République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de L'**E**nseignement **S**upérieur et de la A **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI- **O**UZOU

FACULTE DE **G**ENIE **E**LECTRIQUE ET D' **I**NFORMATIQUE
DEPARTEMENT D'AUTOMATIQUE

Mémoire de Fin d'Etude de MASTER ACADEMIQUE

Spécialité : **Automatique**

Filière : **Commande des systèmes**

Présenté par
Juba CHALLAL
Hocine CHERFA

Mémoire dirigé par **Redouane KARA** et co-dirigé par **Karima TEBANI**

Thème

Commande des Systems à événements discrets à temps critiques : Application aux systèmes de commande en réseau

Mémoire soutenu publiquement le 17 Mai 2018 devant le jury composé de :

M Prénom NOM		
MAIDI Ahmed,	Professeur à l'UMMTO,	Président
M Prénom NOM		
KARA Redouane,	Professeur à l'UMMTO,,	Encadreur
M Prénom NOM		
TEBANI Karima,	MRB au CDTA,	Co-encadreur
M Prénom NOM		
HAMRI Hakima,	MCB à l'UMMTO,	Examinateur
M Prénom NOM		
NAIT ABDESLAME Aldjia,	MAA à l'UMMTO,	Examinateur

Remerciements

Ce mémoire est le résultat d'un travail de recherche de près de six mois. En préambule, on veut adresser tout nos profonds remerciements aux personnes avec lesquelles nous avons échangé et qui nous ont aidé pour la rédaction de ce mémoire.

En commençant par remercier tout d'abord Mme TEBANI Karima notre encadrant pour son aide précieuse et pour le temps qu'elle nous a consacré.

Merci au professeur KARA Redouane pour ces conseils qui nous ont guidés vers les bonnes vois.

Enfin, nous adressons nos plus sincères remerciements à nos familles, ainsi que tous nos proches et amis qui nous ont accompagné, aidé, soutenu et encouragé tout au long de la réalisation de ce mémoire.

A mes chers parents

A mes deux frères Mohend et Mourad

A ma grand-mère et mon oncle

A tous mes amis surtout Cherif, Milouze,

Fateh, Aissa et Hocine

A Anies

A toute ma famille

et tous ceux que j'aime

Juba

A mes chers parents

A mon frere et mes soeurs

A mes neuveus et nieces

A tous mes amis surtout Ramdane, Cherif,

Fateh, Ghiles et Juba

A toute ma famille

et tous ceux que j'aime

Hocine

Table des matières

Table des figures

Table des figures

Symboles

$\oplus$: Addition dans un dioïde.

$\otimes$: Multiplication dans un dioïde.

ε : Elément neutre pour la loi $\oplus$.

e : Elément neutre pour la loi $\otimes$.

$\top$: Plus grand élément dans un dioïde.

$\bot$: Plus petit élément dans un dioïde.

A^* : Etoile de Kleene de la matrice A($A= E \oplus A \oplus A^2 \oplus \ldots$).

a^* : Etoile de Kleene d'un scalaire ($a = e \oplus a \oplus a^2 \oplus \ldots$).

a^+ : Dérivée de l'étoile de Kleene ($a^+ = a \oplus a^2 \oplus \ldots$).

D : Dioïde.

$D^{n\times n}$: Dioïde matriciel.

$\wedge$: Borne inferieure dans un dioïde.

$\vee$: Borne supérieure dans un dioïde.

$\mathbb{R}_{max}$: Dioïde complet ($\mathbb{R} \cup \{-\infty, +\infty\}, max, +$), appelé aussi algèbre (max,+).

$\mathbb{R}_{min}$: Dioïde complet ($\mathbb{R} \cup \{-\infty, +\infty\}, min, +$), appelé aussi algèbre (min,+).

L_a : Produit à gauche par a, $L_a(x) = a \otimes x$.

R_a : Produit à droite par a, $R_a(x) = x \otimes a$.

$L_a^{\#}$: Résidué de l'application L_a .

$R_a^{\#}$: Résidué de l'application R_a.

$\preccurlyeq$: L'ordre dans $\mathbb{R}_{max}$ coïncide avec l'ordre usuel $\leq$.

$\succcurlyeq$: L'ordre dans $\mathbb{R}_{min}$ coïncide avec l'ordre usuel $\geq$.

$\not\backslash$: Soustraction à gauche dans un dioïde.

$\not/$: Soustraction à droite dans un dioïde.

$Supp$: Support d'une série formelle.

P : Ensemble des places $P = \{p_1, p_2, p_3 \dots\}$.

T : Ensemble des transitions $T = \{t_1, t_2, t_3 \dots\}$.

P_{ij} : La place qui relie la transition t_j $\grave{a}$ t_i.

τ_{ij} : La temporisation de la place p_{ij}.

m_{ij} : Le marquage de la place p_{ij}.

λ : La valeur propre.

μ : Le vecteur propre.

$M_{in}^{ax}[\![\gamma, \delta]\!]$: Dioïde des séries formelle en γ et δ.

$x_i(k)$: Date du $k^{ième}$tir de la transition x_i.

$x_i(t)$: Le nombre de tir de la transition x_i à l'instant t.

RdP : Réseau de Petri.

AAR : Architecture d'automatisation en réseau.

Sci : Semi-continue inférieurement.

Scs : Semi-continue supérieurement.

SED : Systèmes à événements discrets.

Dr : Temps de réponse.

Introduction générale

Introduction générale

Les systèmes dynamique à événements discrets (SED) désignent des systèmes généralement de conception humaine. Leur évolution obéit à l'apparition d'événements qui ont lieu à des instants discrets. Les systèmes de production (ateliers flexibles, lignes d'assemblages [32], les réseaux de transport (routier, ferroviaire ou aérien [33]) et les systèmes informatiques sont des processus que l'on peut considérer comme des systèmes à événements discrets.

En raison de la nature des phénomènes qui entrent en jeu, à savoir des phénomènes de synchronisation où d'exclusion mutuelles ; les systèmes à événements discrets ne peuvent généralement pas être décrits par des équations différentielles. Ces systèmes sont alors souvent représentés par des modèles états-transitions. Les plus utilisés sont les automates à états finis, les chaines de Markov, et les réseaux de Petri pour les systèmes les plus complexes qui comportent à la fois des événements de synchronisation et de concurrence. Dans la plupart des cas les applications dans ce domaine implique l'existence de contraintes temporelles (intervalle de temps, durée de validité,…etc.) dont il faut tenir compte.

Ce problème de contraintes est très fréquent dans les systèmes embarqués, les réseaux de transport, les systèmes de production (traitements thermiques, chimiques,…etc.), et aussi dans les systèmes de commande en réseau (SCR). Dans ce travail on va s'intéresser à cette dernière catégorie (les SCR) qui sont à temps critique.

De nos jours, la plupart des systèmes industriels repose sur des architectures distribuées en réseau ou ce qu'on appelle aussi des architectures d'automatisation en réseau (AAR). Dans les milieux industriels notamment dans l'industrie manufacturière, dans l'aéronautique ou dans les systèmes de production. Ces architectures peuvent se composer de plusieurs sous-systèmes autonomes coopérant entre eux via des réseaux de terrain. Ces systèmes distribués en réseau peuvent aussi être interconnectés avec d'autres en utilisant plus largement le réseau Internet pour permettre des applications de contrôle, de supervision ou d'e-maintenance sur des longues distances. Par comparaison avec les systèmes basés sur des architectures centralisées, la distribution a permis d'améliorer les applications de contrôle/commande industrielles en termes de réduction des coûts de câblage, de modularité, de flexibilité et aussi en termes d'aide au diagnostic et à la maintenance des systèmes. Cependant dans le système de commande en réseau où le facteur temps est une composante très importante il ne suffit pas de délivrer des résultats exacts mais aussi à satisfaire les contraintes de temps

(notamment le temps de réponse) et les délivrer dans des délais imposés en respectant un certain temps de réponse.

La classe des SED que nous étudions ici à savoir les AAR qui sont soumises à des contraintes temporelles (borne maximale) sur son temps de réponse à ne pas dépasser, et celle qui met en jeu des phénomènes de synchronisation et de concurrence, il est donc nécessaire de disposer de techniques et d'outils pour satisfaire ces contraintes. Pour de tel processus, on obtient des modèles mathématiques sous forme non-linéaire, en les traduisant dans une structure algébrique particulière, appelée dioïde, cette représentation devient linéaire. La représentation est dite alors max-plus linéaire. l'algèbre des dioïdes est donc apparue comme la structure mathématique adéquate pour modéliser et étudier cette classe de systèmes [1].

Par analogie avec la théorie conventionnelle de l'automatique, de nombreux travaux ont été développés sur la commande des systèmes max-plus linéaire. Et des lois de commande ont été proposées.

Le travail qu'on va présenter dans ce mémoire a donc pour objectif d'introduire les résultats proposés dans [14] pour résoudre le problème de commande d'une AAR sous contraintes temporelles.

Ce mémoire est structuré en quatre chapitres comme suit :

Dans le premier chapitre on va présenter les outils algébriques nécessaire à l'étude des SED, et rappeler quelques notions de base sur les dioïdes et treillis, ensuite on va intéresser aux techniques propres aux dioïdes permettant de résoudre des équations.

Dans le deuxième chapitre, on s'intéresse aux systèmes à événements discrets (SED). On présente la modélisation par réseau de Petri (RdP) et plus précisément par une sous classe des ces réseaux qui sont les graphes d'événements temporisés (GET), puis nous montrons comment décrire le comportement d'un GET par l'algèbre (max-plus). A l'image de la transformée en z des systèmes échantillonnés, les systèmes (max, +)- linéaires sont dotés de la transformée en γ, δ qui permet de représenter les systèmes dans le dioïde $M_{in}^{ax}[\gamma, \delta]$.

Dans le troisième chapitre, on va effectuer une étude sur les Architectures d'Automatisation en Réseau. On va présenter ses composants et le principe de fonctionnement, puis on passera à sa modélisation par GET, et traduire leurs comportement à l'aide de l'algèbre de (max, +) et $M_{in}^{ax}[\gamma, \delta]$.

Dans le quatrième chapitre, on va présenter la méthode de L.Houssin[14], qu'on va appliquer sur notre système pour calculer un contrôleur qui satisfait le temps de réponse, on termine par une conclusion générale.

Chapitre 1

Introduction

Dans ce premier chapitre, nous allons présentons les outils algébriques nécessaires à l'étude des SED que l'on utilisera par la suite.

Nous allons nous intéresser tout d'abord aux notions de base des structures ordonnées et treillis. Ensuite les dioïdes et leurs relations avec les treillis. Enfin, les techniques propres aux dioïdes permettant de résoudre des équations.

1.1 Structures ordonnées et treillis

L'association d'une relation d'ordre avec un ensemble (ou sous ensemble) définit un ensemble ordonné qui à son tour introduit un treillis. Nous rappelons brièvement dans cette partie un ensemble de notions, de définitions et de propriétés sur les ensembles ordonnés et les treillis.

1.1.1 Relation d'ordre et structures ordonnées

Sur les entiers, on peut définir les deux relations suivantes :

$a \leq b$: Relation d'ordre habituelle

a/b : a devise b

Ces deux relations présentent des propriétés communes :

Réflexivité $a \leq a$ et a/a

Antisymétrie $\begin{cases} si\ a \leq b \text{ et } b \leq a\ alors\ a = b \\ si\ \ \frac{a}{b}\ \ et\ \ \frac{b}{a}\ \ alors\ \ a = b \end{cases}$

Transitivité $\begin{cases} \text{si } a \leq b \text{ et } b \leq c\ alors\ a \leq c \\ si\ \ \frac{a}{b}\ \ et\ \ \frac{b}{c}\ \ alors\ \ \frac{a}{c} \end{cases}$

On dit alors que les relations $\leq$ et $/$ sont des relations d'ordre.

Définition 1.1 (ensemble ordonné)

Un ensemble muni d'une relation d'ordre est dit ordonné.

L'ensemble $\boldsymbol{E}$ muni d'une relation d'ordre $\preccurlyeq$ forme l'ensemble ordonné$(\boldsymbol{E}, \preccurlyeq)$. Cet ensemble est dit totalement ordonné $\boldsymbol{si} \preccurlyeq$ est une relation d'ordre totale c'est-à-dire si deux éléments quelconques $\boldsymbol{x\ et\ y\ de\ E}$ sont comparables ($\boldsymbol{x \preccurlyeq y\ ou\ y \succcurlyeq x}$). Dans le cas contraire, l'ordre est partiel et l'ensemble est dit partiellement ordonné. On dira que deux éléments $\boldsymbol{x\ et\ y\ de\ E}$ sont incomparables, noté $\boldsymbol{x \parallel y}$, s'ils vérifient $\boldsymbol{x \not\preccurlyeq y\ et\ y \not\succcurlyeq x}$.

Un ensemble totalement ordonné est aussi appelé une chaîne tandis qu'un ensemble composé d'éléments incomparables entre eux est appelé une antichaîne.

Exemple 1 (ensembles ordonnés)

On présente ici quelques exemples d'ensembles munis d'une relation d'ordre

- L'ensemble $(\mathbb{N}, \preccurlyeq)$ est totalement ordonné.
- La relation d'ordre $|$ définie dans $\mathbb{N}$ est partielle. En considérons cette relation, on a par exemple : $5 \| 7$
- L'ensemble $(\mathbb{N}^{2\times 2}, \preccurlyeq)$ est partiellement ordonné bien que $(\mathbb{N}, \preccurlyeq)$ le soit totalement. On peut aisément le remarquer en considérons les deux éléments suivants :

$$\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \| \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}$$

Définition 1.2

soit E un ensemble ordonné par$\preccurlyeq$. On peut définir certains éléments particuliers :

Majorant : M est un majorant de $A \subset E$ si $\forall a \in A, a \leq M$. on dit alors que A est une partie majorée de E.

Minorant : m est un minorant de $A \subset E$ si $\forall a \in A, m \leq a. A$ est une partie minorée de E.

Borne supérieure : $S = Sup\ A$ $(egalement\ notée (S = \vee A)$ est la borne supérieur de A si S est un majorant de A et si pour tout majorant M de A, $S \leq M$. La borne supérieur est communément appelée " plus petit majorant ".

Borne inferieure : $I = Inf\ A$ $(également\ notée\ I = \wedge A)$ la borne inferieur de A si I est minorant de A et si pour tout minorant m de A, $m \leq I$. La borne inferieur apparait comme " le plus grand minorant ".

Élément maximal : un élément maximal de $A \subset E$ est un élément a de A tel que A ne contienne aucun élément plus grand que a, ce qui s'écrit :

$$\begin{cases} a \in A \\ x \in A\ a \leq x \end{cases} \Rightarrow a = x.$$

Élément minimal : un élément minimal b de $A \subset E$ est définit par :

$$\begin{cases} b \in A \\ x \in A\ x \leq b \end{cases} \Rightarrow b = x$$

Plus grand élément : On appelle plus grand élément d'un ensemble E, un élément noté $T_E \in E$ tel que pour tout élément $x \in E$, on ait $x \leq T_E$.

Plus petit élément : On appelle plus petit élément d'un ensemble E, un élément noté $\perp_E \in E$ tel que pour tout élément $x \in E$, on ait $\perp_E \leq x$.

Remarque 1 :

- Le plus petit élément (resp. le plus grand élément), s'il existe, est nécessairement unique.
- S *et* I peuvent ne pas appartenir à A. Si ces bornes existent, elles sont uniques.
- Même s'il admet des majorants (resp. minorants), un ensemble n'admet pas toujours de borne supérieure (resp. borne inferieure).
- Un ensemble ordonné peut admettre plusieurs éléments maximaux (resp. minimaux).

1.1.2 Demi-treillis et treillis [3] ,[4]

Définition 1.3 (demi-treillis)

Soit $(E, \preccurlyeq)$ un ensemble ordonné non vide.

$(\boldsymbol{i})$ Si $x \vee y$ existe dans E pour tout couple $(x, y) \in E$, alors E est un *demi-treillis supérieur*.

$(\boldsymbol{ii})$ Si $x \wedge y$ existe dans E pour tout couple$(x, y) \in E$, alors E est un *demi-treillis inferieur*.

Définition 1.4 (treillis)

L'ensemble ordonné $(E, \preccurlyeq)$ est un treillis si il est à la fois un demi-treillis supérieur et un demi-treillis inferieur.

Remarque : Pour toute relation d'ordre, notée $\preccurlyeq$, il existe une relation d'ordre inverse, notée $\succcurlyeq$. Par conséquent, si $(E, \preccurlyeq)$ est un demi-treillis supérieur (resp. demi-treillis inferieur) alors $(E, \succcurlyeq)$ est demi-treillis inferieur (resp. demi- treillis supérieur). C'est ce qu'on appelle le principe de dualité.

Définition 1.5 (demi-treillis complet)

Soient $(P, \succcurlyeq))$ un demi-treillis supérieur et $(Q, \preccurlyeq))$ un demi-treillis inférieur.

$(\boldsymbol{i})$ si $\bigvee S$ existe pour tout $S \subseteq P$, alors P est demi-treillis supérieur complet.

$(\boldsymbol{ii})$ si $\bigwedge S$ existe pour tout $R \subseteq Q$, alors Q est demi-treillis inférieur complet.

Définition 1.6 (treillis complet)

L'ensemble ordonné $(E, \preccurlyeq))$ est un treillis complet si c'est à la fois un demi-treillis supérieur complet et un demi-treillis inférieur complet.

Exemple 2. Le treillis $(\mathbb{N}, \vee, \wedge)$ n'est pas complet. $(\mathbb{N}\{+\infty, -\infty\}, \vee, \wedge)$ est un treillis complet.

Remarque : Tout treillis fini est complet.

Théorème 1 : Un demi-treillis supérieur complet est un treillis complet, s'il contient un plus petit élément.

1.2 Théorie des dioïdes [1]

Dans cette partie, nous présentons les fondements concernant les demi-anneaux idempotents ou dioïdes, et nous montrons aussi le lien entre ces derniers et les structures ordonnées. Les dioïdes apparaissent ainsi comme des demi-treillis. De plus, un dioïde complet a la structure d'un treillis complet.

1.2.1 Notions de base sur les dioïdes :

Les concepts généraux et les notations sur les structures algébriques des dioïdes qui seront utilisés dans ce mémoire, sont donnés dans cette section.

Définition 1.7 (monoïde)

Un monoïde $(M,\oplus)$ est un ensemble M muni d'une loi de composition interne notée $\oplus$, associative et qui possede un élément neutre ε tel que :

$$\forall m \in M, m \oplus \varepsilon = \varepsilon \oplus m = m.$$

Le monoïde est commutatif, si la loi $\oplus$ est commutative, C'est-à-dire,

$$\forall a, b \in M, a \oplus b = b \oplus a.$$

Définition 1.8 (demi-anneau, dioïde)

On appelle demi-anneau un ensemble D muni de deux lois internes $\oplus$ et $\otimes$ tel que :

- $(D,\oplus)$ est un monoïde commutatif dont l'élément neutre ε est appelé élément nul.
- $(D,\otimes)$ est un monoïde, son élément neutre est appelé unité et est noté e.
- La multiplication $\otimes$ est distributive à droite et à gauche par rapport à la loi $\oplus$,

$$\forall a, b, c \in M, c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b),$$

$$(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c).$$

- L'élément nul ε est absorbant pour la loi $\otimes$ $(\forall a \in D, a \otimes \varepsilon = \varepsilon \otimes a = \varepsilon)$.

Si la loi additive $\oplus$ est idempotente $(\forall a \in D, a \oplus a = a)$, alors $(D,\oplus,\otimes)$ est appelé un demi-anneau idempotent ou dioïde.

