LABORATOIRE d'**I**NGÉNIERIE des **S**YSTÈMES **A**UTOMATISÉS

EA 4014 – Université d'Angers

Institut des **S**ciences et **T**echniques de l'**I**ngénieur d'**A**ngers

Juin 2008

Master2 Recherche

Spécialité : Systèmes Dynamiques et Signaux

Présenté par

Codjo Hermann ZANNOU

à l'ISTIA-Université d'Angers

ALGRORITHME DE FOURIER – MOTZKIN. APPLICATION AUX GRAPHES D'ÉVÉNEMENTS TEMPORISÉS.

Responsable de stage : Philipe DECLERCK
Laboratoire : LABORATOIRE D'INGENIERIE DES SYSTEMES AUTOMATISÉS.
62 avenue Notre Dame du Lac, F-49000 ANGERS

UNIVERSITE D'ANGERS

ALGRORITHME DE FOURIER – MOTZKIN
APPLICATION AUX GRAPHES D'ÉVÉNEMENTS TEMPORISÉS

Codjo Hermann ZANNOU

Université d'Angers

Remerciements

Je voudrais tout d'abord remercier le Professeur **Jean-Louis Ferrier**, ancien directeur du laboratoire d'Ingénierie des Systèmes Automatisés (LISA), pour son accueil au sein du laboratoire LISA.

Je remercie le Professeur **Jean-Louis Boimond** de m'avoir donner d'occasion de réaliser mon stage de Master2 Recherche au sein du laboratoire d'Ingénierie des Systèmes Automatisés d'Angers dont il est l'actuel directeur.

Je remercie le Professeur Bertrand Vigouroux, responsable du Master2 Recherche Systèmes Dynamiques et Signaux pour son écoute et ses conseils.

J'adresse mes sincères remerciements à mon encadrant Monsieur **Philipe Declerck**, Maître de conférences à l'Université d'Angers pour sa disponibilité, ses conseils et son aide tout au long de ce stage.

Enfin mes remerciements vont également à tous les membres du LISA ainsi qu'à mes collègues.

A mon père Léopold ZANNOU ,
ma regrettée mère Sidonie Aïssi,
mes frères et sœurs,
et mes amis.

Résumé

Dans ce rapport, nous avons travailler essentiellement sur les systèmes d'inéquations linéaires de la forme $\boldsymbol{Ax \leq b}$; ce qui nous a amené à :

- Développer un programme Scilab basé sur l'algorithme de Fourier-Motzkin pour l'analyse et la résolution des systèmes d'inéquations $\boldsymbol{Ax \leq b}$.
- Modéliser les graphes d'Événements Temporisés sous la forme algébrique $\boldsymbol{Ax \leq b}$ et appliquer l'algorithme de Fourier-Motzkin pour tracer les trajectoires temporelles des systèmes à Événements Discrets.

Table des Matières

Introduction

On se pose souvent la question d'existence d'une solution pour un système d'inéquations linéaires de la forme $\boldsymbol{Ax \leq b}$. Il est important d'avoir une réponse à cette question. En effet, il n'est d'aucun intérêt de travailler sur un système qui n'admet aucune solution.

Joseph Fourier fut le premier à saisir l'importance pratique du maniement des inégalités, à la suite de ses recherches sur la théorie de la chaleur, il mit au point une méthode de résolution directe des systèmes d'inéquations linéaires $\boldsymbol{Ax \leq b}$. Il fit connaître les principes de calcul des conditions d'inégalités dans une série de mémoires au cours des années 1823-1826. Successivement redécouvert par Dines en 1918 et Motzkin en 1936 pour donner jour à l'algorithme de Fourier-Motzkin.

Ce rapport comprend quatre chapitres :

- Dans le premier chapitre intitulé « Quelques rappels sur les réseaux de Petri », Nous avons fait des rappels sur les réseaux de Petri et les gaphes d'Événements Temporisés qui représentent une sous classe particulière des réseaux de Petri.
- Dans le deuxième chapitre intitulé « Algorithme de Fourier-Motzkin », nous avons présenté l'algorithme de Fourier-Motzkin. Nous avons fait également une analyse détaillée sur l'algorithme, ses possibilités et son implémentation informatique sous Scilab.
- Dans le troisième chapitre intitulé « Un exemple d'application de l'algorithme de Fourier-Motzkin, nous allons utiliser le programme développé dans chaque le chapitre2 pour tracer les trajectoires temporelles des graphes d'événements.
- Le quatrième chapitre nous avons la conclusion et les perspectives.

Chapitre 1

Quelques rappels sur les réseaux de Petri

1.1 Définition

Un réseau de Petri Rdp, est un graphe constitué de deux types de nœuds appelés places et transitions. Par convention, chaque place est représentée par un cercle tandis qu'une transition est schématisée par un rectangle (dans d'autres cas par une barre épaisse). Les nœuds de Rdp sont reliés entre eux par des arcs orientés. Chaque arc ne peut relier deux nœuds de même type. Tous les arcs sont pondérés, autrement dit, à chaque arc est associé un poids sous forme d'un entier positif non nul. Les places peuvent contenir chacune un nombre entier positif ou nul de marquages (jetons), d'où la notion de marquage d'un Rdp. Le marquage d'un Rdp est donc un vecteur dont la dimension est égale au nombre de places que contient le Rdp, et chaque composante à la valeur du nombre de jetons que contient la place correspondante.

