(x_k) \cap \Sigma_o$ **faire**
      Créer un état $x_j^{obs} = \{\delta(x_k, \mu)\} \cup R_{uo}(\delta(x_k, \mu))$
      si $x_j^{obs} \notin X_{obs}$ **alors**
        $X_{obs} \leftarrow X_{obs} \cup \{x_j^{obs}\}$
        Marquer x_j^{obs} comme "nouveau"
      fin si
      $\delta_{obs}(x_i^{obs}, \mu) \leftarrow x_j^{obs}$
    fin pour
  fin pour
  Dé-marquer x_i^{obs}
fin tant que

Soit $H : (E_o \cup 2^Q \cup \{\bar{\varepsilon}\}) \times \mathbb{R}^+ \to (\Sigma_o \cup \{\bar{\varepsilon}\})^*$ la fonction de correspondance qui associe à chaque événement temporisé observable une séquence correspondante d'événements observables du SCIA, telle que

$$H(\nu_i, \kappa_i) = (r - \epsilon)^{\lfloor \frac{\kappa_i}{r} \rfloor} \nu_i.$$

En supposant que le temps initial du système et le temps initial d'observation soient nuls, on

peut étendre cette fonction aux observations temporisées de taille n de la manière suivante :

$$H^*(\nu) = H(\nu_1, \kappa_1) \cdots H(\nu_i, \kappa_i - \kappa_{i-1}) \cdots H(\nu_n, \kappa_n - \kappa_{n-1}), \quad \text{pour } i \in \{1, \ldots, n\}.$$

À partir de ces notions, on peut établir que l'état courant s_n d'un TLSPN borné au moment de l'émission d'une séquence d'observations temporisées ν appartient à l'ensemble

$$CS(\nu) = \{s \in x, x \in X : x \in \delta_{obs}(H^*(\nu), x_0^{obs})\}.$$

Example 4. *Considérons le TLSPN $\mathcal{N}_1$ et son SCIA $\mathcal{A}$ avec $\Sigma_{uo} = \{\varepsilon, \epsilon, e_2, \lambda\}$ et $\Sigma_o = \{a, b, 1-\epsilon, e_1\}$. L'algorithme 2 retourne l'observateur représenté sur la Figure 4. Considérons une observation temporisée $\nu = (e_1, 0.3)(a, 1.3)$ qui correspond à la séquence d'événements observables $H^*(\nu) = (1-\epsilon)^{\lfloor 0.3 \rfloor} e_1 (1-\epsilon)^{\lfloor 1.3-0.3 \rfloor} a = e_1(1-\epsilon)a$. L'estimation de l'état courant pour $\mathcal{N}_1$ est $CS(\nu) = \{s \in \{x_5 = (C_2, [0,0]), x_6 = (C_3, [0,0])\}\}$.*

△

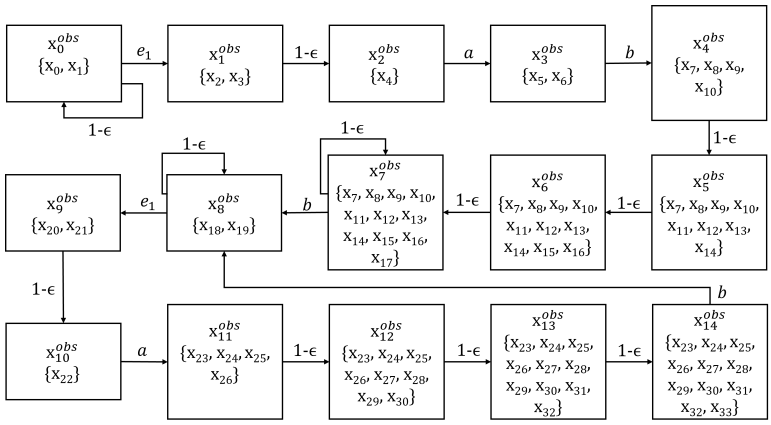

Figure 4: Observateur du SCIA $\mathcal{A}_{syn}$

5 Conclusion

Cet article propose une méthode d'estimation d'état pour les Réseaux de Petri Synchronisés Temporisés Labellisés via la transformation du Graphe des Classes d'États Synchronisés en un automate à états finis, appelé Automate de Classes d'État à Intervalles, dans lequel le temps continu est représenté par des événements discrets de type tick. Il est démontré que la construction d'un observateur d'état pour le SCIA permet d'estimer l'état courant du TLSPN associé.