Exemple 3. $(\mathbb{R} \cup \{-\infty, +\infty\}, max, +)$ Est un dioïde commutatif pour lequel $\varepsilon = -\infty$ et $e = 0$. Ce dioïde est noté $\mathbb{R}_{max}$, et traditionnellement appelé "algèbre $(max, +)$". Dans ce dioïde, la loi $\oplus$ correspond à l'application *max* et la loi $\otimes$ est la somme usuelle.

Notons toutefois que : $\varepsilon \otimes (+\infty) = (-\infty) + (+\infty) = \varepsilon = (-\infty)$ dans le dioïde $\mathbb{R}_{max}$.

Exemple 4. $(\mathbb{R} \cup \{-\infty, +\infty\}, min, +)$ est un dioïde commutatif pour lequel $\varepsilon = +\infty$; et $e = 0$. Ce dioïde est noté $\mathbb{R}_{min}$, et traditionnellement appelé "algèbre $(min, +)$". Dans ce dioïde la loi $\oplus$ correspond à l'application *min* et la loi $\otimes$ est la somme usuelle.

On note que : $\varepsilon \otimes (-\infty) = (+\infty) + (-\infty) = \varepsilon = (+\infty)$ dans le dioïde $\mathbb{R}_{min}$.

Remarque: Dans les équations définies sur les dioïdes, le symbole ⊗ du produit est supprimé ou remplacé par un point (a ⊗ b=ab=a.b).

1.2.2 Relation d'ordre dans un dioïde

Dans un dioïde D donné, la propriété d'idempotence de la loi additive ⊕ donne une relation d'ordre, notée ≼, définie par :$\forall(a,b) \in D^2, a \preccurlyeq b \Leftrightarrow a \oplus b = b$. De plus cette relation d'ordre est compatible avec les lois de structure de D, c'est-à-dire :

$$\forall(a,b,c) \in D^3, a \preccurlyeq b \Rightarrow a \oplus c \preccurlyeq b \oplus c,$$

$$a \preccurlyeq b \Rightarrow a \otimes c \preccurlyeq b \otimes c.$$

Remarque : L'ordre ≼ définie dans $\mathbb{R}_{max}$ est total est coïncide avec l'ordre usuel ≤. En revanche, l'ordre total ≼ définie dans $\mathbb{R}_{min}$ est l'inverse de l'ordre usuel.

Exemple 5. La relation ≼, associée à l'application *max* est une relation d'ordre qui correspond à l'ordre usuel ≤, $a \preccurlyeq b \Leftrightarrow b = max(a,b) \Leftrightarrow a \leq b$.

$$(1 \preccurlyeq 3) \Leftrightarrow 3 = \max(1,3) \Leftrightarrow 1 \leq 3.$$

La relation ≼, associée à l'application *min* est une relation d'ordre qui correspond à l'inverse de l'ordre usuel ≥, $a \preccurlyeq b \Leftrightarrow b = min(a,b) \Leftrightarrow a \geq b$.

$$(3 \preccurlyeq 1) \Leftrightarrow 1 = min(1,3) \Leftrightarrow 3 \geq 1.$$

1.3 Dioïdes et treillis

L'idempotence de la somme dans un dioïde induit une structure de demi-treillis supérieur, pour lequel la borne supérieure, notée ∨, correspond à la loi additive ⊕ du dioïde, ($(a \oplus b)$ est le plus petit majorant de $a\ et\ b$).

De même, en considérant la définition algébrique d'un *sup-demi-treillis*, on peut noter que la loi ∨ d'un demi-treillis supérieur est associative, commutative et idempotente, c'est-à-dire que ∨ possède les mêmes axiomes que la loi additive ⊕ d'un dioïde. De plus, on sait qu'un dioïde possède un élément minimum ε (le plus petit que tous les autres éléments du dioïde), ce qui confère au dioïde une structure de treillis.

Définition 1.9 (dioïde complet)

Un dioïde $(D,\oplus,\otimes)$ s'il est fermé pour les sommes infinies et si le produit ⊗ distribue à gauche et à droite des sommes infinies. Autrement dit, pour tout $d \in D$ et tout sous-ensemble $A \subset D$, les propriétés suivantes sans vérifiées:

$$d \otimes (\oplus_{a \in A}\, a) = \oplus_{a \in A}\, (d \otimes a),$$

$$(\bigoplus_{a\in A} a)\otimes d = \bigoplus_{a\in A} (d\otimes a).$$

Puisqu'un dioïde D a une structure de treillis $(D, \preccurlyeq)$, s'il est complet, il admet un plus grand élément. On notera T ce plus grand élément. L'élément T correspond à la somme de tous les éléments de D, $T = \bigoplus_{x\in D} x$.

L'élément T est absorbant pour la loi additive $\forall a, T \oplus a = T$, Rappelons néanmoins que, puisque ε est absorbant pour la loi $\otimes$ on a : $T \otimes \varepsilon = \varepsilon \otimes T = \varepsilon$.

Définition 1.10 (sous-dioïde)

Soit $(D,\oplus,\otimes)$ un dioïde. Le sous-ensemble $C \subseteq D$ est qualifié de sous-dioïde de $(D,\oplus,\otimes)$, noté $(C,\oplus,\otimes)$, si et seulement si $\varepsilon, e \in C$ et le sous-ensemble C est fermé pour les lois $\oplus$ et $\otimes$, c'est-à-dire :$\forall a, b \in C, a \oplus b \in C$ et $a \otimes b \in C$.

Exemple 6. L'ensemble $\overline{\mathbb{N}}_{min} = (\mathbb{N} \cup \{+\infty, -\infty\}, min, +)$ est un sous-dioïde complet du dioïde $\overline{\mathbb{R}}_{min}$. On remarque que le dioïde $\overline{\mathbb{N}}_{min}$ est fermé pour les lois $\oplus$ et $\otimes$, $\varepsilon, e \in \overline{\mathbb{N}}_{min}$ et e le plus grand élément de ce dioïde(l'ordre est en Min-Plus qui est l'inverse de l'ordre usuel).

1.3.1 Applications définies sur des dioïdes

Définition 1.11 (**Isotonie, antitonie)**

Soit f une application définie d'un dioïde $(D,\oplus,\otimes)$ dans un dioïde $(C,\oplus,\otimes)$,

f est dite *isotone* si :

$$\forall\, a, b \in D, a \preccurlyeq b \Rightarrow f(a) \preccurlyeq f(b).$$

f est dite *antitone* si :

$$\forall\, a, b \in D, a \preccurlyeq b \Rightarrow f(a) \succcurlyeq f(b).$$

Remarque : Une application f est dite *monotone* si elle est *isotone* ou *antitone***.** L'isotonie et l'antitonie sont des notions utilisées pour caractériser les applications respectivement croissantes et décroissantes, définies sur des ensembles ordonnés.

Définition 1.12 (Application injective, surjective et bijective)

Soit f une application définie d'un dioïde $(D,\oplus,\otimes)$ dans un dioïde $(C,\oplus,\otimes)$,

- f est injective si $\forall\, a, b \in D, f(a) = f(b) \Leftrightarrow a = b$.
- f est surjective si $\forall\ b \in C, \exists\ a \in D,\ f(a) = b$.
- f est bijective si elle est à la fois injective et surjective.

Définition 1.12 (Homomorphisme)

Une application f définie d'un dioïde D vers un dioïde C est un homomorphisme si :

$$\forall a, b \in D: f(a \oplus b) = f(a) \oplus f(b) \quad et\ f(\varepsilon) = \varepsilon, \qquad \textbf{(1.1)}$$

$$f(a \otimes b) = f(a) \otimes f(b) \quad et\, f(e) = e. \qquad \textbf{(1.2)}$$

Notons que chaque application qui vérifie la propriété (1.1) sera appelée $\oplus$-morphisme. Aussi, une application qui ne vérifie que la propriété (1.2) sera appelée $\otimes$-morphisme. Un homomorphisme est donc $\oplus$-morphisme et $\otimes$-morphisme.

Définition 1.13 (Isomorphisme)

Une application f est dit isomorphisme si est seulement si : l'application f est un **homomorphisme** et f est **bijective**

Définition 1.14 (Continuité)

Soient $(D,\oplus,\otimes)$ et $(C,\oplus,\otimes)$ deux dioïdes complets. Une application f de D dans C est dite semi-continue inférieurement (s.c.i), si, pour tout sous-ensemble B $\subset$ D,

$$f(\mathsf{V}_{x\in B}\, x) = \mathsf{V}_{x\in B}\, f(x)$$

Et semi-continue supérieurement (s.c.s.), si :

$$f(\Lambda_{x\in B}\, x) = \Lambda_{x\in B} f(x)$$

Remarque : Une application **s.c.s** ou **s.c.i** est nécessairement isotone puisque

$$a \geqslant b \Leftrightarrow \begin{matrix} a = a \oplus b \Rightarrow f(a) = f(a \oplus b) = f(a) \oplus f(b) \Leftrightarrow f(a) \geqslant f(b) \\ b = a \wedge b \Rightarrow f(b) = f(a \wedge b) = f(a) \wedge f(b) \Leftrightarrow f(b) \leqslant f(a) \end{matrix}$$

Définition 1.15 (Linéarité)

Une application f d'un dioïde $(D,\oplus,\otimes)$ dans un dioïde $(C,\oplus,\otimes)$ est dite linéaire si elle satisfait les propriétés d'additivité et d'homogénéité :

$$\forall a,b \in D, \forall \alpha \in D, f(a \oplus b) = f(a) \oplus f(b) \qquad \text{(Additivité)}$$

$$f(\alpha\, a) = \alpha\, f(a) \qquad \text{(Homogénéité)}$$

La combinaison des deux conditions mentionnées est connue sous le nom de principe de superposition, soit :

$$\forall\, a,b \in D, \forall \alpha\, \beta \in D, \qquad f(\alpha a \oplus \beta b) = \alpha f(a) \oplus \beta f(b).$$

Remarque : En toute rigueur, on devrait plutôt parler de $(\oplus,\otimes)$-linéarité du fait de la structure algébrique particulière d'un dioïde. Dans l'algèbre $(max,+)$ on parlera de $(max,+)$-linéarité.

Exemple 7. La multiplication par $A \in D^{n\times n}$, définie par :

$$L_A \;:\; D^{n\times n} \to D^{n\times n}$$

$$x \to A \otimes x$$

est une application linéaire.

1.3.2 Dioïdes matriciels [6]

Soit $(D, \oplus, \otimes)$ un dioïde, soit $A \in D^{m\times p}, B \in D^{p\times n}\ et\ C \in D^{m\times n}$ des matrices à coefficients dans D. La somme et le produit des matrices sont définis de la façon suivante :

$$\text{La somme} \qquad A \oplus B : (A \oplus B)_{ij} = A_{ij} \oplus B_{ij}\ ,$$

$$\text{Le produit}\ \ A \otimes B : (A \otimes B)_{ij} = \bigoplus_{k=1}^{p} A_{ij} \otimes B_{ij}\ .$$

On peut montrer que l'ensemble D muni de ces deux opérations est un dioïde matriciel, dont l'élément nul noté ε, est la matrice composée exclusivement de ε. L'élément unité est la matrice notée Id_n composée de e sur la diagonale et de ε partout ailleurs

Exemple 8. Nous donnons ici un exemple d'un produit et une somme de deux matrices carrées dans l'algèbre $(max, +)$

$$A = \begin{pmatrix} 2 & 4 \\ 8 & 3 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 3 \\ 6 & 9 \end{pmatrix}$$

$$A \oplus B = \begin{pmatrix} \max[2,1] & \max[4,3] \\ \max[8,6] & \max[3,9] \end{pmatrix} = \begin{pmatrix} 2 & 4 \\ 8 & 9 \end{pmatrix}$$

$$A \otimes B = \begin{pmatrix} \max[2+1,4+6] & \max[2+3,4+9] \\ \max[8+1,3+6] & \max[8+3,3+9] \end{pmatrix} = \begin{pmatrix} 10 & 13 \\ 9 & 12 \end{pmatrix}$$

On peut définir la multiplication d'une matrice A par une constante α :

$$(\alpha \otimes A)_{ij} = \alpha \otimes A_{ij} = \alpha + A_{ij}$$

En général, la multiplication des matrices dans $D^{n\times n}$ n'est pas commutative, même si $(D, \oplus, \otimes)$ est commutative. $D^{n\times n}$ est distributive si D l'est. $A \succcurlyeq B$ dans $D^{n\times n} \Leftrightarrow \{A_{ij} \succcurlyeq B_{ij}\ dans\ D, i = 1, \dots n, j = 1, \dots n\}$.

Exemple 9. Exemple d'un produit et une somme de deux matrices carrées dans l'algèbre $(min, +)$

$$A = \begin{pmatrix} 2 & 4 \\ 8 & 3 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 3 \\ 6 & 9 \end{pmatrix}$$

$$A \oplus B = \begin{pmatrix} \min[2,1] & \min[4,3] \\ \min[8,6] & \min[3,9] \end{pmatrix} = \begin{pmatrix} 1 & 3 \\ 6 & 3 \end{pmatrix}$$

$$A \otimes B = \begin{pmatrix} \min[2+1,4+6] & \min[2+3,4+9] \\ \min[8+1,3+6] & \min[8+3,3+9] \end{pmatrix} = \begin{pmatrix} 3 & 5 \\ 9 & 11 \end{pmatrix}$$

1.3.2.1 Propriétés spectrales des matrices définies dans un dioïde

Les matrices carrées à coefficients dans un dioïde présentent certaines propriétés spectrales intéressantes qui permettent notamment d'étudier le comportement asymptotique de certains systèmes dynamique. Plus particulièrement, pour chaque matrices carrés, on peut lui associer un

graphe valué orienter, appelé graphe de précédence. Si un graphe de précédence est fortement connexe, alors la matrice qui lui correspond est dite irréductible.

Définition 1.16 (graphe valué orienté)

Un graphe orienté est définit par un ensemble de nœud interconnectés par des arcs orientés. Un graphe est dit valué, si des poids positifs sont associés aux arcs reliant les nœuds j et les nœuds i, ces poids correspondent aux termes notés A_{ij}, de la matrice A.

Définition 1.17 (Graphe de précédence)

On appelle graphe de précédence d'une matrice $A \in D^{n\times n}$ noté $G(A)$, le graphe qui est composé de n nœuds et des arcs, notés (j,i), qui sont pondérés par le coefficient A_{ij}. Si $A_{ij} \neq \varepsilon$, alors il existe un arc qui relie le nœud j au nœud i, sinon l'arc (j,i), n'existe pas. Dualement, pour tous graphes orientés valués composés de n nœud, on peut associer une matrice carrée de dimension $n \times n$, telle que les coefficients de cette matrice correspondent aux poids des arcs du graphe.

Définition 1.18 (Chemin)

Un chemin est une suite de nœuds que l'on peut parcourir séquentiellement en empruntant les arcs de $G(A)$.

Un chemin est élémentaire si aucun nœud n'est rencontré deux fois. La longueur d'un chemin est égal à son nombre d'arcs.

Définition 1.19 (Circuit)

Un circuit est un chemin dont l'origine est confondue avec l'extrémité.

Un circuit est élémentaire si tout nœud n'est l'origine que d'un seul arc appartenant au circuit (on peut dans cette définition remplacer le mot "origine" par le mot "extrémité").

Définition 1.20 (Graphe fortement connexe)

Un graphe est dit fortement connexe si pour deux nœuds i et j quelconques il existe un chemin allant de i à j.

Définition 1.21 (matrice irréductible)

Une matrice $A \in D^{n\times n}$ est dite irréductible si pour toute paire de nœuds (j,i), il existe un entier k tel que $(A^k)_{ij} \neq \varepsilon$. Cette matrice admet une unique valeur propre notée $\lambda \in D$, et donnée par : $\lambda = \bigoplus_{k=1}^{n} (traceA^k)^{1/k}$.

Avec : $traceA^k = \bigoplus_{k=1}^{n} (A^k)_{ii}$ et n désigne l'ordre de la matrice A.

Dans l'algèbre usuelle, l'expression des valeurs propres λ s'écrit comme suit :

$$\lambda = max_{k=1}^{n} \left[\frac{1}{k}(max_{k=1}^{n}(A^k)_{ii})\right].$$

Cette valeur propre correspond au maximum des poids moyens des circuits élémentaires du graphe de précédence $G(A)$.

Le graphe de précédence associé à une matrice réductible n'est pas fortement connexe.il est décomposable, dans ce cas, en plusieurs sous graphe fortement connexe. Cette matrice réductible, peut avoir plusieurs valeurs propres, en la décomposant en blocs irréductibles. La détermination des vecteurs propres d'une matrices A définie sur un dioïde, revient à résoudre l'équation suivante : $Au = \lambda u$.

On appelle u le vecteur propre associé à la valeurs propre λ. Le vecteurs propre associé a cette valeurs propre, ne contient pas de ε. Si une composante du vecteur u est égale à ε, il faut que la matrice A contiennent au moins une ligne avec des ε, ce qui correspondra à un graphe $G(A)$ non fortement connexe.

Exemple 10. Nous considérons le graphe orienté et valué de la figure 1.1 suivante :

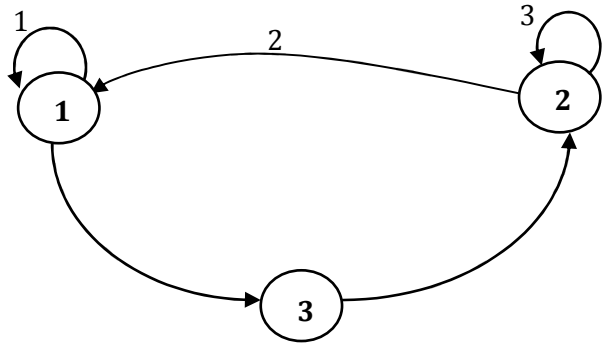


Figure 1.1 : Graphe orienté et valué

$A = \begin{pmatrix} 1 & 2 & \varepsilon \\ \varepsilon & 3 & e \\ e & \varepsilon & e \end{pmatrix}$ est la matrice de précédence associée au graphe de la figure 1.1

Le graphe de la figue 1.1 est composé de trois nœuds et pour chaque paire de nœuds, il existe toujours un chemin orienté qui relie ses nœuds. Le graphe est donc fortement connexe, par conséquent sa matrice de précédence associée A est irréductible. La valeur propre de cette matrice est unique, elle est notée λ et est donnée par : $\lambda = \oplus_{k=1}^{3} (trace A^k)^{1/k}$ dans l'algèbre $(max, +)$ on a :

$$A^2 = \begin{pmatrix} \max(1+1, 2+\varepsilon, \varepsilon+e) & \max(1+2, 2+3, \varepsilon+\varepsilon) & \max(1+\varepsilon, 2+e, \varepsilon+e) \\ \max(\varepsilon+1, 3+\varepsilon, e+e) & \max(\varepsilon+2, 3+3, e+\varepsilon) & \max(\varepsilon+\varepsilon, 3+e, e+e) \\ \max(e+1, \varepsilon+\varepsilon, e+e) & \max(e+2, \varepsilon+3, e+\varepsilon) & \max(e+\varepsilon, \varepsilon+e, e+e) \end{pmatrix}$$

$A^2 = \begin{pmatrix} 2 & 5 & 2 \\ 3 & 6 & 3 \\ 1 & 3 & e \end{pmatrix}, A^3 = \begin{pmatrix} 5 & 8 & 5 \\ 6 & 9 & 6 \\ 3 & 6 & 3 \end{pmatrix}$.

$$\lambda = (\oplus_{i=1}^{3} A_{ii}) \oplus \frac{1}{2}(\oplus_{i=1}^{3} (A^2)_{ii}) \oplus \frac{1}{3}(\oplus_{i=1}^{3} (A^3)_{ii})$$

$$\lambda = (1 \oplus 3 \oplus e) \oplus \frac{1}{2}(2 \oplus 6 \oplus e) \oplus \frac{1}{3}(5 \oplus 9 \oplus 3) = (3 \oplus \frac{6}{2} \oplus \frac{9}{3}) = 3$$

Le vecteur propre, noté u, associé à la valeur propre λ vérifie l'équation suivante : $Au = \lambda u$ le calcul du vecteur propre ce fait comme suit :

$$\begin{pmatrix} 1 & 2 & \varepsilon \\ \varepsilon & 3 & 0 \\ 0 & \varepsilon & 0 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = 3 \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \Longrightarrow \begin{pmatrix} \max(1+u_1, 2+u_2, \varepsilon+u_3) \\ \max(\varepsilon+u_1, 3+u_2, 0+u_3) \\ \max(0+u_1, \varepsilon+u_2, 0+u_3) \end{pmatrix} = \begin{pmatrix} 3+u_1 \\ 3+u_2 \\ 3+u_3 \end{pmatrix}$$

On trouve le vecteur propre suivant : $u = \begin{pmatrix} 3 \\ 4 \\ 0 \end{pmatrix}$.