1.2 Présentation des réseaux de Petri [1][2]

D'une manière générale, un réseau de Petri est donné par $PN = (P,T,A,W,M_0)$ où :

- $P = \{P_1,P_2,....\}$ ensemble fini de places
- $T = \{t_1,t_2,...\}$ ensemble fini de transitions
- $A \subset (P \times T) \cup \{T \times P\}$ ensemble fini d'arcs
- W : A→{1,2,…} la fonction poids attachée aux arcs
- M_0 : P→ {1,2…} le marquage initial

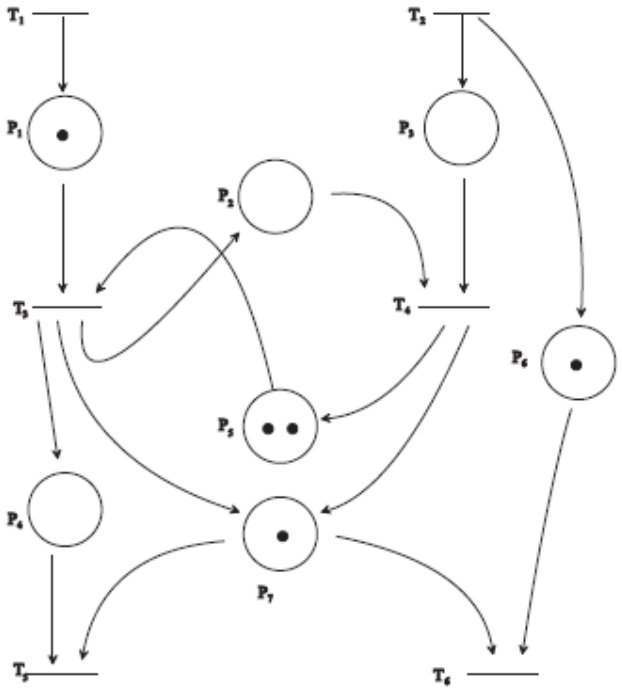


Figure1.1 : Réseau de Petri

La **figure1.2** montre bien un Rdp, qui contient six transitions et sept places. t_1 et t_2 sont dites : transitions sources, t_5 et t_6 sont des transitions puits. Le vecteur M_0^t = (1,0,0,0,2,1,1) représente le marquage initial du Rdp ci-dessus qui représente aussi l'état initial du système. Pour définir sa dynamique on se base sur les règles d'évolution du marquage. En effet, dans cet exemple t_3 est validée (peut être franchie) car ces places amont P_1 et P_5 contiennent chacune un nombre de jetons suffisant pour franchir t_3. Ce franchissement va consister donc à prélever une marque de chacune des places P_1 et P_5 et ajouter une marque dans des places avales de t_3 qui sont P_4, P_7 et P_2 puisque le poids de tous les arcs vaut 1. En revanche, dans le cas général, le franchissement d'une telle transition consiste à prélever un nombre de marques de chacune de ses places en amont égale au poids de chaque arc qui les relie, et à ajouter à chacune de ses places avales un nombre de jetons égal au poids de chaque arc qui les relie.

L'équation fondamentale qui caractérise l'évolution et la dynamique d'un Rdp est :

$\boldsymbol{M_k = M_0 + W.S}$ où :

M_k : le nouveau marquage obtenu à partir du marquage M_0 après une séquence de franchissements σ

S : un vecteur qui représente cette séquence σ. Sa dimension est égale au nombre de transitons que contient le Rdp. Chaque composante S_j vaut le nombre de fois où la transition correspondante t_j a été franchie pendant la séquence σ.

W : une matrice dite d'incidence du (Rdp) définie de la façon suivante : $W = W^+ - W^-$

W^- : appelée matrice d'incidence avant tel que : $W^- = [w_{ij}^-]$

w_{ij}^- = Pré(pi,t_j) : poids de l'arc qui relie une transition t_j à une place en amont pi

W^+ : appelée matrice d'incidence arrière tel que : $W^+ = [w_{ij}^+]$

w_{ij}^+ = Pos(pi,t_j) : poids de l'arc qui relie une transition t_j à une place en avale pi

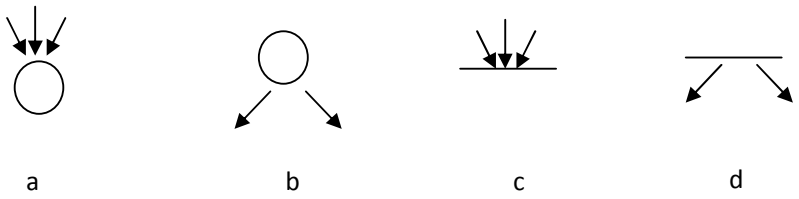


Figure1.2 : Phénomènes de synchronisation et concurrence

Dans l'étude des systèmes dynamiques à événement discrets (SDED), on retrouve fréquemment des problèmes de type concurrence, synchronisation et parallélisme. La **Figure1.2** ci-dessus montre bien la modélisation de ces phénomènes par Rdp :

-Le problème de concurrence à la fourniture de jetons dans une place est schématisé par la **figure1.2.a** tandis que la concurrence de la consommation des jetons d'une place, autrement dit un conflit structurel est représenté par la **figure1.2.b**.