Ce travail ouvre plusieurs perspectives de recherche dans le domaine des Réseaux de Petri Synchronisés Temporisés. Plus particulièrement, les recherches futures pourront exploiter l'observateur SCIA, en tant qu'abstraction finie du comportement temporel, pour la vérification de l'opacité d'état courant. De plus, l'étude de mécanismes de détection de cyber-attaques actives sur les systèmes cyber-physiques sera envisagée, en tirant parti de la capacité de l'observateur à reconstruire le comportement du système sous observabilité partielle.

References

[1] Rajeev Alur and David L Dill. A theory of timed automata. *Theoretical computer science*, 126(2):183–235, 1994.

[2] Francesco Basile, Maria Paola Cabasino, and Carla Seatzu. State estimation and fault diagnosis of labeled time Petri net systems with unobservable transitions. *IEEE Trans. on Automatic Control*, 60(4):997–1009, 2015.

[3] Francesco Basile and Luigi Ferrara. Validation of industrial automation systems using a timed model of system requirements. *IEEE Trans. on Control Systems Technology*, 31(1):130–143, 2022.

[4] Bernard Berthomieu and Miguel Menasche. An enumerative approach for analyzing Time Petri Nets. In *IFIP 9th World Computer Congress*, Paris, France, September 1983.

[5] Chao Gao, Dimitri Lefebvre, Carla Seatzu, Zhiwu Li, and Alessandro Giua. State estimation of timed automata under partial observation. *IEEE Trans. on Automatic Control*, 70(3):1981–1987, 2025.

[6] Mouna Gaouar, Rabah Ammour, Isabel Demongodin, and Dimitri Lefebvre. Timed output synchronized Petri nets and basics of synchronized state class graph. *IFAC-PapersOnLine*, 58(1):78–83, 2024. 17th IFAC Workshop on Discrete Event Systems (WODES).

[7] Mouna Gaouar, Rabah Ammour, Isabel Demongodin, and Dimitri Lefebvre. Current state estimation of timed labeled synchronized petri nets. *IEEE Control Systems Letters*, pages 1–1, 2025.

[8] Mohamed Ghazel, Armand Toguyéni, and Pascal Yim. State observer for DES under partial observation with time Petri nets. *Discrete Event Dynamic Systems*, 19:137–165, 2009.

[9] Jan Komenda and Dimitri Lefebvre. On tick automata for distributed timed DESs with synchronisations and minimal time constraints. *IFAC-PapersOnLine*, 56(2):8635–8640, 2023. 22nd IFAC World Congress.

[10] Liang Li, Mingxi Deng, Bin Liu, and Zhiwu Li. State estimation in labeled time Petri net systems using observed modified state class graph. *Information Sciences*, 656:119922, 2024.

[11] D. Lime and O. Roux. State class timed automaton of a time Petri net. In *10th International Workshop on Petri Nets and Performance Models.*, pages 124–133, 2003.

[12] Samuel Oliveira, André B. Leal, Marcelo Teixeira, and Yuri K. Lopes. A classification of cybersecurity strategies in the context of discrete event systems. *Annual Reviews in Control*, 56:100907, 2023.

[13] Tao Qin, Li Yin, Gaiyun Liu, Naiqi Wu, and Zhiwu Li. Strong current-state opacity verification of discrete-event systems modeled with time labeled Petri nets. *IEEE/CAA Journal of Automatica Sinica*, 12(1):54–68, 2025.

[14] Tao Qin, Li Yin, Naiqi Wu, and Zhiwu Li. Verification of current-state opacity in time labeled Petri nets with its application to smart houses. *IEEE Trans. on Automation Science and Engineering*, 21(4):7616–7628, 2024.

[15] Dan You, ShouGuang Wang, MengChu Zhou, and Carla Seatzu. Supervisory control of petri nets in the presence of replacement attacks. *IEEE Transactions on Automatic Control*, 67(3):1466–1473, 2021.

[16] J. Zaytoon and S. Lafortune. Overview of fault diagnosis methods for discrete event systems. *Annual Reviews in Control*, 37(2):308–320, 2013.