1.3.3 Résolution d'équation dans un dioïde [5]

Dans ce travail, nous serons amenés à résoudre principalement des équations de la forme : $f(x) = b$, qui sont définit sur un dioïde.

Les lois $\oplus$ et $\otimes$ n'étant pas inversibles, en particulier les applications matricielles. Il n'est donc pas possible, en général d'inverser les applications définies sous forme analytique dans un dioïde. La théorie de la résiduation permet cependant, sous certaines conditions, de définir des pseudo-inverses pour certaines de ces applications et, par conséquent, permet de déterminer des solutions extrêmes de l'équation précédente. C'est-à-dire la plus petite ou la plus grande solution.

Cette théorie concerne de façon générale les fonctions croissantes est souvent on parlera d'application isotones définies sur des ensembles ordonnés.

Mais il peut ne pas exister de solution à cette équation (quand f est non surjective) et/ou une telle solution peut ne pas être unique (quand f est non injective). C'est pourquoi nous étudierons plus précisément deux types d'inéquations de la forme $f(x) \leq b$ ou $f(x) \geq b$, dont on cherchera respectivement la plus grande et la plus petite solution.

1.3.4 Théorie de la résiduation [2]

Définition 1.22. (application résiduable et sa résiduée)

Une application isotone $f: (D, \oplus, \otimes) \longrightarrow (C, \oplus, \otimes)$ définie sur des dioïdes complets est dite résiduable si $\forall b \in C$, l'équation $f(x) \preccurlyeq b$ admet une plus grande solution dans D. L'application qui associe à b la plus grande solution de $f(x) \preccurlyeq b$ est notée $f^{\#}$ et est appelée application résiduée de f. Ainsi :

$$f^{\#}(b) = \oplus\{x \in D \mid f(x) \preccurlyeq b\}.$$

1.3.4.1 Application résiduable sur les dioïdes complets (résiduation de ax ≼ b et xa ≼ b)

Soient L_a (produit à gauche) et R_a(produit à droite), les applications définies sur un dioïde D complet comme suit : $L_a: x \to a \otimes x$

$$R_a: x \to x \otimes a$$

Le fait que le dioïde D soit complet implique que les applications L_a et R_a sont semi-continues inférieurement (distributivité de produit à gauche et à droit sur les sommes infinies). En outre, comme ε est absorbant pour le produit $\otimes$, alors $L_a(\varepsilon) = a \otimes \varepsilon = \varepsilon$ et

$R_a(\varepsilon) = \varepsilon \otimes a = \varepsilon$. Autrement dit, L_a et R_a sont isotones donc résiduable, et leurs applications résiduées sont notées :

$$L_a^{\sharp} : x \to a \backslash x$$

$$R_a^{\sharp} : x \to x / a$$

et respectivement appelées quotient à gauche et quotient à droite. Ainsi, $L_a^{\sharp}(b) = a \backslash b$ et

$R_a^{\sharp}(b) = b / a$ sont les plus grandes solutions des inégalités $a\,x \leq$ b et $x\,a \leq$ b.

Remarque : Lorsque D est un dioide commutatif, $L_a = R_a$ cela implique que $L_a^{\#} = R_a^{\#}$.

Les applications L_a *et* R_a possèdent les propriétés suivantes, qui sont vraies pour tout dioïde complet $(D, \oplus, \otimes)$, avec $a, b, x \in D$.

Propriétés :

$$a \backslash \top = \top \qquad \top \not/ a = \top,$$

$$a(a \backslash x) \preccurlyeq x \qquad (x \not/ a)a \preccurlyeq x,$$

$$a \backslash (ax) \succcurlyeq x \qquad (xa) \not/ a \succcurlyeq x,$$

$$a(a \backslash (ax)) = ax \qquad ((xa) \not/ a)a = xa,$$

$$a \backslash (a(a \backslash x)) = a \backslash x \qquad ((x \not/ a)a) \not/ a = x \not/ a,$$

$$a \backslash (x \wedge y) = a \backslash x \wedge a \backslash y \qquad (x \wedge y) \not/ a = x \not/ a \wedge y \not/ a,$$

$$a \backslash (x \oplus y) \succcurlyeq (a \backslash x) \oplus (a \backslash y) \qquad (x \oplus y) \not/ a \succcurlyeq (x \not/ a) \oplus (y \not/ a),$$

$$(a \wedge b) \backslash x \succcurlyeq (a \backslash x) \oplus (b \backslash x) \qquad x \not/ (a \wedge b) \succcurlyeq (x \not/ a) \oplus (x \not/ b),$$

$$(a \oplus b) \backslash x = a \backslash x \wedge b \backslash x \qquad x \not/ (a \oplus b) = x \not/ a \wedge x \not/ b,$$

$$(ab) \backslash x = b \backslash (a \backslash x) \qquad x \not/ (ba) = (x \not/ a) \not/ b,$$

$$(a \backslash x)b \preccurlyeq a \backslash (xb) \qquad b(x \not/ a) \preccurlyeq (bx) \not/ a,$$

$$b(a \backslash x) \preccurlyeq (a \not/ b) \backslash x \qquad (x \not/ a)b \preccurlyeq x \not/ (b \backslash a),$$

1.3.4.2 Extension aux dioïdes de matrices :

Concernant le cas matriciel, définissons comme suit les applications produit à gauche $L_A : D^{n\times p} \to D^{m\times p}$ et produit à droite $R_{A'} : D^{p\times n} \to D^{p\times m}$ sur les dioïdes de matrices à coefficients dans le dioïde complet D :

$$L_A : X \to A \otimes X$$

$$R_{A'} : X \to X \otimes A'$$

Dans ces applications, $A \in D^{m\times n}$ et $A \in D^{n\times m}$.

Les plus grandes solutions des inéquations $AX \preccurlyeq B$ et $XA' \preccurlyeq B'$ pour lesquelles $B \in D^{m\times p}$ et $B' \in D^{p\times m}$, sont respectivement la résiduée du produit à gauche notée $L_A^{\sharp}(B) = A \backslash B$ et la résiduée du produit à droite notée $R_{A'}^{\sharp}(B') = B' \not/ A'$. Les valeurs de ces matrices sont alors données comme suit :

$$L_A^{\sharp}(B) = A \backslash B : (A \backslash B)_{ij} = \bigwedge_{k=1}^{m} A_{ki} \backslash B_{kj} \qquad R_{A'}^{\sharp}(B') = B' \not/ A' : (B' \not/ A')_{ij} = \bigwedge_{k=1}^{m} B'_{ik} \not/ A'_{jk}$$

En toute rigueur, seul le produit de matrices carrées constitue une application résiduable. Cependant, il est toujours possible de réécrire les applications L_A et $R_{A'}$ dans le dioïde de matrices $D^{q\times q}$ pour lequel $q = \max(m,n,p)$. Les lignes et les colonnes doivent alors être complétées par l'élément ε, ce qui peut compliquer les écritures.

1.3.5 Equations aux points fixes

Grâce à leur structure de treillis, il est possible d'appliquer aux dioïdes les résultats concernant les points fixes d'applications isotones définies sur des treillis complets. Ainsi, des équations du type $f(x) = x$ peuvent être résolues.

Définition 1.23 (Ensemble des points fixes d'application)

Soit f une application isotone définie sur un dioïde complet D. Les ensembles des points fixes, post-fixes et pré-fixes d'application sont respectivement définis comme suit :

$$\mathcal{F}_f = \{x \in D \mid f(x) = x\}$$
$$\mathcal{P}_f = \{x \in D \mid f(x) \geqslant x\}$$
$$\mathcal{Q}_f = \{x \in D \mid f(x) \leqslant x\}$$

Notons que les ensembles des points post-fixes P_f et pré-fixes Q_f peuvent être interprétés dans $\mathcal{F}_f$ selon les équivalences suivantes :

$$f(x) \geqslant x \Leftrightarrow f(x) \oplus x = x,$$
$$f(x) \leqslant x \Leftrightarrow f(x) \oplus x = x.$$

1.3.6 Résolution d'équation par étoile de Kleene

Les équations de type point fixes formulées dans les dioïde, de la forme $f(x) = x$ ou $f(x) = ax \oplus b$, sont importante, car on cherchera systématiquement à représenter les systèmes étudiés sous des formes récurrentes explicites de types point fixe. Il a notamment été mis en évidence que sous certaines hypothèses de continuité de l'application f, les ensembles de solutions de ces équations sont ordonnés, nous nous intéressons plus particulièrement à leurs solutions nominales.

Nous allons voir la manière de résoudre une équation de type $ax \oplus b$ par l'étoile de Kleene ainsi que les propriétés relatives à cette méthode qui nous intéresse essentiellement dans notre travail par la suite.

1.3.6.1 Notion (étoile de Kleene)

Soit * l'opérateur défini sur un dioïde (D,⊕,⊗) par

$$\forall a \in D, a^* = \bigoplus_{i\in\mathbb{N}} a^i, \text{ avec } a^0 = e$$

Cet opérateur est cohérent avec la relation d'ordre $\leqslant$ par l'implication suivante :

$$\forall a, b \in D, si\ a \leqslant b, \text{ alors } a^* \leqslant b^*.$$

De la même façon que pour un dioïde de scalaires, l'étoile d'une matrice carrée $A \in D^{n\times n}$, notée A^*, est défini par $A^* = \bigoplus_{i\in\mathbb{N}} A^i$ avec par convention $A^0 = I_n$.

L'étoile de Kleene possède certaines propriétés dont les plus courantes ont été regroupées si dessous. Ces propriétés sont valables $\forall a, b \in D$.

$$(a^*) = a^* \quad (1)$$

$$a^* a = a a^* \quad (2)$$

$$a(b\ a)^* = a(a\ b)^* a \quad (3)$$

$$(a\ b^*)^* = e \oplus a(a \oplus b)^* \quad (4)$$

$$(a \oplus b)^* = (a^*\ b)^* a^* = a^*(b a^*)^* = (a^*\ b^*)^* \quad (5)$$

$$(a \oplus b)^* = (a^* \oplus b)^{\,*} = (a \oplus b^*)^{\,*} = (a^* \oplus b^*)^* \quad (6)$$

Lorsque D est commutatif :

$$(a \oplus b)^* = a^* b^*$$

On note " $+$ " l'opérateur défini dans un dioïde complet $(D,\oplus,\otimes)$ par :

$a \in D$, $\quad a^+ = \bigoplus_{n\geq 1} a$

On a : $\quad a^* = e \oplus a^+$ et $a^+ = a \otimes a^*$.

Exemple 11 . Soit dans le dioïde $\mathbb{R}_{max}$, $A_0 = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon \\ 6 & 4 & \varepsilon \end{pmatrix}$

$$A_0^* = \bigoplus_{i\in N} A_0^i = I_3 \oplus A_0 \oplus A_0^2 \oplus A_0^3$$

$$A_0^* = \begin{pmatrix} e & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{pmatrix} \oplus \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon \\ \varepsilon & 4 & \varepsilon \end{pmatrix} \oplus \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ 6 & \varepsilon & \varepsilon \end{pmatrix}$$

$$A_0^* = \begin{pmatrix} e & \varepsilon & \varepsilon \\ 2 & e & \varepsilon \\ 6 & 4 & e \end{pmatrix}$$

Théorème : Dans un dioïde complet, la quantité A^*B est la plus petite solution de l'équation $x = Ax \oplus B$ et de l'inéquation $x \geqslant Ax \oplus B$.

Remarque : Dans le dioïde $\mathbb{R}_{max}$, la quantité (A^*B) est la plus petite solution de l'inéquation $x \geqslant Ax \oplus B$, par contre (A^*B) correspond à la plus petite solution au sens usuel et à plus grande solution de l'inéquation $x \leqslant Ax \oplus B$ au sens inverse de l'ordre usuel dans $\mathbb{R}_{min}$.

Exemple 12. Soit le système

$$x = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & 3 \\ 4 & \varepsilon & \varepsilon \end{pmatrix} x \oplus \begin{pmatrix} 2 \\ \varepsilon \\ 5 \end{pmatrix}$$

Le calcul de A^* donne

$$A^* = \begin{pmatrix} e & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{pmatrix} \oplus \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & 3 \\ 4 & \varepsilon & \varepsilon \end{pmatrix} \oplus \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon \\ 7 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{pmatrix} \oplus \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{pmatrix} = \begin{pmatrix} e & \varepsilon & \varepsilon \\ 7 & e & 3 \\ 4 & \varepsilon & e \end{pmatrix}$$

Cela donne la solution suivante :

$$x = A^*B = \begin{pmatrix} e & \varepsilon & \varepsilon \\ 7 & e & 3 \\ 4 & \varepsilon & e \end{pmatrix} \begin{pmatrix} 2 \\ \varepsilon \\ 5 \end{pmatrix} = \begin{pmatrix} 2 \\ 9 \\ 6 \end{pmatrix}$$

1.4 Dioïde de séries formelle [7]

1.4.1 Transformées en γ et en δ

On introduit ici les transformées en γ et en δ qui sont des opérateurs permettant un décalage respectivement événementiel et temporel, à l'aide desquels on exprime une fonction sous la forme d'une série entière, où $\boldsymbol{e = \gamma^0 \delta^0}$ est l'élément neutre du produit et $\boldsymbol{\varepsilon = (\gamma^{-1})^* (\delta^{-1})^*}$ est l'élément neutre de la somme.

1.4.1.1 Transformée en γ:

L'opérateur γ effectue un décalage en numérotation, on a :

γ *d(k) = d(k-1)*, pour toute fonction dateur d(k) on peut écrire :

$$D(\gamma) = \oplus\, d(k)^k$$

1.4.1.2 Transformée en δ:

L'opérateur δ effectue un décalage temporel, autrement dit :

$\delta c\,(t) = c(t-1)$, pour toute fonction compteur c(k) on peut écrire :

$$C(\delta) = \oplus c(t) \delta\, t$$

1.4.1.3 Model d'état en $\boldsymbol{M_{in}^{ax}[\gamma, \delta]}$

A l'image de la transformée en Z, utilisée pour représenter la trajectoire des systèmes à temps discret, on peut utiliser la transformée en **γ, δ** pour représenter les GET. Cette transformée nous permet de représenter le model d'état dans le dioide des séries formelles $\boldsymbol{M_{in}^{ax}[\gamma, \delta]}$.un exemple est donné dans la page 35 pour mieux comprendre comment obtenir le model d'état dans le dioide $\boldsymbol{M_{in}^{ax}[\gamma, \delta]}$.

Conclusion

Dans ce premier chapitre nous avons présenté les outils mathématiques utilisés dans ce travail. Après un bref rappel sur les ensembles ordonnés et treillis, leurs relation avec la structure algébrique des dioïdes, nous avons donné les applications définit sur les dioïdes. Ensuite, plusieurs types de fonctions définies sur les dioïdes sont considérés, fonctions qui généralement ne sont pas inversibles. Cependant, grâce à la théorie de la résiduation, il est possible de considérer les pseudo-inverses d'application isotone, ainsi que la résolution des équations de type point fixes qui nécessitent la définition de l'application étoile de Kleene.

Chapitre 2

Introduction

Dans ce deuxième chapitre, notre travail se focalisera sur les systèmes à événements discrets (SED).

Tout d'abord on va faire un bref rappel sur les (SED), ensuite on va passer à leurs modélisations. L'une des techniques les plus utilisées pour la modélisation est les réseaux de Petri, ces réseaux nous donnent des représentations exactes des systèmes étudiés, on peut les trouver dans les études de [24] et [25]. Dans notre travail nous allons utiliser une sous-classe des ces réseaux qui est les Graphes d'Evénements Temporisés, cette sous-classe nous permet de mettre en évidence les temps, qui est notre sujet dans le 4ème chapitre. A partir de ces graphes, on peut représenter la dynamique des systèmes avec l'algèbre des dioïdes. Dans notre travail on va utiliser la représentation (max, +) pour obtenir des représentations d'état linéaires.

Enfin, A l'image de la transformée en z des systèmes échantillonnés, les systèmes (max, +)- linéaires sont dotés de la transformée en γ, δ qui permet de représenter les systèmes par leur fonction de transfert.

2. Systèmes à événements discrets : [1]

Un système à évènements discrets est un système dont l'espace d'état est discret. Les transitions d'état sont provoquées par l'occurrence d'événements. L'ensemble des événements est également discret. Un événement est validé si l'ensemble des conditions nécessaires à son occurrence sont vérifiées.

Pour prendre en compte le temps, des temporisations peuvent être associées aux événements, une temporisation est un laps de temps qui doit être écoulé avant qu'un événement validé puisse survenir.

Plusieurs outils mathématiques ont été proposés pour modéliser les systèmes à événements discrets on cite par exemple : le GRAFCET, LADER, et les réseaux de Petri.

2.1 Modélisation des systèmes à événements discrets par réseaux de Petri

Les réseaux de Petri (RdP) sont un formalisme privilégié pour l'étude des propriétés qualitatives d'un système complexe. On peut citer la bornitude, la vivacité et l'accessibilité d'état d'un RdP.

Cependant dans notre travail, nous nous intéressons plutôt aux propriétés quantitatives qui peuvent exister entre les différentes places et/ou transitions d'un RdP. Il s'agit d'établir des relations entre elles, qui pourront aisément être formalisées dans un dioïde approprié. Nous traitons plus particulièrement une sous-classe de réseaux de Petri : les graphes d'événements temporisés.

2.2 Réseau de Petri [24]

Les Réseaux de Petri (RdP) constituent un formalisme graphique propre à la modélisation des Systèmes à Evénements Discrets (SED) introduit en 1962 par Carl Adam Petri [Petri 1962]. Ils sont particulièrement adaptés à l'étude des processus complexes mettant en jeu des propriétés de synchronisme et de partage de ressources. Leur support mathématique a permis en outre d'établir de nombreux résultats analytiques. Pour l'étude des SED dans l'algèbre des dioïdes, les RdP sont souvent utilisés comme un outil de modélisation intermédiaire.

2.2.1 Rappels et notions de base des réseaux de Petri :

Un réseau de Petri est un graphe biparti fait de deux types de sommets : places (représentées par des cercles) et transitions (représentées par des barres). Des arcs orientés relient certaines places à certaines transitions, ou certaines transitions à certaines places. Une place peut contenir un nombre entier de marques ou jeton. A chaque arc on associe un poids (entier positif). Un arc sur lequel il ne figure aucun poids veut dire que son poids est égal à 1.