-les **figures** (**1.2.c** et **1.2.d**) représentent la synchronisation à la consommation et à la fourniture de jetons à plusieurs places.

1.3 Graphes d'Evénements Temporisés

On a vu précédemment comment on peut modéliser des problèmes tels que la concurrence et la synchronisation par les Rdp. Dans ce qui suit, on va aborder une sous classe importante des Rdp appelée Graphes d'Événements Temporisés (GET), avec laquelle on pourra étudier des phénomènes de synchronisation et de parallélisme et non de concurrence d'où la définition suivante :

1.3.1 Définition

Un Graphe d'événement, est un Rdp tel que toute place a exactement une transition en amont et une transition en aval, avec une pondération de tous les arcs à la valeur 1.

D'une manière analogue, on définit un graphe d'état comme une forme duale de graphe d'événement. C'est une sous classe des (Rdp) tel que chaque transition a une seule place en amont et une seule place en aval.
Les graphes d'événements nous permettent donc d'étudier et de modéliser des phénomènes de concurrences ou de conflit. Dans le cadre de notre étude, on se limitera à l'utilisation et l'exploitation des Graphes d'Événements Temporisés.

1.3.2 Propriétés des graphes d'événements

• **Théorème [2]** : soit X = [x_1 . . . x_n] un vecteur à composantes entières non négatives, où n est nombre de places du graphe d'événements considéré. Soit γ un circuit élémentaire de ce graphe et $P = \{p_1,\ldots,p_n\}$ l'ensemble des places du graphe.
Alors si, pour i = 0 à n

$$x_i = \begin{cases} 1 \text{ si } p_i \in \gamma \\ 0 \quad \text{sinon}. \end{cases}$$

X est un P-invariant. Autrement dit, le nombre de jetons du circuit élémentaire γ est invariant quelque soit l'évolution du système.
Le formalisme mathématique qui nous permet de chercher tous les P-invariants est donné par : $X.W = 0$: W : matrice d'incidence, X : vecteur ligne

• **Théorème** : Soit un graphe d'événements avec m transitions. Le vecteur à m composantes toutes égales à 1 est l'unique T-invariant. Une autre manière d'exprimer cette est de dire que l'on retrouve le même marquage initial près avoir franchi une seule fois chaque transition. L'équation mathématique qui génère cette propriété est :

$$W^t.X = 0$$

1.3.3 Temporisation des graphes d'événements

L'introduction d'un nouveau paramètre qui est le temps sur les graphes d'événements va nous permettre de définir les graphes d'événements temporisés (GET). Deux façons se présentent :

• On peut associer à chaque transition t une durée minimale de tir $\theta(t)$ qui représente le temps de réservation d'un jeton dans une place amont avant d'être disponible à nouveau dans une place aval (GET T-temporisés).

• On peut associer à chaque place une durée minimale $\theta(p)$ qui représente le temps d'indisponibilité d'une marquage après son arrivée a cette place avant d'être utile pour un nouveau franchissement (GET P-temporisés).

Dans les deux cas, la temporisation est associée soient aux places, soient aux transitions. On peut montrer l'existence d'une équivalence entre les GET P-temporisés et T-temporisés, du fait qu'on a la possibilité de transformer une place temporisée en une transition temporisée et vice-versa.
Dans le cadre de notre étude, on considère des durées constantes et donc des GET déterministes. Néanmoins, il existe d'autres GET dans lesquels la temporisation n'est pas constante car elle est générée par des variables aléatoires : on est dans le cas des GET stochastiques. Ces derniers permettent de modéliser des tâches avec des temps d'exécution non déterministes (modélisation des pannes par exemple).

1.3.4 Description aux dateurs

On considère l'exemple suivant :

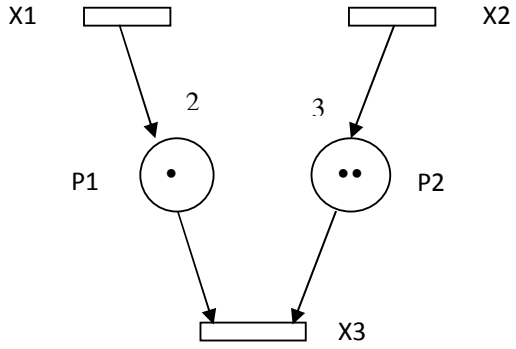


Figure2.3 Description aux dateurs

Pour la transition x_3, et à partir d'un marquage initial $M_0 = [1\ 2]^t$ à un instant initial t_0 (on suppose $t_0 = 0$), on va commencer à numéroter tous les événements (un événement correspond au tir d'une transition) en les initialisant avec une valeur k qui est un nombre entier qui peut être négatif et qui est le même pour x_1 et x_2. De ce fait, notre domaine de travail est l'espace des événements appelé domaine événementiel. Sur ce dernier on définit la fonction $x_3(k)$ par :
$x_3(k)$: date à laquelle l'événement numéroté k (tir) associé à x_3 a lieu.
D'une manière générale $x_i(k)$ est une fonction monotone (croissante) du fait qu'on avance dans le temps lorsque les événements se succèdent, donc :

$$x_i(k+1) \geq x_i(k)$$

Considérons l'exemple de la **figure2.3** , on cherche à mettre en relation les fonctions $x_1(k), x_2(k)$ et $x_3(k)$. Le calcul se fera sur deux étapes :