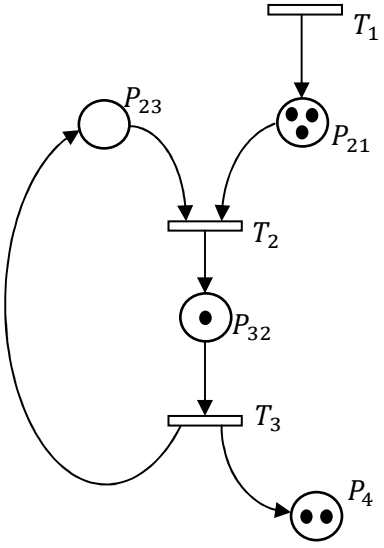


Figure 2.1 : Exemple de réseau de Petri

Définition 2.1 (marquage) [24]

Chaque place contient un nombre entier positif ou nul de marquage (jeton). Le marquage M définit l'état du système décrit par le réseau a un instant donné. C'est un vecteur colonne de dimension m (le nombre de places dans le réseau). La j^{ème} composante de ce vecteur représente le nombre de marque dans la j^{ème} place du réseau.

On note : M_0 marquage initial

Définition 2.2 (Réseaux de Petri P-temporisés) [25]

Pour chaque couple de transition T_i, T on note P_{ji} la place qui relie la transition t_i à t_j. Si cette place P_{ji} existe, la temporisation (en valeur rationnelle positive) correspondante est notée τ_{ji} et son marquage noté m_{ji} .

La plus grande temporisation du graphe d'évènement considéré est notée τ^{max}, définie par : $\tau^{max} = max\{\tau_{ij}\}_{ij/p_{ij}\in P}$.

Définition 2.3 (Réseaux de Petri T-temporisés) [25]

Pour chaque couple de place P_i, P_j, on note T_{ji} la transition qui relie la place P_i à P_j. Si cette transition T_{ji} existe, la temporisation (en valeur rationnelle positive) correspondante est notée τ_{ji} .

La plus grande temporisation du graphe d'évènement considéré est notée τ^{max}, définie par : $\tau^{max} = max\{\tau_{ij}\}_{ij/t_{ij}\in T}$.

2.2.2 Evolution d'un RdP:

L'évolution d'un RdP correspond à l'évolution de son marquage au cours du temps (évolution de l'état du système) : il se traduit par un déplacement de marques ce qui s'interprète comme la consommation/production de ressources déclenchée par des événements ou des actions. Déterminer l'évolution d'un RdP correspond en fait à le simuler, terme plus généralement utilisé en modélisation.

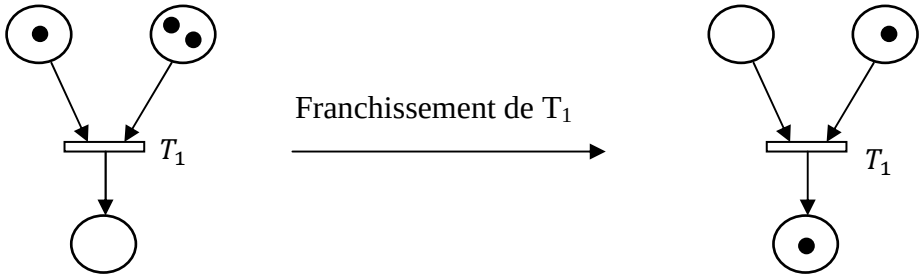


Figure 2.2 : Franchissement de la transition T_1

2.2.3 Quelques propriétés des RdP [20]

- **Accessibilité :** Un marquage M_f est accessible à partir du marquage initial M_0 s'il existe une séquence de franchissement S telle que : du franchissement de cette séquence à partir de M_0, résulte un nouveau marquage M_f et on note : $M_0[S > M_f]$

Cette propriété permet de savoir si un état non désiré risque de se produire.

- **Bornitude :** Une place P_i est dite *bornée* pour un marquage initial M_0 si pour tout marquage accessible à partir de M_0, le nombre de marques dans P_i est fini.

Un RdP est dit borné si toutes ses places sont bornées.

- **Vivacité :** un réseau de Petri est dit *vivant* pour un marquage initial M_0 si pour tout marquage M accessible à partir de M_0 et pour toute transition T_i, il existe une séquence de franchissement S qui inclut la transition T_i.

Cette propriété permet de déduire si le système ne comporte pas de blocage.

2.3 Graphes d'événements temporisés (GET)

Définition 2.5 [8]

Un graphe d'événements est un réseau de Petri où chaque place possède exactement une transition d'entée et une transition de sortie. Un graphe d'événements est dit temporisé si des temporisations sont associées aux places et/ou transitions.

Dans la suite de ce mémoire, nous considérons des graphes d'événement P-temporisés où les temporisations sont associées aux places.

2.3.1 Principe de fonctionnement :

Lorsqu'une marque arrive dans une place temporisée, on dit qu'elle est indisponible pendant un temps τ_{ij} . Quand le temps est écoulé, la marque devient disponible.

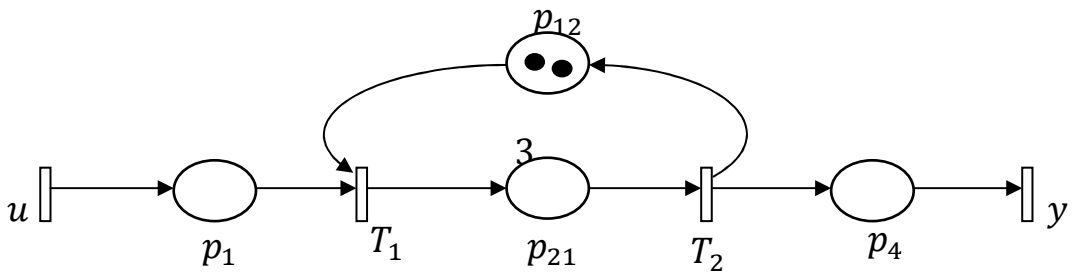


Figure 2.3: Exemple d'un graphe d'événement temporisé d'une cellule de production.

2.3.2 Le temps de cycle d'un graphe d'évènement temporisé

Le temps de cycle d'un système de production correspond à la durée moyenne nécessaire pour produire une seule pièce. Le temps de cycle λ d'un graphe modélisant un processus de production est caractérisé par l'équation suivante : $\lambda=\max \frac{\boldsymbol{T(\rho)}}{\boldsymbol{M(\rho)}}$

où : δ est l'ensemble des circuits élémentaires du GET considéré, $\boldsymbol{T(\gamma)}$ est la somme des temporisations le long des circuits **ρ** et $\boldsymbol{M(\rho)}$ est le nombre de jetons dans les places du circuit **ρ**.

Nous calculons le temps de cycle du graphe d'évènements temporisés en utilisant l'expression donnée précédemment. Nous décomposons le graphe d'évènements de la figure 2.3 en circuit élémentaire. On obtient 1 seul circuit, qu'on note $\boldsymbol{\rho}_1$ pour. Ce circuit est donné comme suit : (d'après la définition 1.19)

$\rho_1=(T_1,P_{21},T_2,P_{12})$

on appliquant la formule de temps de cycle donnée précédemment, le temps de cycle λ du graphe est donnée par :

$$\lambda=\max \frac{\boldsymbol{T(\rho)}}{\boldsymbol{M(\rho)}} = \max \left(\frac{\boldsymbol{3}}{\boldsymbol{2}}\right) = \frac{\boldsymbol{3}}{\boldsymbol{2}}$$

2.4 Modèle linéaire des graphes d'événements temporisés

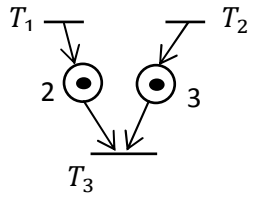
2.4.1 Fonctions compteurs, domaine temporel

La mise en équation de GET peut se faire dans le domaine temporel, où le système est écrit par des fonctions dépendant du temps t. Dans ce cas, on ne s'intéresse plus aux dates d'activation des transitions mais au nombre d'activations de ces dernières jusqu'à un moment donné. En effet, cette représentation consiste à associer à chaque transition un vecteur $\theta(t) \in \mathbb{R}^n_{min}$ que l'on appellera compteur. Le comportement d'un graphe d'évènements temporisé, si on s'intéresse à l'évolution au plus tôt, est représenté par l'équation dans R^n_{min}.

On associe à chaque transition t_i une fonction appelée compteur $X(t) \in \mathbb{R}_{min}^{n}$ qui désigne le nombre de tirs de transition t_i cumulés à l'instant t.

Les compteurs correspondants aux transitions source sont les composantes du vecteur $u(t) \in \mathbb{R}_{min}^{m}$.

Exemple



$$X_3(t) \leq \min\{1 + X_1(t-2), 1 + X_2(t-3)\}$$

Plus généralement, le comportement d'un graphe d'événements temporisé est représenté par l'inéquation suivante :

$$X(t) \leq \oplus_{\tau=0}^{\tau\,max}\left(A_\tau . X(t-\tau) \oplus \mathrm{B}_\tau . u(t-\tau)\right).$$

où $A_\tau \in \mathbb{R}_{min}^{n\times n}$ est la matrice dont le terme $A_{\tau,ij}$ est égal a m_{ij}, qui correspond au nombre de jetons initiaux dans la place p_{ij} si cette place existe et ε sinon. Les termes des matrices $B_\tau(k) \in \mathbb{R}_{min}^{n\times m}$ correspondent aux marquages initiaux des places de sortie des transitions sources. En général, on s'intéresse à l'évolution au plus tôt des graphes d'événements temporisés, c'est-à-dire qu'une transition est franchie dés qu'elle est franchissable. Cette évolution correspond à la solution maximale de l'inéquation précédente dans $\mathbb{R}_{min}$. Cette solution satisfait l'équation linéaire suivante :

$$X(t) = \oplus_{\tau=0}^{\tau\,max}\left(A_\tau . X(t-\tau) \oplus \mathrm{B}_\tau . u(t-\tau)\right). \qquad (2.2)$$

L'équation (2.2) est implicite en général. Elle est souvent remplacée par la solution explicite (forme ARMA).

Définition 2.7 (forme ARMA)[26]

Appelée aussi forme explicite**,** définit l'ensemble des dates au plus tôt pour le fonctionnement du système. Ainsi, à partir d'un vecteur de commande $u(t)$ et des états précédents, l'équation (2.3) permet de déterminer toutes les valeurs du vecteur d'état. ARMA signifie (Auto-Régressif à Moyenne Ajustée (Auto Regressive-Moving Average en anglais).

En fait, le passage à la forme explicite se fait par la résolution de la partie implicite, en sélectionnant la plus grande solution de l'équation implicite. Ceci correspond bien au fonctionnement « au plus tôt ». La forme ARMA est donné comme suit :

$$X(t) = \bigoplus_{\tau=1}^{\tau\,max} (A_0^*.A_\tau.X(t-\tau) \oplus A_0^* B_\tau u(t-\tau)) \qquad (2.3)$$

Où : $A_0{}^*$ est l'étoile de Kleene de la matrice A_0.

Du point de vue de la l'interprétation, la suppression de la partie implicite correspond dans le Graphe d'événements à une transformation au cours de laquelle les places internes (entre deux transitions) sans temporisation sont supprimées.

Equation d'état dans $\mathbb{R}_{min}$

Par analogie avec la théorie des systèmes linéaires classiques, l'équation explicite (2.3) peut être transformée en une forme d'état. Pour obtenir un modèle d'état, nous décomposons toutes les places dont la temporisation $\tau > 1$ en τ places temporisées à 1. Nous ajoutons donc (τ-1) transitions intermédiaires. On associe des compteurs à ces transitions intermédiaires, au nombre de *n'* qui sont les composantes du vecteur $\bar{\theta}(t) \in \mathbb{R}_{min}^{n'}$. Le vecteur d'état résultant, noté *x(t)*, appartient à $\mathbb{R}_{min}^{N}$, avec *N= n +n'*, et est défini par :

$$x(t) = \begin{pmatrix} X(t) \\ \bar{X}\,(t) \end{pmatrix}.$$

Le comportement dynamique du graphe d'événements temporisé étendu est décrit par une équation de la forme :

$$x(t) = \hat{A}_0.x(t) \oplus \hat{A}_1.x(t-1) \oplus \hat{B}.u(t),$$

Qui peut s'écrire sous la forme explicite :

$$x(t) = A.x(t-1) \oplus B.u(t), \qquad (2.4)$$

Avec $A = \hat{A}_0^*.\hat{A}_1$ *et* $B = \hat{A}_0^*.\hat{B}$

Propriété 2. Pour un graphe d'événements temporisé avec des temporisations commensurables, l'équation d'état (2.4) est équivalente à la formulation suivante :

$$x(t) = A^\tau.x(t-\tau) \oplus [\bigoplus_{k=0}^{\tau-1} A^k.B.u(t-k)], \qquad (2.5)$$

Pour tout τ entier tel que $\tau \geq 1$.

Démonstration

Nous allons démontrer cette propriété par récurrence. Etant donné un graphe d'événement temporisé dont le comportement est décrit par l'équation d'état (2.5). Il est clair que la propriété est vérifiée pour $\tau = 1$. Supposons qu'elle est aussi vérifiée pour $\tau = k$, c'est-à-dire :

$$x(t) = A^k.x(t-k) \oplus [\oplus_{k=0}^{k-1} A^k.B.u(t-k)]. \qquad (2.6)$$

Du fait que l'hypothèse que l'équation d'état est satisfaite, nous avons

$$x(t-k) = A.x(t-k-1) \oplus B.u(t-k).$$

En remplaçant $x(t-k)$ par son expression dans (2.4), nous obtenons alors

$$x(t) = A^{K+1}.x(t-(k+1)) \oplus [\oplus_{k=0}^{k-1} A^k.B.u(t-k)].$$

Nous montrons que la propriété est satisfaite pour $\tau = k+1$, donc elle est vrai pour tout $\tau \geq 1$

2.4.2 Fonctions dateurs, domaine événementiel

Pour la représentation en dateurs, on s'intéresse aux dates d'activation des transitions du GET. Dans ce cas, on associe à chaque transition une fonction $X(k) \in \mathbb{R}_{max}^n$ cette fonction est appelée dateur. Les dateurs correspondants aux transitions source sont les composantes du vecteur $u(k) \in \mathbb{R}_{max}^m$.

La dynamique d'un graphe d'événements temporisé est représentée par l'inéquation suivante :

$$X(k) \geq \oplus_{l=0}^{m\ max}\big(A_l.X(k-l) \oplus B_l.u(k-l)\big).$$

où $A_l \in \mathbb{R}_{max}^{n\times n}$ est la matrice dont le terme $A_{l,ij}$ est égal à τ_{ij}, qui correspond à la temporisation de la place p_{ij} marquée à *l*. Si cette place n'existe pas, le terme $A_{l,ij}$ est égal à ε. Les termes des matrices $B_l \in \mathbb{R}_{max}^{n\times m}$ correspondent aux temporisations des places de sortie des transitions source.

En général, on s'intéresse à l'évolution au plus tôt des graphes d'événements temporisés, c'est à dire qu'une transition est franchie dés qu'elle est franchissable. Cette évolution correspond à la solution minimale dans $\mathbb{R}_{max}$ de l'inéquation précédente. Cette solution satisfait l'équation linéaire suivante :

$$X(k) = \oplus_{l=0}^{m\ max}\big(A_l.X(k-l) \oplus \mathrm{B_l}.u(k-l)\big) \qquad (2.7)$$

L'équation (2.7) est implicite en général. Elle est souvent remplacée par sa solution explicite suivante :

$$X(k) = \oplus_{l=1}^{m\ \max}(A_0^*.A_l\theta(m-l)\oplus A_0^*B_l u(m-l)) \qquad (2.8)$$

Où $A_0{}^*$ est l'étoile de Kleene de la matrice A_0.

Equation d'état dans $\mathbb{R}_{max}$.

Pour obtenir un modèle d'état dans $\mathbb{R}_{max}$ pour des GET, nous décomposons toutes les places dont le marquage m > 1 en m places marquées à 1, et donc, nous ajoutons (m-1) transitions intermédiaires. On associe des dateurs à ces transitions intermédiaires au nombre de *n''* qui sont les composantes d'un vecteur $\bar{X}(k) \in \mathbb{R}_{max}^{n''}$. Le vecteur d'état résultant, noté *x(k)*, appartient à $\mathbb{R}_{max}^{N}$, avec *N= n +n''*, et est défini par :

$$x(k) = \begin{pmatrix} X(k) \\ \bar{X}\ (k) \end{pmatrix}.$$

Dans notre cas, nous considérons des graphes d'événements dont le marquage des places qui ont des transitions source en amont est nul. Le comportement dynamique du graphe d'événements temporisé étendu est décrit par une équation de la forme :

$$x(k) = \hat{\mathrm{A}}_0.x(k) \oplus \hat{\mathrm{A}}_1.x(k-1) \oplus \hat{B}.u(\,k),$$

Qui peut s'écrire sous la forme explicite suivante :

$$x(k) = A.x(k-1) \oplus B.u(k), \qquad (2.9)$$

Avec $A = \hat{\mathrm{A}}_0^*.\hat{\mathrm{A}}_1\ etB = \hat{\mathrm{A}}_0^*.\hat{B}$

Exemple :

Nous considérons le graphe d'événement temporisé suivant :

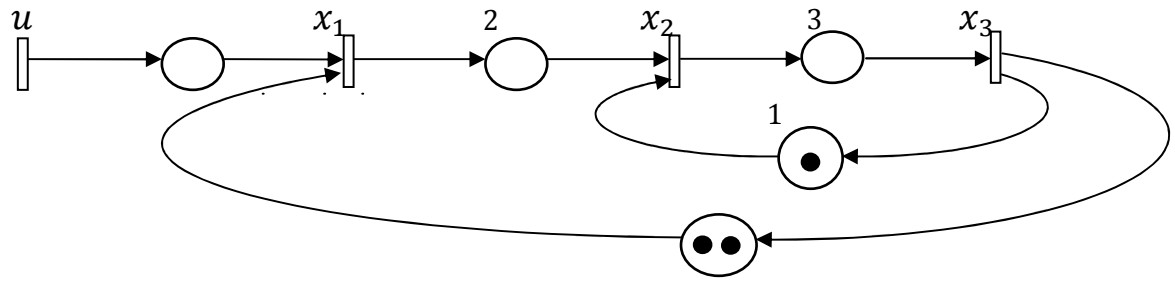


Figure 2.4 Graphe d'événements P-temporisés

Equation compteur :

On considère un franchissement au plus tôt, c'est-à-dire que toute transition validée est immédiatement franchie. Les fonctions compteurs vérifient alors les équations suivantes :

$$\begin{cases} x_1 \preccurlyeq \min\{u(t), x_3(t)+2\} \\ x_2 \preccurlyeq \min\{x_1(t-2), x_3(t-1)+1\} \\ x_3 \preccurlyeq \min\{x_2(t-3)\} \end{cases}$$

Dans l'algèbre $(min, +)$, La solution au plus tôt de ces équations est comme suit :

$$x_1 = u(t) \oplus 2 \otimes x_3(t)$$
$$x_2 = x_1(t-2) \oplus 1 \otimes x_3(t-1)$$
$$x_3 = x_2(t-3)$$

On pose $X(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{pmatrix}$, Matriciellement, on peut écrire

$$X(t) = \begin{bmatrix} \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} X(t) \oplus \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} X(t-1) \oplus \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} X(t-2)$$
$$\oplus \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \end{bmatrix} X(t-3) \oplus \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \end{bmatrix} u(t)$$