- Dans un premier temps : entre x_3 et x_1 il existe la place p_1 temporisée à deux unités de temps, et contenant déjà un jeton. L'activation de x_3 pour la $k^{ième}$ fois se fait après la date d'activation de x_1 pour la $(k-1)^{ieme}$ fois qui a fourni le jeton existant à p_1 à cet instant. De plus,

ce jeton doit séjourner deux unités de temps dans p_1 avant d'être disponible pour franchir x_3. On peut donc écrire l'inéquation suivante :

$$x_3(k) \geq x_1(k-1) + 2$$

- Dans un deuxième temps, on se base sur le même raisonnement pour établir une inéquation entre x_3 et x_2 :

$$x_3(k) \geq x_2(k-2) + 3$$

1.4 Conclusion

Dans ce chapitre, nous avons présenté un aperçu sur les réseaux de Petri et les Graphes d'Événements Temporisés qui représentent une sous classe des réseaux de Petri. Dans ce rapport, nous nous intéressons uniquement aux Graphes d'Événements Temporisés.

Chapitre 2

Algorithme d'élimination de Fourier-Motzkin

Cette procédure a été inventé par Joseph Fourier en 1826 et a été successivement redécouvert par Dines en 1918 et Motzkin en 1936.
Elle est applicable à des systèmes d'inéquations linéaires de la forme :

$$\boldsymbol{Ax} \leq \boldsymbol{b} \qquad \text{avec :}$$

A : Une matrice de dimension mxn à valeurs dans IR.
m : le nombre d'inégalités du système
n : Le nombre de variables du systèmes d'inéquations
b : Vecteur de dimension mx1 à valeurs dans IR
$x = [x_1\ x_2\ \ldots\ldots x_n]^T$: les variables du système d'inéquations

2.1 Objectif de la méthode :

Elle consiste à éliminer successivement les variables du système d'inéquations linéaires en générant de nouvelles inégalités. **Elle permet de tester l'existence de solution pour les systèmes d'inéquations linéaires** $\boldsymbol{Ax} \leq \boldsymbol{b}$ (phase de descente) **et de trouver une solution quelconque si elle existe** (phase de remontée)**.**

2.2 Algorithme de Fourier - Motzkin

Nous décrivons maintenant les deux phases de l'algorithme.

2.2.1 Phase de descente

Soit une matrice A à m lignes, et n colonnes et un vecteur b de taille m tel que $\boldsymbol{Ax} \leq \boldsymbol{b}$

- On considère la première colonne de A et donc la première variable x_1. Le vecteur composé des autres variables est noté $\boldsymbol{x'} = [\boldsymbol{x_2\ x_3\ \ldots\ x_n}]^{\mathbf{T}}$.
  On peut multiplier chaque ligne du système d'inéquations $\boldsymbol{Ax} \leq \boldsymbol{b}$ par un scalaire tel que le coefficient de la première colonne soit 1 ou -1 s'il est différent de 0. La forme **(S)** après classement des m lignes est :

$$\textbf{(S)} \begin{cases} x_1 + a_i x' \leq b_i , & (i = 1,\ldots\ldots\ldots, m') , I_+ \\ -x_1 + a_i x' \leq b_i , & (i = m'+1,\ldots m'') , I_- \\ 0x_1 + a_i x' \leq b_i , & (i = m''+1,\ldots, m) , I_0 \end{cases} \qquad \textbf{[1]}$$

  a_i est la ligne de la matrice A avec la première composante x_1 supprimée.

b_i est la i^{ème} composante du vecteur *b*.

I_+ : le système d'inéquations formé par la première ligne de (S)
I_- : le système d'inéquations formé par la première ligne de (S)
I_0 : le système d'inéquations formé par la troisième ligne de (S)

- En considérant les deux premières lignes du système **(S)**, on peut donc écrire la condition (C) suivante sur la variable x_1. Celle–ci est encadrée par une borne inférieure et une borne supérieure.

[2] $$I = \underset{m'+1\le j\le m''}{Max}\left(a_j x' - b_j\right) \le x_1 \le \underset{1\le i\le m'}{Min}\left(b_i - a_i x'\right) = S \quad \textbf{(C)}$$

La variable inconnue x_1 peut maintenant être éliminée. En partant du fait que chaque borne supérieure de x_1 doit être supérieure ou égale à chaque borne inférieure de x_1, le système **(S)** est donc équivalent à :

$$\begin{cases} a_j x' - b_j \le b_i - a_i x' , & \left(i = 1,...,m' \;;\; j = m'+1,.....,m''\right) \\ a_i x' \le b_i , & \left(i = m''+1,.....,m\right) \end{cases} \Leftrightarrow$$

Ce dernier système est équivalent au **nouveau système (S')** suivant :

(S') $$\begin{cases} \left(a_i + a_j\right)x' \le b_i + b_j , & \left(i = 1,...,m' \;;\; j = m'+1,.....,m''\right) \\ a_i x' \le b_i , & i = m''+1,....,m \end{cases} \quad \textbf{[3]}$$

Dans ce nouveau système **(S')**, la variable x_1 est éliminée. D'autre part, nous avons $m'(m'' - m') + (m - m'')$ contraintes et **(*n*-1)** inconnues.

- La procédure est répétée jusqu'à l'élimination de **(*n*-1)** composantes de *x*. On a alors un système d'inéquations à une inconnue x_n.

2.2.2 Phase de remontée

Connaissant la condition sur la variable x_n et en lui attribuant une valeur, on peut remonter à la condition sur la variable x_{n-1} ainsi de suite on remonte à la variable x_1.
En effet toute solution $\boldsymbol{x}'$ du système **(S')** peut être étendue à une solution $[\boldsymbol{x_1}\ \boldsymbol{x}']^T$ du système **(S)**, de façon que la relation **(C)** soit satisfaite (condition sur x_1). C'est-à-dire que la connaissance d'une solution $\boldsymbol{x}'$ du système **(S')** nous fait remonter à la condition sur x_1 donnée par la relation **(C)** qui dépend uniquement de $\boldsymbol{x}'$. On peut donc déterminer la borne inférieure *I* et la borne supérieure *S* de la variable x_1. En choisissant une valeur pour x_1, on trouve donc une solution $\mathbf{x} = [\boldsymbol{x_1}\ \boldsymbol{x}']^T$ du système **(S)**.

Exemple1 **[4]**

$$(a)\begin{cases} x_1 - 2x_2 \le -2 \\ x_1 + x_2 \le 3 \\ x_1 + 0x_2 \le 2 \\ -2x_1 + x_2 \le 0 \\ -x_1 + 0x_2 \le -1 \\ 0x_1 + 8x_2 \le b \end{cases}$$

Appliquons l'algorithme de Fourier – Motzkin :

a) Phase de descente

$$(a) \Leftrightarrow \begin{cases} x_1 - 2x_2 \le -2 \\ x_1 + x_2 \le 3 \\ x_1 + 0x_2 \le 2 \\ -x_1 + \frac{1}{2}x_2 \le 0 \\ -x_1 + 0x_2 \le -1 \\ 0x_1 + 8x_2 \le b \end{cases} \Leftrightarrow \begin{cases} x_1 \le -2 + 2x_2 \\ x_1 \le 3 - x_2 \\ x_1 \le 2 \\ x_1 \ge \frac{1}{2}x_2 \\ x_1 \ge 1 \\ 8x_2 \le b \end{cases} \Leftrightarrow$$

$$\begin{cases} Max\left(\frac{1}{2}x_2, 1\right) \le x_1 \le Min(-2 + 2x_2, 3 - x_2, 2) \\ 8x_2 \le b \end{cases}$$

La condition sur x_1 est :

$$\left\{ Max\left(\frac{1}{2}x_2, 1\right) \le x_1 \le Min(-2 + 2x_2, 3 - x_2, 2) \right.$$

Le nouveau système (b) après élimination de la variable x_1 est :

$$(b)\begin{cases} \frac{1}{2}x_2 \le -2 + 2x_2 \\ \frac{1}{2}x_2 \le 3 - x_2 \\ \frac{1}{2}x_2 \le 2 \\ 1 \le -2 + 2x_2 \\ 1 \le 3 - x_2 \\ 1 \le 2 \\ 8x_2 \le b \end{cases} \Leftrightarrow \begin{cases} x_2 \le -4 + 4x_2 \\ x_2 \le 6 - 2x_2 \\ x_2 \le 4 \\ 1 \le -2 + 2x_2 \\ 1 \le 3 - x_2 \\ 1 \le 2 \\ x_2 \le \frac{1}{8}b \end{cases} \Leftrightarrow \begin{cases} \frac{4}{3} \le x_2 \\ x_2 \le 2 \\ x_2 \le 4 \\ \frac{3}{2} \le x_2 \\ x_2 \le 2 \\ 1 \le 2 \\ x_2 \le \frac{1}{8}b \end{cases}$$

La condition sur la variable x_2 donne :

$$Max\left(\frac{4}{3},\frac{3}{2}\right) \le x_2 \le Min\left(2,4,2,\frac{1}{8}b\right) \quad \Leftrightarrow \quad \frac{3}{2} \le x_2 \le Min\left(2,\frac{1}{8}b\right)$$

Cas1 En prenant ***b* = 15**, on a :

$$\frac{3}{2} \le x_2 \le Min\left(2,\frac{15}{8}\right) \quad \Leftrightarrow \quad \frac{3}{2} \le x_2 \le \frac{15}{8} \text{ : } cohérent$$

Le système (a) admet donc une solution réelle dans IR (**test d'existence réussit**).

b) Phase de remontée

On peut prendre par exemple $x_2 = 3/2$ et la condition sur x_1 donne :

$$Max\left(\frac{3}{4},1\right) \le x_1 \le Min\left(1,\frac{3}{2},2\right) \Leftrightarrow 1 \le x_1 \le 1 \Leftrightarrow x_1 = 1$$

Une solution quelconque du système (a) est donc : $x = \begin{bmatrix} \frac{3}{2} & 1 \end{bmatrix}^T$

Cas2 En prenant ***b* = 11**, la condition sur x_2 donne :

$\frac{3}{2} \le x_2 \le Min\left(2,\frac{11}{8}\right) \Leftrightarrow \frac{3}{2} \le x_2 \le \frac{11}{8}$ ce qui est impossible et il n'y a donc pas de solution réelle pour le système (a).