Cette solution est implicite, et peut être remplacée par sa solution explicite :

$$X(t) = A_0^* [X(t-1) \oplus A_2 X(t-2) \oplus A_3 X(t-3) \oplus B_0 U(t)]$$

Avec

$$A_0 = \begin{bmatrix} \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} ; \; A_1 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} ; A_2 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix}, A_3 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \end{bmatrix} \; et \; B_0 = \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \end{bmatrix}.$$

Premièrement, on calcule A_0^*, tel que $A_0 = \begin{bmatrix} \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix}$

$$A_0^* = \bigoplus_{i \in \mathbb{N}} A_0^i = I_3 \oplus A_0 \oplus A_0^2 \oplus A_0^3$$

I est une matrice identité tel que :

$$I = \begin{bmatrix} e & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{bmatrix}$$

Calcul de A_0^* :

$$A_{0=}^2 \begin{bmatrix} \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix}$$

Donc $A_0^* = I \oplus A_0$

$$A_0^* = \begin{bmatrix} e & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{bmatrix} \oplus \begin{bmatrix} \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} = \begin{bmatrix} e & \varepsilon & 2 \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{bmatrix}$$

Calcul de la forme explicite :

$$A_0^* A_1 = \begin{bmatrix} e & \varepsilon & 2 \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{bmatrix} \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix}$$

$$A_0^* A_2 = \begin{bmatrix} e & \varepsilon & 2 \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{bmatrix} \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix}$$

$$A_0^* A_3 = \begin{bmatrix} e & \varepsilon & 2 \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{bmatrix} \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \end{bmatrix} = \begin{bmatrix} \varepsilon & 2 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \end{bmatrix}$$

$$A_0^* B_0 = \begin{bmatrix} e & \varepsilon & 2 \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{bmatrix} \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \end{bmatrix} = \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \end{bmatrix}$$

La forme explicite s'écrit sous la forme suivante :

$$X(t) = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} X(t-1) \oplus \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} X(t-2) \oplus \begin{bmatrix} \varepsilon & 2 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \end{bmatrix} X(t-3) \oplus \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \end{bmatrix} U(t)$$

Pour avoir la forme d'état, nous avons étendu le graphe d'événements temporisés initial pour avoir un nouveau graphe équivalent avec des temporisations égales à 1 ou à 0. Nous obtenons alors le graphe de la Figure 2.5

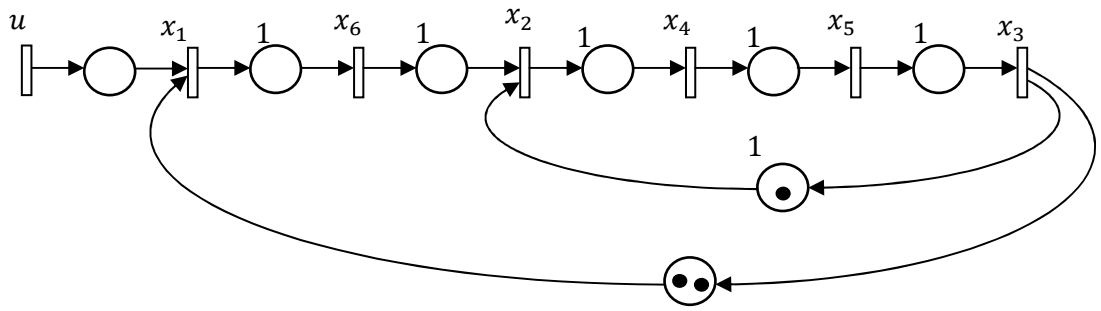


Figure 2.5: Graphe d'événements P-temporisés étendu

On calcul le modèle d'état :

$$\begin{cases} x_1(t) = 2 \otimes x_3(t) \oplus u(t) \\ x_2(t) = 1 \otimes x_3(t-1) \oplus x_6(t-1) \\ x_3(t) = x_5(t-1) \\ x_4(t) = x_2(t-1) \\ x_5(t) = x_4(t-1) \\ x_6(t) = x_1(t-1) \end{cases}$$

$$\begin{cases} x_1(t) = 2 \otimes x_5(t-1) \oplus u(t) \\ x_2(t) = 1 \otimes x_3(t-1) \oplus x_6(t-1) \\ x_3(t) = x_5(t-1) \\ x_4(t) = x_2(t-1) \\ x_5(t) = x_4(t-1) \\ x_6(t) = x_1(t-1) \end{cases}$$

La représentation d'état s'écrit comme suit :

$$X(t) = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & 2 & \varepsilon \\ \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon \\ e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix} X(t-1) \oplus \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \end{bmatrix} u(t)$$

Equation aux dateurs :

Pour le graphe d'événements de la Figure 2.14, on associe une fonction dateur $x_i(k)$ à chaque transition. Pour un franchissement au plus tôt, les fonctions dateurs vérifient les équations suivantes :

$$\begin{cases} x_1(k) \geqslant \max\{(x_3(k-2), u(k)\} \\ x_2(k) \geqslant \max\{2 + x_1(k), 1 + x_3(k-1)\} \\ x_3(k) \geqslant \max\{3 + x_2(k)\} \end{cases}$$

Dans l'algèbre (max,+) , La solution au plus tôt de ces équations est comme suit :

$$\begin{cases} x_1(k) = x_3(k-2) \oplus u(k) \\ x_2(k) = 2 \otimes x_1(k) \oplus 1 \otimes x_3(k-1) \\ x_3(k) = 3 \otimes x_2(k) \end{cases}$$

La forme implicite :

$$X(k) = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon \end{bmatrix} x(k) \oplus \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} x(k-1) \oplus \begin{bmatrix} \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} x(k-2) \oplus \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \end{bmatrix} u(k)$$

Avec

$$A_0 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon \end{bmatrix},\ A_1 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix},\ A_2 = \begin{bmatrix} \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} et\ B_0 = \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \end{bmatrix}$$

Ecriture sous forme explicite :

$$X(k) = A_0^* A_1 x(k-1) \oplus A_0^* A_2 x(k-2) \oplus A_0^* B_0 u(k)$$

$$A_0^* = I \oplus A_0 \oplus A_0^2$$

$$A_0^2 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon \end{bmatrix} \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ 5 & \varepsilon & \varepsilon \end{bmatrix}$$

$$A_0^3 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ 5 & \varepsilon & \varepsilon \end{bmatrix} \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix}$$

$$A_0^* = \begin{bmatrix} e & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & e \end{bmatrix} \oplus \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon \end{bmatrix} \oplus \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ 5 & \varepsilon & \varepsilon \end{bmatrix} \oplus \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} = \begin{bmatrix} e & \varepsilon & \varepsilon \\ 2 & e & \varepsilon \\ 5 & 3 & e \end{bmatrix}$$

$$A_0^* A_1 = \begin{bmatrix} e & \varepsilon & \varepsilon \\ 2 & e & \varepsilon \\ 5 & 3 & e \end{bmatrix} \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & 4 \end{bmatrix}$$

$$A_0^* A_2 = \begin{bmatrix} e & \varepsilon & \varepsilon \\ 2 & e & \varepsilon \\ 5 & 3 & e \end{bmatrix} \begin{bmatrix} \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix} = \begin{bmatrix} \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & 5 \end{bmatrix}$$

$$A_0^* B_{0=} \begin{bmatrix} e & \varepsilon & \varepsilon \\ 2 & e & \varepsilon \\ 5 & 3 & e \end{bmatrix} \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \end{bmatrix} = \begin{bmatrix} e \\ 2 \\ 5 \end{bmatrix}$$

On aura l'équation explicite suivante :

$$X(k) = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 \\ \varepsilon & \varepsilon & 4 \end{bmatrix} x(k-1) \oplus \begin{bmatrix} \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & 5 \end{bmatrix} x(k-2) \oplus \begin{bmatrix} e \\ 2 \\ 5 \end{bmatrix} u(k)$$

Pour avoir la forme d'état, nous avons étendu le graphe d'événements temporisés de la figure 2.14 pour cela nous décomposons chacune des deux places ayons deux jetons en deux places avec un seul jeton, et ajoutons aussi deux transitions intermédiaires. Nous obtenons un GET équivalent dans les marquages sont égaux à 1 ou 0. Nous obtenons alors le graphe de la figure 2.16

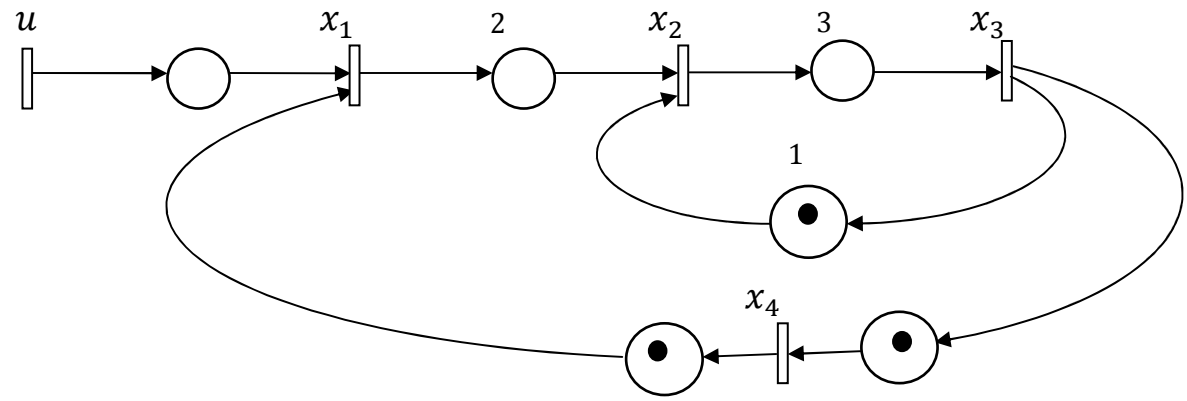


Figure 2.6 : Graphe d'événements t-temporisés étendu.

Le modèle d'état obtenu est donnée par :

On calcule le modèle d'état :

$$\begin{cases} x_1(k) = x_4(k-1) \oplus u(k) \\ x_2(k) = 2 \otimes x_1(k) \oplus 1 \otimes x_3(k-1) \\ x_3(k) = 3 \otimes x_2(k) \\ x_4(k) = x_3(k-1) \end{cases}$$

$$\begin{cases} x_1(k) = x_4(k-1) \oplus u(k) \\ x_2(k) = 2 \otimes x_4(k-1) \oplus 2 \otimes u(k) \oplus 1 \otimes x_3(k-1) \\ x_3(k) = 5 \otimes x_4(k-1) \oplus 5 \otimes u(k) \oplus 4 \otimes x_3(k-1) \\ x_4(k) = x_3(k-1) \end{cases}$$

Le modèle d'état obtenu est donnée par :

$$X(t) = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & 1 & 2 \\ \varepsilon & \varepsilon & 4 & 5 \\ \varepsilon & \varepsilon & e & \varepsilon \end{bmatrix} X(t-1) \oplus \begin{bmatrix} e \\ 2 \\ 5 \\ \varepsilon \end{bmatrix} u(t)$$

Le model d'état dans le dioide $M_{min}^{ax}[\gamma, \delta]$:

Le model d'état dans le dioide $M_{min}^{ax}[\gamma, \delta]$ de figure 2.4 s'écrit comme suit :

$$\begin{cases} x_1 = \gamma^2 x_3 \oplus u \\ x_2 = \delta^2 x_1 \oplus \gamma^1 \delta^1 x_3 \\ x_3(k) = \delta^3 x_2 \end{cases}$$

On écrit matriciellement :

$$A = \begin{bmatrix} \varepsilon & \varepsilon & \gamma^2 \\ \delta^2 & \varepsilon & \gamma^1 \delta^1 \\ \varepsilon & \delta^3 & \varepsilon \end{bmatrix}; \quad B = \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \end{bmatrix}$$

Avec : $x = Ax \oplus Bu$

Conclusion :

Dans ce deuxième chapitre, on a principalement abordé les problèmes liés à la modélisation des systèmes linéaires dans les dioïdes. Nous avons vu la modélisation de systèmes à événements discrets par des réseaux de Petri et plus particulièrement, par une sous-classe des réseaux de Petri, à savoir, les graphes d'événements temporisés avec les représentations aux dateurs et aux compteurs.

Les représentations d'état basées sur des dateurs et des compteurs d'évènements nous ont permis de représenter les systèmes sous une forme explicite qui nous donne la forme d'état.

Enfin nous avons donné un exemple de la transformé en $M_{min}^{ax}[\gamma, \delta]$.

Chapitre 3

Introduction

Dans ce troisième chapitre, nous allons présenter les architectures d'automatisation en réseau, leurs composants et le fonctionnement de ces éléments principaux. Ces AAR peuvent fonctionner selon plusieurs protocoles. Dans notre travail, on a choisit le protocole producteur/consommateur.

Par la suite, nous passerons à la modélisation des AAR fonctionnant avec le protocole choisit en utilisant les graphes d'événements temporisés, puis à leurs représentations en équations (max, +)-linéaire, puis$M_{in}^{ax}[\gamma, \delta]$.

En fin, on va s'intéresser au temps de réponse qui est un critère très important pour le bon fonctionnement des AAR, plusieurs travaux ont été réalisés sur ce sujet, on cite les travaux de [9,12].

3. Les architectures d'automatisation en réseau

3.1 Présentation des AAR [11]

Une Architecture d'Automatisation en Réseau (AAR), ou ce qu'on appelle aussi SCR qui est un système de contrôle-commande distribué. Où les modules d'entrée/sortie (capteurs et actionneurs) communiquent en temps réel via un réseau de communication appelé réseau de terrain avec les contrôleurs logiques programmables (comme les systèmes de commande).

Dans les industries, on peut implémenter une AAR par réseau Ethernet (Ethernet IP, UDP/IP Modbus TCP, Profinet,…).

3.2 Les composants d'une AAR : [17]

Les architectures d'automatisation en réseau (AAR) se composent de contrôleurs logique CL ou calculateurs, de modules d'entrées/sortie (MES) connectés à un réseau de terrain par lequel ces différents composants échangent des données. Le but de ce système est les échanges entre les contrôleurs et la partie opérative.

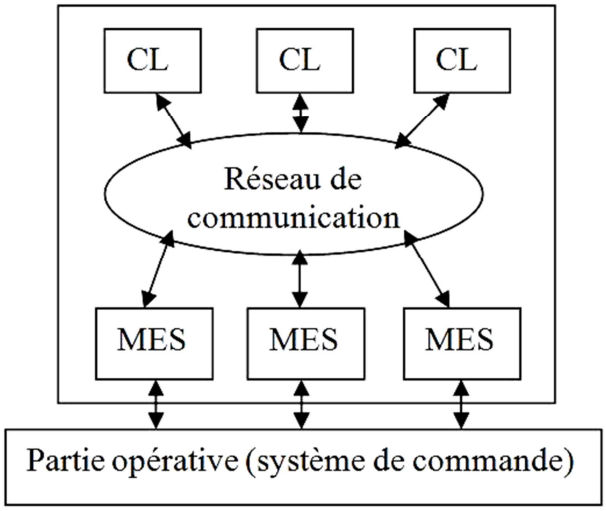


Figure 3.1. Structure générale d'une AAR

3.2.1 Caractéristique du réseau de communication

Les réseaux de terrain sont une solution à l'augmentation de la complexité du câblage dans le niveau inférieur du processus de production. Le but de ces réseaux et de remplacer la liaison point à point par un réseau où toutes les informations sont multiplexées temporellement sur le même support (bus).Ce support fournit un lien direct entre les capteurs, les actionneurs et les équipements de contrôle.

Parmi ces avantages on cite :

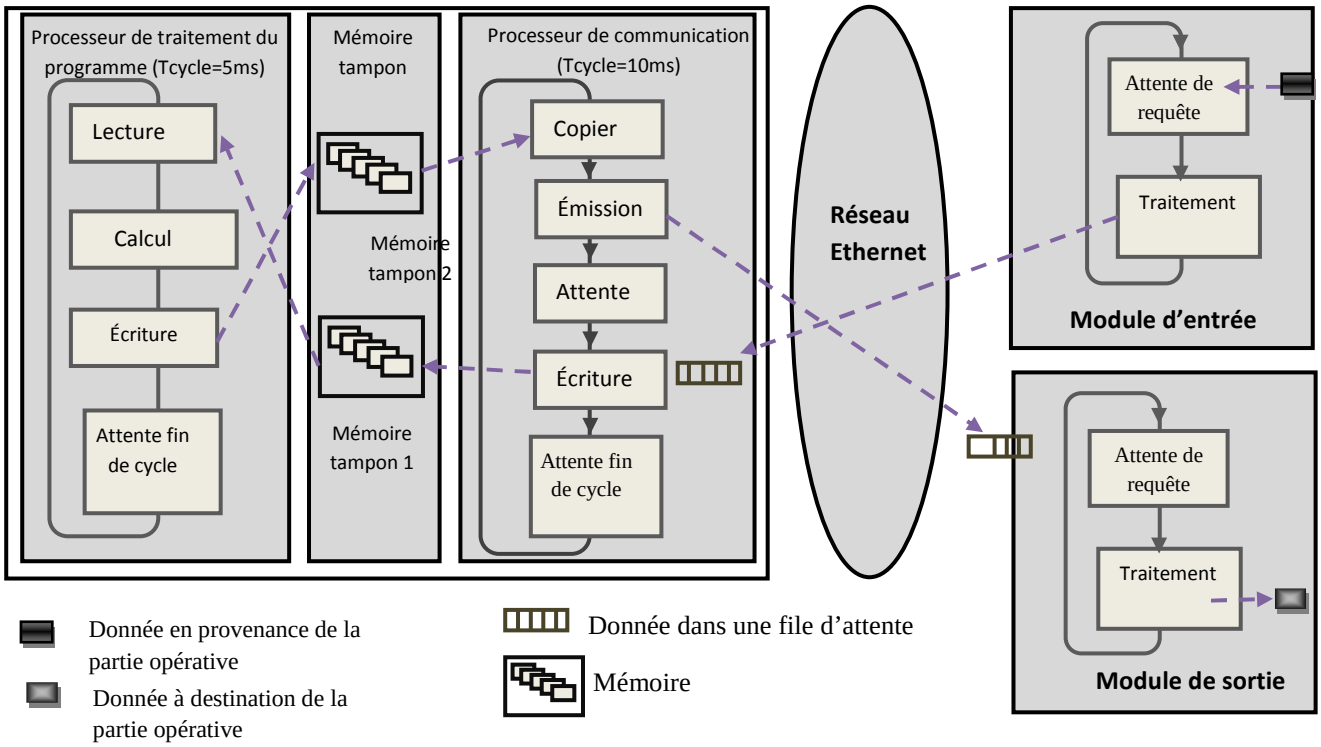
- Faible coût de câblage.
- Simplification de l'installation et réduction de la complicité de manutention.
- Modularité : l'expansion du système est plus facile et rapide. l'addition d'un nouveau capteur au groupe de capteurs se fait simplement avec une connexion sur le réseau, et non sur un nouveau « layout », avec de nouvelle interface dans le contrôleur central.
- Possibilité de diffusion (″broadcasting″ et ″multicasting″) : si la même information physique d'un processus est nécessaire en divers endroits du système (control, opérateur, alarme,…), avec un réseau, on peut la rendre accessible, d'une façon virtuelle, à tous les autres utilisateurs. Un réseau de terrain avec capacité de ″broadcasting″ permet l'acquisition simultané de tous les capteurs mis en relation (mais situé en des endroits différèrent). la diffusion d'information pour tous les consommateurs sur le réseau peut aussi se faire d'une façon simultané.