2.3 Complexité de l'algorithme de Fourier-Motzkin [5]

Le nombre d'inéquations du nouveau système est : $|I_-| * |I_+| + |I_0|$

Si $|I_+|=|I_-|=m/2$ alors le nombre d'inéquations du nouveau système est : $m^2/4+|I_0|$

La complexité dans le pire des cas est : $(n+1)\left(\frac{|I|}{2}\right)^{2^n}$ avec

n : le nombre de variables
I : le système initial.
Il y a donc un risque de croissance exponentielle.
Remarque
Le nombre d'inéquations n'augmente pas si $|I_+|=1$ ou $|I_-|=1$ ou $|I_+|=|I_-|=2$

L'algorithme de Fourier – Motzkin est ainsi est non polynomial et est donc consommateur de temps.

2.4 Possibilités de l'algorithme de Fourier - Motzkin

L'algorithme de Fourier-Motzkin peut être utilisé pour résoudre différents problèmes. En effet, des transformations sont possibles pour ramener ces problèmes sous la forme :
$\boldsymbol{Ax \leq b}$**.** Nous illustrons maintenant cette affirmation.

a) Résolution d'équations $Ax = b$ dans IR

Résoudre $Ax = b$ dans IR revient à résoudre le système suivant :

$$\begin{cases} Ax \leq b \\ Ax \geq b \end{cases} \Leftrightarrow \begin{cases} Ax \leq b \\ Ax \geq b \end{cases} \Leftrightarrow \begin{bmatrix} +A \\ -A \end{bmatrix} x \leq \begin{bmatrix} +b \\ -b \end{bmatrix}$$

b) Résolution d'équations $Ax = b$ dans IR^+

Résoudre $Ax = b$ dans IR^+ revient à résoudre le système suivant :

$$\begin{cases} \begin{bmatrix} +A \\ -A \end{bmatrix} x \leq \begin{bmatrix} +b \\ -b \end{bmatrix} \\ \mathrm{I}x \geq b' \end{cases} \Leftrightarrow \begin{bmatrix} +A \\ -A \\ -\mathrm{I} \end{bmatrix} x \leq \begin{bmatrix} +b \\ -b \\ -b' \end{bmatrix}$$

Avec A de dimension mxn, b de dimension nx1, I de dimension nxn, b' vecteur nul de dimension nx1.

c) Résolution d'équations $Ax = b$ dans IR^-

$$\begin{cases} \begin{bmatrix} +A \\ -A \end{bmatrix} x \leq \begin{bmatrix} +b \\ -b \end{bmatrix} \\ \mathrm{I}x \leq b' \end{cases} \Leftrightarrow \begin{bmatrix} +A \\ -A \\ +\mathrm{I} \end{bmatrix} x \leq \begin{bmatrix} +b \\ -b \\ +b' \end{bmatrix}$$

Avec A de dimension mxn, b de dimension nx1, I de dimension nxn, b' vecteur nul de dimension nx1.

d) Maximisation de $Z = Cx$ sous $Ax \leq b$

A : matrice de dimension mxn
C : vecteur de dimension 1xn
b : vecteur de dimension nx1
On introduit une nouvelle variable λ qui devra être aussi grande que possible.
Comme on prend $\lambda - Cx \leq 0$, Cx sera maximisé. Il faudra appliquer la méthode d'élimination de Fourier-Motzkin à

$$\begin{bmatrix} A & a \\ -C & 1 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \leq \begin{bmatrix} b \\ 0 \end{bmatrix}, \; a \text{ : vecteur nul de dimension } m\text{x}1$$

de manière que λ soit la dernière variable du système que l'on pourra choisit aussi grande que possible.
Ainsi donc l'algorithme de Fourier-Motzkin peut traiter les mêmes problèmes que l'algorithme de Simplexe.

e) Minimisation de Z = *Cx* sous *Ax* $\leq$ *b*

A : matrice de dimension *m*x*n*
C : vecteur de dimension 1x*n*
b : vecteur de dimension *n*x1

Il suffira de maximiser Z = -*Cx* sous $Ax \leq b$
Comme précédemment, On introduit une nouvelle variable λ qui devra être aussi grande que possible. Comme on prend $\boldsymbol{\lambda + Cx \leq 0}$, *Cx* sera maximisé. Il faudra appliquer la méthode d'élimination de Fourier-Motzkin à

$$\begin{bmatrix} A & a \\ C & 1 \end{bmatrix}\begin{bmatrix} x \\ \lambda \end{bmatrix} \leq \begin{bmatrix} b \\ 0 \end{bmatrix}$$, a : vecteur nul de dimension *m*x1

de manière que λ soit la dernière variable du système que l'on pourra choisit aussi grande que possible.
Ainsi donc l'algorithme de Fourier-Motzkin sait bien résoudre un problème de minimisation.

Exemple2

$$\text{(a)} \begin{cases} Maximisation\ Z = x_1 + 2x_2\ ,\ sous \\ x_1 + x_2 \leq 2 \\ 0x_1 + x_2 \leq 1 \\ -x_1 + 0x_2 \leq 0 \\ 0x_1 - x_2 \leq 0 \end{cases}$$, en introduisant la nouvelle variable λ (a) devient :

$$\begin{cases} x_1 + x_2 + 0\lambda \leq 2 \\ 0x_1 + x_2 + 0\lambda \leq 1 \\ -x_1 + 0x_2 + 0\lambda \leq 0 \\ 0x_1 - x_2 + 0\lambda \leq 0 \\ -x_1 - 2x_2 + \lambda \leq 0 \end{cases} \Leftrightarrow \begin{cases} x_1 \leq 2 - x_2 \\ x_2 \leq 1 \\ 0 \leq x_1 \\ 0 \leq x_2 \\ \lambda - 2x_2 \leq x_1 \end{cases} \Leftrightarrow Max(0, \lambda - 2x_2) \leq x_1 \leq Min(2 - x_2)$$

Le nouveau système après élimination de la variable x_1 donne est :

$$\begin{cases} 0 \le 2 - x_2 \\ \lambda - 2x_2 \le 2 - x_2 \\ x_2 \le 1 \\ 0 \le x_2 \end{cases} \Leftrightarrow \begin{cases} x_2 \le 2 \\ \lambda - 2 \le x_2 \\ x_2 \le 1 \\ 0 \le x_2 \end{cases} \Leftrightarrow Max(0, \lambda - 2) \le x_2 \le Min(1, 2) \Leftrightarrow Max(0, \lambda - 2) \le x_2 \le 1$$

Le nouveau système après élimination de la variable x_2 est : $\begin{cases} 0 \le 1 \text{ cohérent} \\ \lambda - 2 \le 1 \end{cases} \Leftrightarrow \lambda \le 3$

Comme on choisit la plus grande valeur de λ pour maximiser Z, on prend donc **λ = 3** d'où

$Max(0, 3-2) \le x_2 \le 1 \Leftrightarrow Max(0,1) \le x_2 \le 1 \Leftrightarrow 1 \le x_2 \le 1 \Leftrightarrow$ ***x₂* = 1.**

$Max(0, 3-2*1) \le x_1 \le Min(2-1) \Leftrightarrow Max(0,1) \le x_1 \le Min(1) \Leftrightarrow$ ***x₁* = 1.** D'où la solution optimale $x_{opt} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, et $Z_{\max} = 3$.

2.5 Mise en place informatique

Soit une matrice *A* à *m* lignes, et *n* colonnes, et un vecteur *b* de taille *m*, tel que $\boldsymbol{Ax} \le \boldsymbol{b}$.

- On considère la première colonne de *A* et donc à la première variable x_1. L'ensemble des autres variables est noté $\boldsymbol{x}' = [\boldsymbol{x_2\ x_3\ ..\ x_n}]^T$.
  On peut multiplier chaque ligne de système d'inéquations par un scalaire tel que le coefficient de la première colonne soit 1 ou -1 s'il est différent de 0. D'où la forme **(S)** après classement des m lignes est :

$$(S) \begin{cases} a^1_{pos} x_1 & + A^1_{pos} x' & \le & b^1_{pos} \\ -a^1_{neg} x_1 & + A^1_{neg} x' & \le & b^1_{neg} \\ a^1_{zer} x_1 & + A^1_{zer} x' & \le & b^1_{zer} \end{cases}$$

2.5.1 Analyse fine de l'algorithme de Fourier-Motzkin

L'algorithme de Fourier-Motzkin peut rencontrer différentes situations possibles que nous allons décrire maintenant.

D'une manière générale, pour l'élimination de la variable x_l on a :

$$\textbf{(S)} \begin{cases} a^l_{pos} x_l & + A^l_{pos} x' & \le & b^l_{pos}\ ,\ I_+ \\ -a^l_{neg} x_1 & + A^l_{neg} x' & \le & b^l_{neg}\ ,\ I_- \\ a^l_{zer} x_1 & + A^l_{zer} x' & \le & b^l_{zer}\ ,\ I_0 \end{cases}$$

Notations :

I_+ : correspond au sous système d'inéquations décrit par la première ligne de (S)

I_- : correspond au sous système d'inéquations décrit par la deuxième ligne de (S)

I_0 : correspond au sous système d'inéquations décrit par la troisième ligne de (S)

$npos$: le nombre d'inéquations représentées par le système I_+

$nneg$: le nombre d'inéquations représentées par le système I_-

$nzer$: le nombre d'inéquations représentées par le système I_0

A^l_{pos} : matrice de dimension $npos$ x $(n-l)$ A^l_{neg} : matrice de dimension $nneg$ x $(n-l)$

a^l_{pos} : vecteur unitaire de dimension $npos$ x 1 , a^l_{neg} : vecteur unitaire de dimension $nneg$ x 1

b^l_{pos} : vecteur de dimension $npos$ x 1 b^l_{neg} : vecteur de dimension $nneg$ x 1