Dans notre étude, nous considérons que le réseau de communication est un élément de l'AAR, et on va s'intéresser qu'aux types de protocoles d'échange.

Les protocoles d'échanges les plus utilisés dans une architecture d'automatisation en réseau sont les suivants :

- Maître/Esclave : dans ce cas, l'esclave ne peut rien faire, il ne peut pas emmètre d'informations que lorsqu'il reçoit l'ordre du maître. Le maître interroge cycliquement les esclaves et leur donne la main, chacun son tour, pour pouvoir émettre des données.
- Client/serveur : avec ce protocole, le client interroge de manière libre un serveur pour lui demander des informations et le serveur lui répond simplement. Dans ce cas les contrôleurs sont les clients et les modules d'entrée/sortie sont les serveurs.
- Producteur/Consommateur : dans ce cas, le producteur n'attend pas la réception d'un aval ou une requête du consommateur pour émettre l'information, il le fait indépendamment (à chaque fois qu'une variable change d'état) et l'information est envoyée avec précision aux seuls consommateurs intéressés par la variable publiée. Dans notre cas, pour l'AAR, le module d'entrée qui est producteur envoie les requêtes vers la CPU qui est le consommateur sans qu'il reçoive d'information de ce dernier. Dans la phase de retour de l'information vers le module de sortie, la CPU prend le rôle de producteur et le module de sortie devient consommateur.

3.2.2 Fonctionnement de l'AAR (producteur/consommateur) [10]



.2 Comportement d'une AAR

Dans cette section, nous allons voir le principe de fonctionnement de L'AAR en considérant que le protocole est producteur/consommateur.

Les contrôleurs logiques et les modules d'entrée-sortie communiquent pour effectuer la fonction d'automatisation, avec le protocole sélectionné, dans la première phase de communication les modules d'entrés qui sont producteurs communiquent avec le contrôleur qui est consommateur, et dans la deuxième phase de communication le contrôleur devient producteur et communique avec les modules de sorties qui sont consommateurs.

3.2.2.1 Contrôleur logique programmable :

Dans des systèmes pareils, l'automate doit accomplir deux fonctions : la première consiste à exécuter le programme utilisateur et la seconde consiste à transmettre des données aux modules d'entrée/sortie(E/S) via le support de communication. Il se décompose en deux parties :

Processus informatique (CPU) et carte de communication (COM).

a) La CPU :

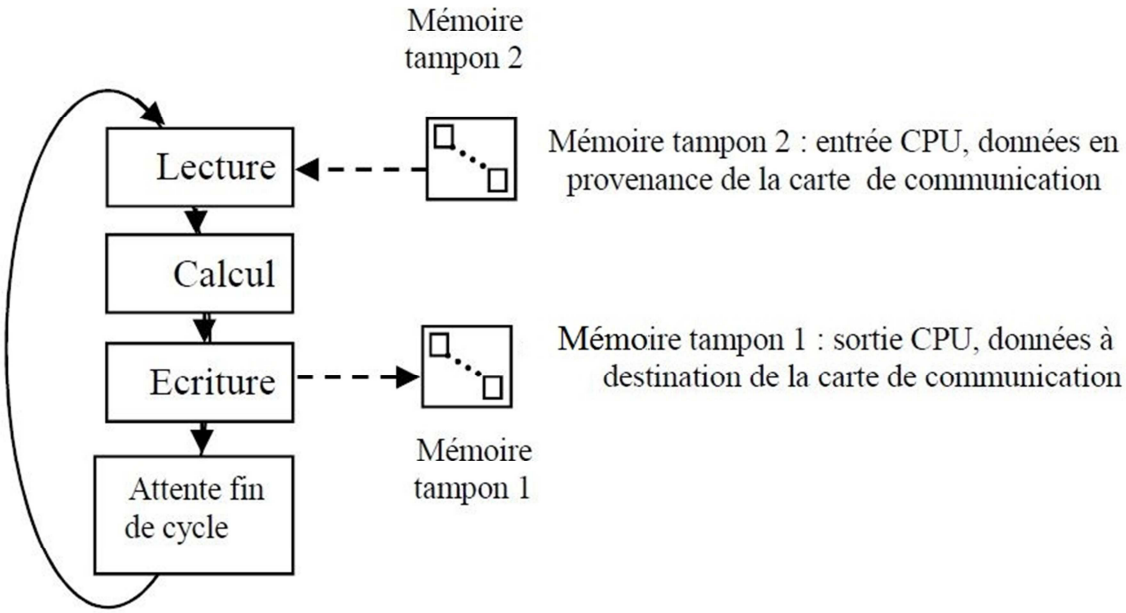


Figure 3.3 Fonctionnement de la CPU

La CPU effectue périodiquement quatre tâches consécutives : elle lit les données d'entrée (durée fixe), exécute le programme utilisateur (durée variable), écrit les données de sortie (durée fixe) et exécute d'autres traitements, indépendamment du programme (durée variable). Car la lecture des donnés d'entrée et l'écriture des donnés de sortie sont beaucoup plus rapide que le calcul du programme ou les autre traitements, ils sont négligés. Le temps de cycle de la CPU est constant.

b) La carte de communication (COM)

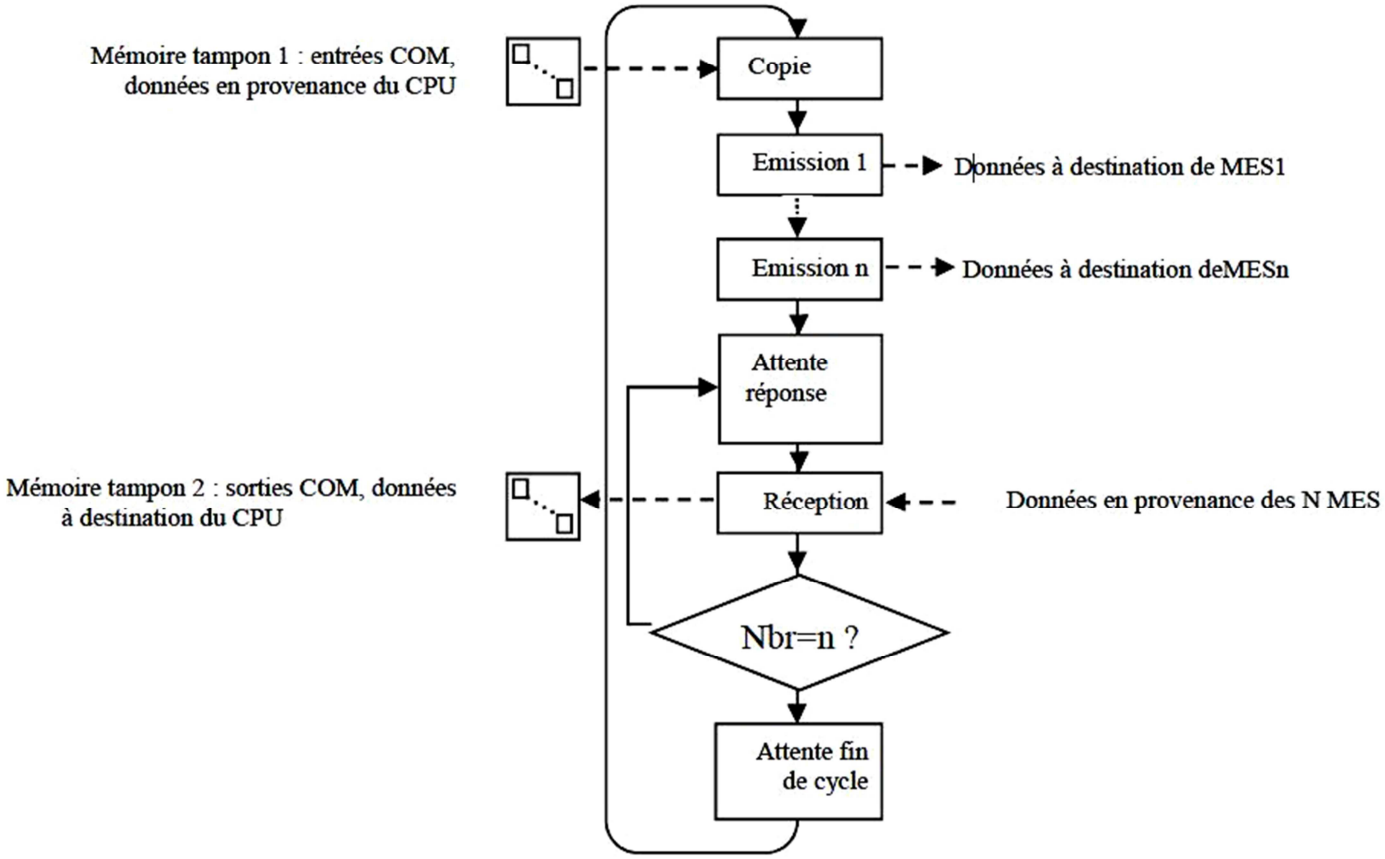


Figure 3.4 Fonctionnement de la COM

La COM a un comportement périodique : en attente d'une demande du module d'entrée (durée variable), envoi de trames à la CPU (durée fixe), lecture des données de sorties (tampon 1) de la CPU (durée fixe), envoi d'un cadre de donnée au module de sortie et en attendant la fin du cycle (durée variable). Le cycle de COM est constant.

3.2.2.2 Réseau (NET)

Il est fait uniquement pour les communications entre l'automate et les modules d'entrée-sortie. Il est toujours prêt à envoyer les requêtes des modules d'entrée à la COM (retard fixé) et de la trame du COM aux modules de sortie (délai fixe).

3.2.2.3 Modules d'entrée/sortie (MES)

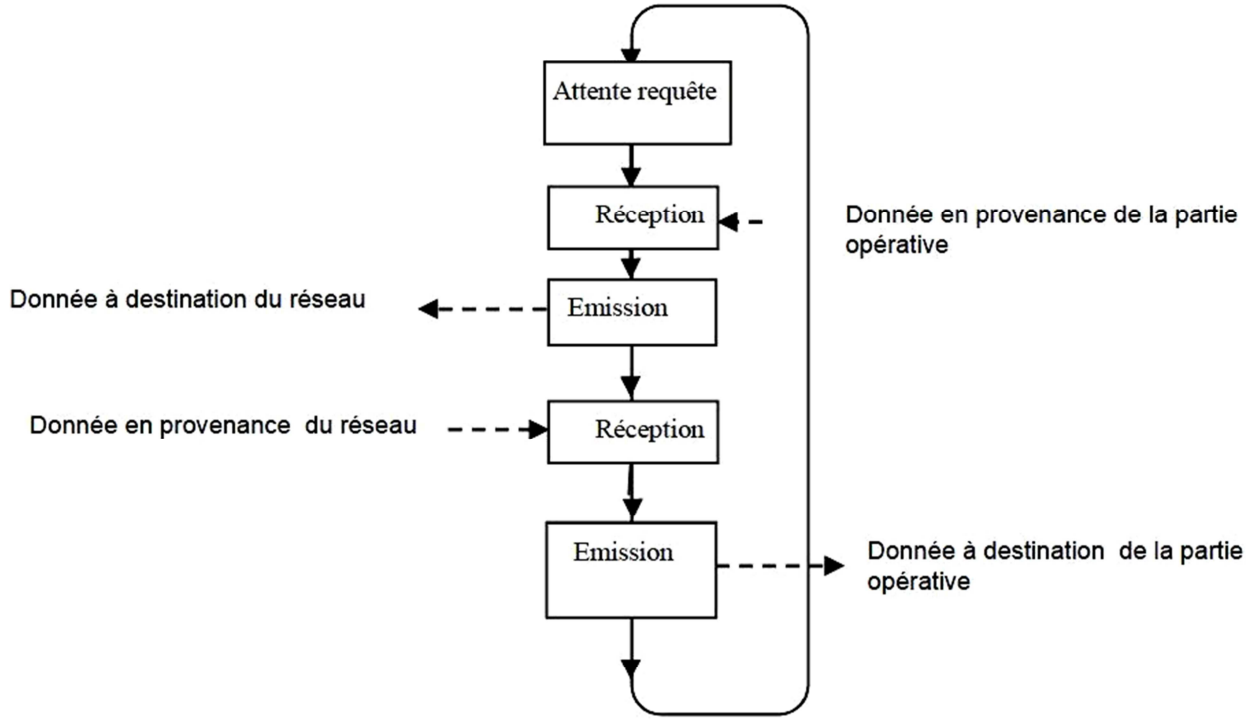


Figure 3.5. Fonctionnement des modules d'E/S

Les modules (E/S) représentent le lien direct entre l'automate et l'installation contrôlée. Le changement d'état des modules d'entrée est le début de l'évolution du modèle de fonctionnement, et le changement d'état de la sortie indique la fin de l'évolution du modèle par rapport au premier changement.

3.3 Modélisation d'une AAR avec GET P-temporisé [10]

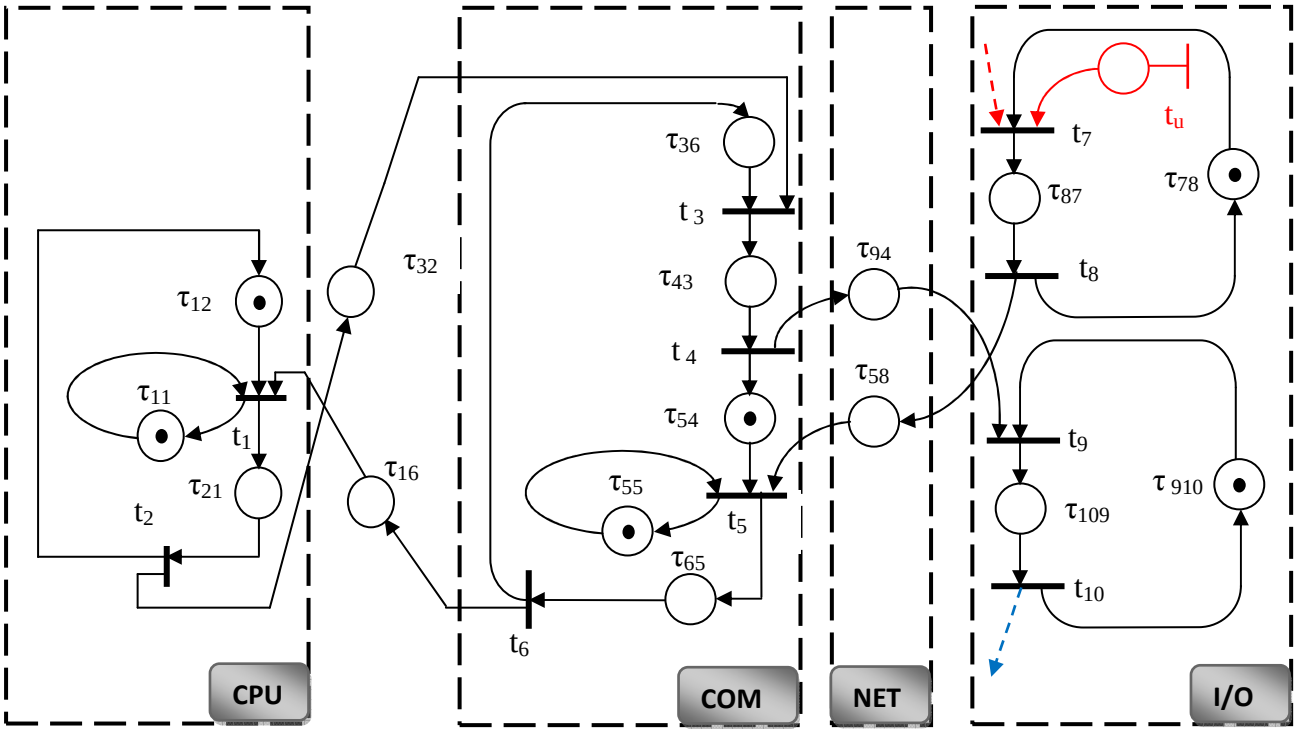


Figure 3.6 Model d'un système de commande en réseau producteur/consommateur avec un seul module d'E/S

En concertant le fonctionnement de l'architecture de contrôle décrite dans la section **3.2.2**, nous avons tiré de [10] le GET (**figure 3.6)** de l'AAR. Ce GET traduit directement le modèle fonctionnel de l'AAR et représente toutes ses composantes. Il a trois parties principales. La partie de la CPU et la partie de la carte de communication sont toutes les deux liées par deux mémoires tampons. La troisième partie représente le module d'E/S. Ce dernier est lié à la carte de communication via un réseau de terrain qui permet d'échanger entre ces deux modules. Les places p12, p21, p11 avec les temporisations τ12, τ21, τ11 sont respectivement la phase d'attente, l'exécution du programme utilisateur (lecture et écriture inclus) et l'indisponibilité de la CPU. Avant le traitement, la CPU lit toutes les nouvelles valeurs d'entrée qui ont été placées dans la mémoire tampon1 (p16) par la carte de communication. Après l'exécution du programme utilisateur, la CPU a mis les résultats dans la mémoire tampon2 (p32). Les tampons sont le lien unique existant entre la CPU et la carte de communication. τ55 représente la période de la carte de communication et un jeton à l'endroit p55 signifie qu'il est occupé pendant au moins cette période. L'envoi d'une requête commence par le tir t8, se termine par le tir t5. Un jeton dans p36 signifie que la requête est

envoyée et que la COM est valable pour la réponse, t43 étant le temps requis pour envoyer une réponse. Les places p58 et p94 modèlent le basculement en raison des retards imposés aux demandes envoyées et aux réponses renvoyées. Les places p87 et p109 modèlent respectivement, le module d'entrée et le module de sortie du traitement.

3.4 Représentation (max,+) du comportement de l'AAR [10]

Nous avons vu dans le chapitre précédent qu'on peut décrire le GET par des équations (max,+)-linéaires. Pour représenter le comportement dynamique du graphe de la figure 6, nous associons à chaque transition $t_{i,}$ avec i=1 à 10, une fonction dateur *x(k),* qui représente la date du $k^{ème}$ franchissement de la transition t_i. La fonction dateur attribuée à la transition d'entrée est notée u(k). On considère un franchissement au plus tôt, c'est-à-dire que toute transition validée est immédiatement franchie. Les fonctions dateurs vérifient alors les équations suivantes :

Avec :

$\tau_{12}=\tau_{36}=\tau_{54}=\tau_{78}=\tau_{910}=\tau_{16}=\tau_{32}=0$, les temporisations associées aux places $P_{12},P_{36},P_{54},P_{78},P_{910},P_{16},P_{32}$

$\tau_{43}=\tau_{87}=\tau_{109}=\tau_{58}=\tau_{94}=\tau_{65}=1$, les temporisations associées aux places $P_{43},P_{87},P_{109},P_{58},P_{94},P_{65}$

$\tau_{11}=5,\tau_{55}=10,\tau_{21}=2$, les temporisations associées aux P_{11},P_{55},P_{21}.

$$\begin{cases} x_1(k) \geqslant 5\otimes x_1\ (k-1)\oplus e\otimes x_2\ (k-1)\oplus e\otimes x_6\ (k) \\ x_2(k) \geqslant 2\otimes x_1\ (k) \\ x_3(k) \geqslant e\otimes x_2\ (k)\oplus e\otimes x_6\ (k) \\ x_4(k) \geqslant 1\otimes x_3\ (k) \\ x_5(k) \geqslant 1\otimes x_8\ (k)\oplus e\otimes x_4\ (k-1)\oplus 10\otimes x_5\ (k-1) \\ x_6(k) \geqslant 1\otimes x_5\ (k) \\ x_7(k) \geqslant e\otimes x_8\ (k-1)\oplus e\otimes u(k) \\ x_8(k) \geqslant 1\otimes x_7\ (k) \\ x_9(k) \geqslant 1\otimes x_4\ (k)\oplus e\otimes x_{10}\ (k-1) \\ x_{10}(k) \geqslant 1\otimes x_9\ (k) \end{cases}$$

L'équation d'état implicite dans $\mathbb{R}_{max}$ est donnée :

$$x(k) = A_0\otimes x(k)\oplus A_1\otimes x(k-1)\oplus B_0\otimes u(k) \qquad (3.1)$$

Où

$$A_0 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon \end{bmatrix}, A_1 = \begin{bmatrix} 5 & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & e & 10 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix},$$

$$B_0 = \begin{bmatrix} \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ e \\ \varepsilon \\ \varepsilon \\ \varepsilon \end{bmatrix}$$

On calcul la forme explicite comme suit :

$$x(k) = A \otimes x(k-1) \oplus B \otimes u(k) \qquad (3.2)$$

$$A = A_0^* \otimes A_1 = \begin{bmatrix} 5 & e & \varepsilon & 1 & 11 & \varepsilon & \varepsilon & 3 & \varepsilon & \varepsilon \\ 7 & 2 & \varepsilon & 3 & 13 & \varepsilon & \varepsilon & 5 & \varepsilon & \varepsilon \\ 7 & 2 & \varepsilon & 3 & 13 & \varepsilon & \varepsilon & 5 & \varepsilon & \varepsilon \\ 8 & 3 & \varepsilon & 4 & 14 & \varepsilon & \varepsilon & 6 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & e & 10 & \varepsilon & \varepsilon & 2 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 1 & 11 & \varepsilon & \varepsilon & 3 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon \\ 9 & 4 & \varepsilon & 5 & 15 & \varepsilon & \varepsilon & 7 & \varepsilon & e \\ 10 & 5 & \varepsilon & 6 & 16 & \varepsilon & \varepsilon & 8 & \varepsilon & 1 \end{bmatrix}, \quad B = A_0^* \otimes B_0 = \begin{bmatrix} 3 \\ 5 \\ 5 \\ 6 \\ 2 \\ 3 \\ 0 \\ 1 \\ 7 \\ 8 \end{bmatrix}$$

3.5 Représentation $M_{in}^{ax}[\gamma, \delta]$ du comportement de l'AAR

Dans $M_{in}^{ax}[\gamma, \delta]$ nous obtenons le model d'état suivant :

$$A = \begin{bmatrix} \gamma^1\delta^5 & \gamma^1 & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \delta^2 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \gamma^1 & \gamma^1\delta^{10} & \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \gamma^1 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \gamma^1 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^1 & \varepsilon \end{bmatrix}; B = \begin{bmatrix} \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ e \\ \varepsilon \\ \varepsilon \\ \varepsilon \end{bmatrix}$$

3.6 Le temps de réponse d'une AAR : [9]

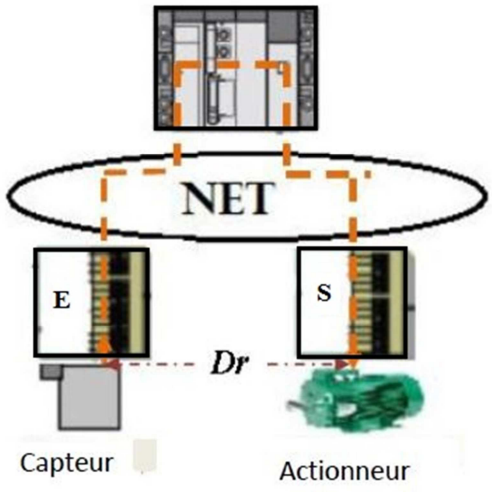


Figure 3.7 Temps de réponse d'une AAR

Dans les systèmes d'automatisation en réseau, le temps de réponse (temps de réactivité) désigné par (Dr) est le temps entre la variation de l'état d'une entée (capteur) et ses conséquences sur la sortie (actionneurs) du système contrôlé, voir figure3.7. Ce temps de réponse n'est pas constant mais présenté par une répartition de valeurs [11,12]. La recherche des limites supérieur et inferieur de cette distribution est un problème crucial pour ces systèmes. Dans la littérature, il existe plusieurs contributions qui évaluent les performances temporelles de l'AAR. On peut citer, par exemple les travaux de [1], [20] où les auteurs

déterminent les limites supérieures du temps de réponses pour différentes catégories d'architectures d'automatisations.

3.6.1 Calcul du temps de réponse de l'AAR

Dans notre cas, on obtient le temps de réponse entre deux bornes $[Dr_{min}, Dr_{max}]$, où Dr_{min} et Dr_{max} sont le temps de réponse minimum et maximum respectivement.

Donc on obtient :

$$Dr_{min}=\tau_{87}+\tau_{58}+\tau_{65}+\tau_{16}+\tau_{21}+\tau_{32}+\tau_{43}+\tau_{94}+\tau_{109}$$

$$=8 \qquad (3.3)$$

$$Dr_{max}=\tau_{87}+\tau_{58}+\tau_{55}+\tau_{65}+\tau_{16}+\tau_{11}+\tau_{21}+\tau_{32}+\tau_{43}+\tau_{94}+\tau_{109}=19 \qquad (3.4)$$

Donc on obtient un temps de cycle compris entre [8,19].

Dans le prochain chapitre on va voir comment satisfaire cette contrainte de temps.

Conclusion

Dans ce troisième chapitre nous avons présenté les architectures d'automatisation en réseau, leurs composantes ainsi que leurs fonctionnements, puis nous avons présenté le model de cette AAR par un graphe d'événement P-temporisé et la représentation de la dynamique de cette dernière dans l'algèbre (max-plus). Vers la fin de ce chapitre nous avons parlé du temps de réponse d'une architecture d'automatisation en réseau qui sera notre sujet dans le prochain chapitre.

Chapitre 4

Introduction

Dans ce dernier chapitre, nous allons voir comment synthétiser un contrôleur pour une architecture d'automatisation en réseau sous contrainte temporelle pour assurer un temps de réponse souhaité, Pour cela nous allons utiliser la méthode proposée par L.Houssin dans [14].

Tout d'abord nous allons présenter le problème de contrainte de temps imposé aux SED qui est un critère primordiale pour le bon fonctionnement de ces systèmes, on peut citer les travaux de [8,10], [12] et [15], ensuite nous allons exposer cette méthode de L.Houssin, enfin que l'on puisse l'appliquer sur le modèle obtenu dans la section 3.3.

4. Synthèse d'un contrôleur pour une architecture d'automatisation en réseau sous contrainte temporelle

De nos jours, la plupart des systèmes industrielles repose sur des architectures d'automatisations en réseau. Cependant les réseaux de communication utilisés dans les systèmes distribués, génèrent en particulier des retards durant les échanges de données voir des pertes d'informations, par exemple, dans l'industrie manufacturière on utilise parfois des procédés dont les temps d'activités doivent être compris entre des durées minimales et des durées maximales. C'est le cas par exemple de l'industrie utilisant des réactions chimiques pour assurer le traitement d'une pièce [27]. Il est clair que le décapage d'une pièce par immersion dans un bain d'acide nécessite un temps de trempe minimum,et ne doit pas dépasser un temps maximum, sous peine de détérioration de la qualité de la production.

Plusieurs méthodes ont été proposés pour remédier à ce problème on cite par exemple [15], [28] et [29]**.**

Dans ce travail on a comme problème, les contraintes imposées sur le temps de réponse d'une AAR.

4.1 Position du problème

Considérant un SED modélisé dans $M_{in}^{ax}[\gamma, \delta]$ par l'équation

$$x = Ax \oplus Bu \qquad (4.1)$$

on s'intéresse à la synthèse d'un contrôleur de type retour d'état pour les systèmes (max,+)-linéaires. Plus précisément, si on considère un SED modélisé par un GET en utilisant la transformé en $M_{in}^{ax}[\gamma, \delta]$, nous allons calculer un transfert pour le contrôleur de retour d'état. Ce transfert sera réalisé par un GET, et l'application du contrôleur conduira à lier le GET du contrôleur avec le GET du système. Dans ce GET contrôlé les arcs supplémentaires sont dues au contrôleur, ces arcs autorisent ou interdisent le franchissement des transitions contrôlées(figure 4.2) cette structure de contrôle est comparable à certaines méthodes de contrôle de réseau de Petri [30], la synthèse d'un contrôleur de retour d'état pour un GET à déjà été abordé dans les documents tels [28] et [29]. Dans ces travaux les retours d'états visent à retarder les événements dans le système le plus possible de sorte que le système contrôlé n'est pas plus long qu'un modèle de référence.

Dans cette méthode, l'objectif de contrôle est différent :

- Nous visons à assurer des contraintes données sur les états x (plutôt que de satisfaire une adaptation de model de référence) pour toutes les entrées, ces contraintes sont définis par une matrice $\boldsymbol{\phi}$ et donnée par l'inégalité implicite suivante :

$$\phi x \preccurlyeq x \qquad (4.2)$$

- Nous cherchons un retour d'état qui retarde le fonctionnement du système aussi peu que possible (ce qui rapporte les événements d'entrée aussi peu que possible, contrairement au critère juste-à-temps) en d'autres termes, nous cherchons à calculer le plus petit retour d'état de telle sorte que l'état du système commandé satisfait les contraintes données par l' équation (4.2).

Dans ce qui suit, nous illustrons 3 contraintes qui peuvent être imposés aux systèmes contrôlés comme l'inégalité (4.2). Ensuite, le problème de contrôle est formalisé et résolue comme une synthèse par retour d'état.

4.2 Spécification des contraintes

Maintenant, nous allons détailler 3 types de contraintes pour les SED modélisés par les GET qui peuvent être formulés par inégalité (4.2) :

- Certaines variables internes peuvent être soumise à une séparation de temps minimum entre 2 tirs successives pour une variable d'état x_i, et un intervalle de temps désigné par Δ_{min}nous exigeons que

  $$x_i(k+1) \geqslant \Delta_{min} x_i(k)$$

  Ensuite, en appliquant la transformé en $M_{in}^{ax}[\gamma, \delta]$, on obtient :

  $$\gamma\delta^{\Delta_{min}} x_i \leqslant x_i$$

- On peut également viser à délimiter le temps de séjour des jetons dans les chemins donnés d'un GET (contrainte de temps critique). considérant un chemin de la transition x_i à x_j contenant α jetons au départ, et on note τ le temps de séjour maximal souhaité dans ce chemins. Cela nous donne :

  $$x_j(k+\alpha) - x_i(k) \leqslant \tau$$

  Ceci peut être formulé en $M_{in}^{ax}[\gamma, \delta]$ par :

  $$\gamma^{-\alpha}\delta^{-\tau} x_i \leqslant x_i$$

- On peut également limiter le nombre de jetons dans certains chemins d'un GET. Considérant un chemin de la transition x_i à x_j contenant initialement α jetons, on note k le nombre maximum de jetons dans ce chemin.

Cette contrainte peut être décrite par :

$$\gamma^{k-\alpha} x_i \leqslant x_i$$

4.3 Formalisation

On considère un contrôleur de type retour d'état, désigné par F est ajouté entre l'état externe

x_c et l'entrée u. Le processus d'entré est décrit par $u = Fx_c \oplus v$

Avec v l'entrée de référence. Un tel contrôleur implique que les événements retardés ne sont que ceux des entrés. Les variables concernées sont celles de l'ensemble $U_c = \{u_i \,/\, B_{ii} = e\}$, l'évolution de l'état du système est décrite par :

$$x_c = Ax_c \oplus BFx_c \oplus Bv$$

On considérant la règle de fonctionnement au plutôt, la fonction de transfert de tel système est :

$$x_c = (A \oplus BF)^* Bv = H_c v \qquad (4.3)$$

Remarque : le retour d'état sur les entrées a un effet sur les variables d'états qui sont directement contrôlables, ces variables d'états x_i sont tel que $B_{ii} = e$. Ces variables d'états x_i sont tel que $x_i = u_i$, puisqu'il n y a pas de décalage entre elles, nous désignons cette ensemble par

$$X_C = \{x_{ii} |\, B_{ii} = e\}$$

De (4.3), il est évident que l'état du système contrôlé est tel que :

$$x_c \geqslant A^* Bv, \ \ \forall v$$

En outre, x_c devrait satisfaire l'objectif de la contrainte (4.2), qui est $x_c \geqslant \phi x_c$

Ce qui donne : $x_c \geqslant A^* Bv \oplus \phi x_c, \ \ \forall v$

Nous visons à retarder le système le moins possible, nous cherchons donc la transition x_c la moins contrôlée donnée par :

$$x_c \geqslant \phi^* A^* Bv, \ \ \forall v$$

On utilisant (4.3), on va chercher le plus petit retour d'état tel que :

$$(A \oplus BF)^* Bv \geqslant \phi^* A^* Bv, \ \ \forall v$$

$$\Leftrightarrow \ \ (A \oplus BF)^* B \geqslant \phi^* A^* B \qquad (4.4)$$

Hypothèse: la matrice des contraintes ϕ est sensé satisfaire $= \phi$. Cette hypothèse revient à formuler toutes les contraintes $\phi_{ij} x_j \preccurlyeq x_i$ (voir la section 4.2) c'est-à-dire $x_i \in X_c$, les états x_i qui sont directement contrôlables. Pour une contrainte donnée sur un chemin entre deux variables d'états $x_i \in X_c$ et $x_j \in X_c$, notre approche nécessite de refondre la contrainte telle que $x_i \in X_c$ où $x_j \in X_c$. Par exemple, considérerons le GET de la figure (4.1) et supposons que les jetons ne doivent pas rester plus que 4 unités de temps du trajet entre les transitions x_3 et x_5 . Par conséquent il conduit à une matrice ϕ qui ne satisfait pas l'hypothèse.

Néanmoins, il est possible de reformuler cette contrainte de manière à ce que l'hypothèse soit satisfaite. Au lieu de considérer la contrainte entre x_3 et x_5 on peut sélectionner le chemin entre x_1 et x_5 et un temps de séjour maximum de 5 unités de temps. Cette nouvelle contrainte implique également une nouvelle contrainte pour le temps de séjour entre x_1 et x_3 (au plus 2 unités de temps). Notez qu'une autre possibilité est de considérer le chemin, entre x_2 et x_5 et un temps de séjour maximum de 6 unités de temps. La proposition suivante donne une condition nécessaire et suffisante sur les contraintes données ϕ pour l'existence d'une réponse causale satisfaisante (4.5).

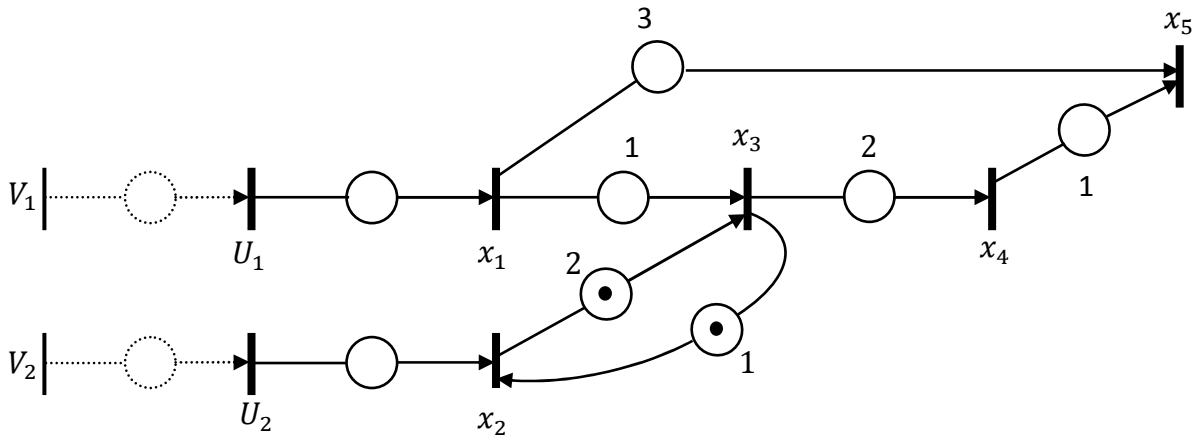


Figure 4.1 Exemple d'un GET P-temporisé

Corollaire 1[14] : soit x un élément $M_{in}^{ax}[\gamma, \delta]$ qui admet une approximation causale, chaque élément $y \preccurlyeq x$ admet aussi une approximation causale

Proposition[14] : il existe un retour d'état causal F qui satisfait (4.5) si et seulement si,$\phi^* A^* B$ admet une approximation causale. Si elle existe, l'approximation supérieure causale est liée par G.

Preuve :

$\Longrightarrow$Si un retour d'état existe, alors $(A \oplus BF)^* B$ est également causale (parce que A et B sont causales) .nous pouvons déduire du corollaire 1 et (4.5) que $\phi^* A^* B$ admet une approximation supérieur causale.

$\Longleftarrow$Si $\phi^* A^* B$ admet une approximation supérieure causale, on peut trouver un élément causale X telle que :

$$X \geqslant \phi^* A^* B. \text{ De } B^2 = B \; et \; B\phi^* = B(I \oplus \phi \oplus \phi\phi \oplus \dots) = \phi^*$$

Nous avons :

$$BXB \; \geqslant \phi^* A^* B$$

$$\Longrightarrow (BX)^* B \geqslant \phi^* A^* B (\text{de } a^* \geqslant a)$$

$$\Longrightarrow (BX)^* B \oplus A^* B \geqslant \phi^* A^* B$$

$$\Longrightarrow ((BX)^* B \oplus A)^* B \geqslant \phi^* A^* B$$

$$\Longrightarrow ((BX) \oplus A)^* B \geqslant \phi^* A^* B \qquad (\text{de } (a \oplus b^*) \geqslant a^* \oplus b^*)$$

Ce qui prouve l'existence d'un retour d'état causale nommé X qui satisfait (4.4).

Corollaire 2[14] : l'approximation supérieur causale G, si elle existe, est telle que :$GB = G$ et $BG = G$.

Preuve : nous démontrons d'abord que $GB = G$, de $B \leqslant e$, on a $GB \leqslant G$. de la proposition 1, G est telle que $G \geqslant \phi^* A^* B$ et on a $GB \geqslant \phi^* A^* B$ $(B = B^2)$ .la matrice GB est causale puisque G et B le sont. Et comme G est le plus petit élément causale supérieur à $\phi^* A^* B$, on déduit que $GB \geqslant B$. Pour le même raisonnement on peut facilement prouver que $GB = G$.

Remarque : dans [31] compte tenue des fonctions dateurs sur un horizon de prédiction, les systèmes (max,+)-linéaires sont décrit par l'équation d'état $x(k) = Ax(k-1) \oplus Bu$ et l'équation de sortie $y(k) = Cx(k)$, le model prédictive de control (MPC) est étendue à cette

classe de systèmes en définissant un horizon de control, un critère de coût ainsi que des contraintes données par :

$$E(k)u(k) + F(k)(y) \leq \ h(k) \ldots \ldots (4.5)$$

Dans lesquelles $E(k)$, $F(k)$ et $h(k)$ (matrices de dimensions adéquates, sont choisit en fonction des objectifs de control). Le MPC a été considérer pour le programme de sortie en juste à temps dans [31], et il peu aussi être appliqué aux problèmes de contrôle actuel. Mentionnant les contours de telle formulation.

- On considère un critère de coût qui va commander la minimisation des instants de temps des entrées.
- On admet que chaque contrainte exprimé en (4.2) appliquer entre une entrée et une sortie d'un système pour le remanier (avec possibilité d'augmenter l'horizon de prédiction) tel que exprimé en (4.5).
- Prenons en compte l'entrée v comme inégalité $-\mathrm{u(k)} \preccurlyeq -\mathrm{v(k)}$compatible avec (4.6).Formuler de cette façon, c'est-à-dire comme un problème d'optimisation quasi-connexe, plusieurs algorithme on été proposés pour résoudre le problème de MPC. Il devrait être claire que cette solution dépend de v. cela implique que v doit être connu (au moins dans l'horizon de prédiction) et que la loi de contrôle doit être mise œuvre en ligne. En revanche, les retours d'états proposés dans cet article sont calculés hors connexion (utilisant la proposition 2), et sont valables pour toute entrée de référence possible v. (v est supposé être connu). En outre avec l'approche MPC chaque contrainte doit être appliquée entre l'entée et la sortie du système. et cela est plus restrictif que l'hypothèse 1.

4.4 Calcul du retour d'état :

Dans cette section, nous allons voir comment calculer la solution de (4.4).

Proposition 2 : supposant que $\phi^* A^* B$ admet une approximation supérieure causale notée G (condition suffisante et nécessaire pour l'existence d'un retour d'état causal satisfaisant (4.4)). Les solutions de (4.4) sont les éléments de Q_g, avec g : $F \to B$ $(G \ominus (A \oplus BF)^*)$

Preuve : les retours d'état causales utilisés sont tel que :

$(A \oplus BF)^* B \succcurlyeq G$

$\Leftrightarrow (A \oplus BF)^* \succcurlyeq G$ (de$GB = G$ et$B \preccurlyeq e$)

$\Leftrightarrow BF \oplus (A \oplus BF)^* \succcurlyeq G$ (de $(A \oplus BF)^* \succcurlyeq BF$)

$\Leftrightarrow BF \succcurlyeq G \ominus (A \oplus BF)^*$ (de $T_{(A\oplus BF)^*}$est dualement residuée)

$\Leftrightarrow F \succcurlyeq B(G \ominus (A \oplus BF)^*)$

Pour la dernière équivalence :

$(\Longrightarrow) BF \succcurlyeq G \ominus (A \oplus BF)^* \Longrightarrow B^2F \succcurlyeq B(G \ominus (A \oplus BF)^*)$

$\Longrightarrow F \succcurlyeq B(G \ominus (A \oplus BF)^*)$ (de $F \succcurlyeq BF = B^2F$)

$(\Longleftarrow) F \succcurlyeq B(G \ominus (A \oplus BF)^*) \Longrightarrow BF \succcurlyeq B^2(G \ominus (A \oplus BF)^*)$

$\Longrightarrow BF \succcurlyeq B(G \ominus (A \oplus BF)^*)$

$\Longrightarrow BF \succcurlyeq B\,G \ominus B(A \oplus BF)^*$ (de $a(x \ominus b) \succcurlyeq ax \ominus bx$)

$\Longrightarrow BF \succcurlyeq BG \ominus (A \oplus BF)^*$(x→a$\ominus$ b est antitone)

$\Longrightarrow BF \succcurlyeq G \ominus (A \oplus BF)^*(BG = G)$

Corollaire 3 : le calcul de la somme $v = sup\ F_{g^2}$ donne un retour d'état assurant (4.5) (depuis $v \in Q_g$)

Remarque : pour résumer, la proposition 1 donne une condition nécessaire et suffisante pour l'existence d'une solution à notre problème de contrôle.

Si cela est satisfait, alors le corollaire 3 indique comment calculer une solution à savoir $v = SupF_{g^2}$. v est une bonne solution pour notre problème de contrôle, car il se rapproche ou corresponds au retour d'état minimal.

4.5 Application de la méthode sur le modèle de la section 3.3

4.5.1 Problématique

Dans notre modèle, notre objectif est de satisfaire le temps de réponse qui est compris entre les deux transitions t_7 et t_{10} .

On a de (4.1)

$$x(k) \succcurlyeq \phi^* x(k)$$

Donc on a une seule contrainte :

$$x_7(k) \succcurlyeq \phi^* x_{10}(k)$$

Donc dans$M_{in}^{ax}[\gamma, \delta]$ on aura de (3.4) :

$$x_7(k) \succcurlyeq \delta^{-19} x_{10}(k)$$

Ce qui donne :

$$\phi = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^{-19} \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}$$

les calculs on était fait avec le logiciel SCILAB 3.1.1

Présentation de SCILAB :

SCILAB (scientific laboratory) est un logiciel de calcul numérique multi-plateforme fournissant un environnent de calcul pour des applications scientifiques, il possède un langage de programmation orienté calcul numérique de haut niveau. il peut être utilisé pour le traitement de signal, traitement statistiques, traitement d'images, l'optimisation numérique et modélisation et simulation des systèmes explicites et implicites.

La syntaxe et les possibilités offertes par SCILAB sont similaires à celles de MATLAB.

Calcul de ϕ^+ :

$$G \succcurlyeq \phi^+ A^* B$$

$$\phi^+ = (\phi \oplus \phi\phi \oplus \dots)$$

$$\phi^+ = \begin{bmatrix} e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \delta^{-19} \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e \end{bmatrix}$$

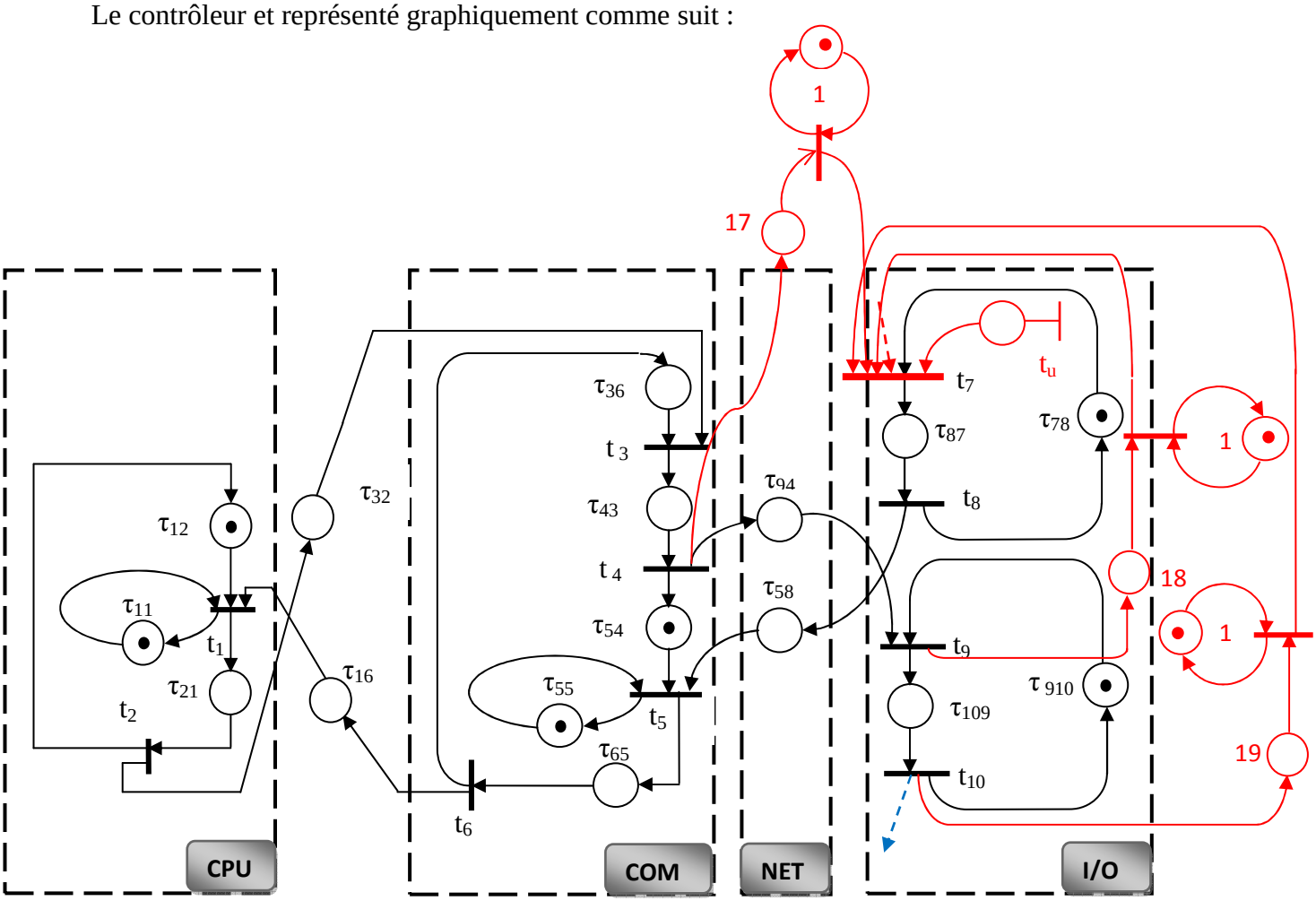
Calcul de la matrice A et B en utilisant la transformé en $M_{in}^{ax}[\gamma, \delta]$:

$$A = \begin{bmatrix} \gamma^1\delta^5 & \gamma^1 & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \delta^2 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \gamma^1 & \gamma^1\delta^{10} & \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \gamma^1 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \delta^1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \gamma^1 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^1 & \varepsilon \end{bmatrix}; B = \begin{bmatrix} \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ \varepsilon \\ e \\ \varepsilon \\ \varepsilon \\ \varepsilon \end{bmatrix}$$

Après calcul on aboutit à :

$$F = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^{17}(\gamma^1\delta^1)^* & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^{18}(\gamma^1\delta^1)^* & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^{19}(\gamma^1\delta^1)^* & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}$$

Le contrôleur et représenté graphiquement comme suit :



Conclusion

Dans ce quatrième chapitre on a vu une méthode [14] qui sert à calculer un contrôleur pour améliorer le temps de réponse d'un système. Cette méthode consiste à ramener la commande à l'entrée du système. Apres l'application de cette méthode sur notre système.

Au regard des recherches qu'on a effectué, nous constatons que les résultats obtenues ne correspondent pas avec exactitude aux résultats escomptés au début de notre travail, néanmoins nous avons procédé à plusieurs essaies en espérant nous approcher considérablement de l'objectif fixé. En ce sens, ce présent mémoire pourrait constituer l'entame d'une ou de plusieurs recherches approfondies dans le domaine.

Conclusion générale

Conclusion générale

Ce mémoire porte sur le problème de commande des architectures d'automatisations en réseau sous contraintes temporelles.

Dans le premier chapitre nous avons fait un rappel sur l'algèbre des dioïdes et treillis qui sont les outils algébriques que nous avons utilisés dans les calculs.

Dans le deuxième chapitre nous avons vu comment modéliser les systèmes à événements discrets à l'aides des réseaux de Petri et la représentation de leurs dynamique à l'aide de l'algèbre de (max,+), nous avons vu aussi qu'une représentation d'états linéaire peut être obtenue à partir du model d'état.

Dans le troisième chapitre nous avons présenté les architectures d'automatisation en réseau, et détailler le fonctionnement de ces différent composant, ensuite nous avons modélisé une AAR avec une sous classe des réseaux de Petri qu'on nomme les graphes d'événements temporisés, à partir de cette représentation graphique nous avons représenté le comportement des ce graphe en utilisant l'algèbre de (max,+). La fin de ce chapitre est consacrée au temps de réponse de ces architectures.

Le quatrième et dernier chapitre nous avons vu comment satisfaire le temps de réponse des architectures d'automatisation en réseau en synthétisant un correcteur avec la méthode proposée dans [14].

L'originalité de cette méthode réside dans la définition d'un nouvel objectif pour la commande en juste-à-temps des systèmes (max; +)-linéaires.
Contrairement aux commandes proposées avant, qui visent à imposer un transfert entrée-sortie ou une poursuite de trajectoire en sortie, nous avons considéré comme objectif le respect d'un ensemble de contraintes définies sur l'état du système. L'apport de cette méthode est de pourvoir spécifier, de façon plus précise, le comportement désiré du système. Les résultats sur les points fixes d'applications isotones définies sur des dioïdes sont utilisés pour la synthèse de la loi de commande.

Les correcteurs adjoints aux systèmes (*max; +*)-linéaires peuvent seulement ralentir le système. Les critères employés avant visent à ralentir au maximum le système tout en assurant un objectif de commande. Dans ce mémoire, nous avons proposé un correcteur qui au

contraire ralentit le moins possible la dynamique du processus tout en satisfaisant l'objectif de commande (le respect d'un ensemble de contraintes sur l'état du système).

Annexe

Programme de calcul du contrôleur avec :

SCILAB 3.1.1

```
A=smatrix(10,10);
B=smatrix(10,10);
P=smatrix(10,10);
G=smatrix(10,10);
F=smatrix(10,10);
Y=smatrix(10,10);
X=smatrix(10,10);
//Déclaration de la matrice d'état :
A(1,2)=series(eps,[1 0],e);
A(2,1)=series(eps,[0 1],e);
A(3,2)=series(eps,[0 1],e);
A(3,3)=series(eps,[1 10],e);
A(3,8)=series(eps,[1 0],e);
A(4,3)=series(eps,[0 1],e);
A(5,4)=series(eps,[0 0],e);
A(5,5)=series(eps,[1 5],e);
A(5,6)=series(eps,[1 0],e);
A(6,5)=series(eps,[0 2],e);
A(7,4)=series(eps,[0 0],e);
A(7,6)=series(eps,[0 0],e);
A(7,8)=series(eps,[0 1],e);
A(9,8)=series(eps,[0 1],e);
A(9,10)=series(eps,[1 0],e);
A(10,9)=series(eps,[0 1],e);
// Déclaration de la matrice B
B(1,1)=series(eps,[0 0],e);
// Déclaration de la matrice P
P(1,10)=series(eps,[0 -19],e);
//Calcul de A* :
AA= stargd(A);
//Calcul de P* :
PP= P*stargd(P);
//Calcul de G :
G=PP*AA;
G=B*G;
X1=G/AA;
X1=B*X1;
X1=prcaus(X1);
X1=A+X1;
Y=stargd(X1);
Y=prcaus(Y);
Y=G/Y;
X2=B*Y;
Y=prcaus(X2)
Y=A+Y;
Y=stargd(Y);
Y=prcaus(Y);
Y=G/Y;
X3=B*Y;
Y=prcaus(X3)
```

Bibliographie

BIBLIOGRAPHIE

[1] F. Baccelli, G. Cohen and G. J. Olsder and J. P. Quadrat, 1992, Synchronization and Linearity,Wiley.

[2] T. S. Blyth, M. F. Janowitz, 1972, Residuation Theory, Pergamon press.

[3] A. Tarski, 1955, A lattice theorical fixed point theorem and its applications.

[4] B. A. Davey and Priestley, 1990, Introduction to lattices and order.

[5] S. Lahaye, 2003, Sur les points fixes d'applications définies sur des dioïdes complets.

[6] S. Gaubert, 1992, Théorie des systèmes linéaires dans les dioïdes, Thèse

[7] G. Cohen, 1993, Two-dimensional domain representation of timed event graphs.

[8] B.Cottenceau, L.Hardouin, JL.Ferrier, 1999, synthesis of greatest linear feedback for timed event graphs in dioid. IEEE Trans Autom Control 44(6): 1258-1262.

[9] K. Tebani, R.Kara, S.Amari,2013, Commande (Max, +) pour garantir un temps de réponse dans les systèmes de commande en réseau.

[10] K. Tebani, R.Kara, S.Amari,2016, closed-loop control of constrained discrete event systems : application to a networked automation systems.

[11] K.Tebani, 2012, mémoire de magister, commande des systèmes à événements discrets à temps critique, application aux systèmes de commande en réseau.

[12] S.Amari, K.Tebani, R.Kara, 2012, Modélisation et commande d'une architecture d'automatisation en reseau sous contrainte temporelle en utilisant l'algèbre (Min,+). 9th International Conference on Modeling, Optimization & SIMulation, Jun 2012, Bordeaux, France.

[13] L.Houssin.2006, Contributions à la commande des systèmes (max,+)-linéaires. Applications aux réseaux de transport.

[14] L.Houssin, S.Lahay, JL.Boimond, 2013, control of (max,+)-linear systems minimizing delays.

[15] S.Amari, 2005, commande des graphes d'événements temporisés sous contraintes temporelle, thèse de doctorat, université de Nantes.

[16] A.Nait-Sidi-Moh, M-A.Manier, A.Elmoudni, 2001, Modélisation et évaluation d'un système de transport par l'algèbre max-plus .

[17] B.Addad, 2011, évaluation analytique du temps de réponse des systèmes de commande en reseau en utilisant l'algèbre (max,+), thèse de doctorat, ENS Cachan.

[18] M.Gondran, Minoux, 2001, Graphe, dioïde et semi-anneau.

[19] O,Boutin, 2009, Thèse doctorat, Modélisation de conflits et calcul de bornes dans les systèmes de production par la théorie des dioïdes .

[20] MK.Didi Alaoui, 2005, thèse de doctorat, Etude et supervision des graphes d'événements temporisés et temporels : vivacité, estimation et commande

[21] I.Ouerghi, 2006, thèse de doctorat, Etude de systèmes (max, +)-linéaires soumis à des contraintes, application à la commande des graphes d'événements P-temporel.

[22] P.Moller, 1988, Théorie algébrique des systèmes à événements discrets.

[23] L.Hardouin, 2008, Sur la Commande des Systèmes (max,+) Linéaires.

[24] CA. Petri, 1962, Kommunikation mit Automaten.

[25] T.Murata,1989, Petri nets : properties, analysis and application .

[26] F.Olivier, 2011, Automates & langages.

[27] W.Khansa, J-P Denat, and S.Collart-Dutilleul. (1996). P-Time Petri Nets for Manufacturing Systems. In Workshop On Discrete Event Systems (WODES'1996), Edinburgh, (U.K.).

[28] L.Hardouin, B.Cottenceau, JL Boimond et JL.Ferrier, 2001, model reference control for timed event graphs dioids.

[29] C.A Maia, L.Hardouin, R.Santos Mendes and B.Cottenceau, 2003, optimal closed-loop control for timed events graphs in dioid.

[30] L.E Hollaway, B.H Krogh and A.Giua, 1997, A survey of Petri net methods for controlled discrete event systems. Journal of Discrete Events Dynamic Systems.

[31] Bart de Schutter and Ton van den Boom, 2001, Model predictive control for (max,+)-linear discrete event systems.

[32] G.Cohen, D.Dubois, J.P.Quadrat, and M.Viot, (1983). Analyse du comportement périodique des systèmes de production par la théorie des dioïdes. Rapport de recherche 191, INRIA, Le Chesnay, France. pages 17, 33

[33] P.Lotito, E.Mancinelli, and J.P.Quadrat, (2005). A minplus derivation of the fundamental car-traffic law. IEEE Trans. on Automatic Control, 50(5):699–705. Pages 33

Commande des Systèmes à événements discrets à temps critiques :
Application aux systèmes de commande en réseau

Résumé :

Dans ce mémoire nous avons calculé un contrôleur qui va satisfaire un temps de réponse maximal pour une architecture d'automatisation en réseau, en utilisant la méthode de L.Houssain.
Pour cela nous avons rappelé les outils algébriques nécessaires dans le premier et le deuxième chapitre. Dans le 3 eme chapitre nous avons présenté les architectures d'automatisation en réseau.
Enfin dans le dernier chapitre, nous avons fait une application pour calculer le contrôleur.

Mots clé :

Discrets

Réseaux de Petri

Max(+)

Min(+)

Contrôleur

Commande

Evènements discrets

Commande en réseau

Architecture d’automatisation en réseau

Temps de réponse

Dioide

Gama delta

Rdp

AAR