A^l_{zer} : matrice de dimension $nzer$ x $(n-l)$

a^l_{zer} : vecteur nul de dimension $nzer$ x 1

b^l_{zer} : vecteur de dimension $nzer$ x 1

$A^l_{pos,i}$, $A^l_{neg,i}$, $A^l_{zer,i}$ respectivement la $i^{ème}$ ligne des matrices A^l_{pos} , A^l_{neg} , A^l_{zer}

$b^l_{pos,i}$, $b^l_{neg,i}$, $b^l_{zer,i}$ respectivement le $i^{ème}$ élément des vecteurs $b^l_{pos}, b^l_{neg}, b^l_{zer}$

Le système **(S)** précédent représente la forme générale présentée par l'algorithme de Fourier-Motzkin. Pour cette forme générale **(S),** lors de l'élimination de x_l, nous avons la condition (C) suivante :

(C) $$I = Max\left(A^l_{neg}x^{'} - b^l_{neg}\right) \le x_l \le S = Min\left(b^l_{pos} - A^l_{pos}x^{'}\right)$$

Le nouveau système généré après l'élimination de la variable x_l est :

(S') $$\begin{cases} \left(A^l_{neg,j} + A^l_{pos,i}\right)x^{'} \le \left(b^l_{neg,j} + b^l_{pos,i}\right) \; ; \; (i = 1 \text{ à } npos \; ; \; j = 1 \text{ à } nneg) \\ A^l_{zer}x^{'} \le b^l_{zer} \end{cases}$$

Avec $\boldsymbol{x^{'} = [x_{l+1}\, x_{l+2} \ldots x_n]^T}$

En réalité, on distingue six autres formes possibles en dehors de la forme générale **(S)** :

- **Forme1 ou système (S1)** :

On considère que les éléments d'indice *zer* dans la forme générale **(S)** sont nuls, c'est à dire que I_0 n'existe pas dans **(S)** d'où **(S)** devient :

(S1) $$\begin{cases} a^l_{pos}x_l + A^l_{pos}x^{'} \le b^l_{pos} \; , I_+ \\ -a^l_{neg}x_l \quad A^l_{neg}x^{'} \le b^l_{neg} \; , \; I_- \end{cases}$$

La condition **(C)** de la forme générale lors de l'élimination de la variable x_l ne change pas car elle ne dépend pas de a^l_{zer}, A^l_{zer} et b^l_{zer} mais nous la notons **(C1)** pour montrer qu'il s'agit de la forme1.

(C1) $I = Max\left(A^l_{neg}x^{'} - b^l_{neg}\right) \le x_l \le S = Min\left(b^l_{pos} - A^l_{pos}x^{'}\right)$

En revanche le nouveau système **(S')** de la forme générale change car la seconde ligne a disparu. On la note **(S'1).**

(S'1) $\left(A^l_{neg,j} + A^l_{pos,i}\right)x^{'} \le \left(b^l_{neg,j} + b^l_{pos,i}\right), \left(i = 1 \text{ à } npos\,;\; j = 1 \text{ à } nneg\right)$

- **Forme2 ou système (S2) :**
  On considère que les éléments d'indices *pos et neg* dans la forme générale **(S)** sont nuls, c'est à dire que I_+ et I_- n'existent pas dans **(S)** d'où **(S)** devient **(S2)** :

  (S2) $a^l_{zer}x_l + A^l_{zer}x^{'} \le b^l_{zer}\ ,\ I_0$

  a^l_{zer} étant un vecteur nul, tous les coefficients de la variable x_l dans **(S2)** sont donc nuls et par conséquent n'importe quelle valeur de x_l dans IR vérifie **(S2)**. La condition sur x_l donne :

  (C2) $x_l \in \text{IR}$

  Le nouveau système **(S'2)** lors de l'élimination de la variable x_l est :

  (S'2) $A^l_{zer}x^{'} \le b^l_{zer}$

- **Forme3 ou système (S3)**
  On considère que les éléments d'indice *neg* dans la forme générale **(S)** sont nuls c'est à dire I_- n'existe pas dans **(S)** d'où **(S)** devient **(S3) :**

  (S3) $\begin{cases} a^l_{pos}x_l + A^l_{pos}x^{'} \le b^l_{pos}\ , I_+ \\ a^l_{zer}x_l + A^l_{zer}x^{'} \le b^l_{zer}\ , I_0 \end{cases}$

  Dans **(S3)**, en considérant la première ligne, nous déduisons la condition **(C3)** sur la variable x_l :

  (C3) $I = -\infty \le x_l \le S = Min\left(b^l_{pos} - A^l_{pos}x^{'}\right)$

  Le nouveau système **(S'3)** lors de l'élimination de la variable x_l est :

  (S'3) $A^l_{zer}x^{'} \le b^l_{zer}$

- **Forme4 ou système (S4)**
  On considère que les éléments d'indice *pos* dans la forme générale **(S)** sont nuls c'est à dire I_+ n'existe pas dans **(S)** d'où **(S)** devient **(S4) :**

(S4) $\begin{cases} -a^l_{neg}x_l + A^l_{neg}x' \leq b^l_{neg}, I_- \\ a^l_{zer}x_l + A^l_{zer}x' \leq b^l_{zer}, I_0 \end{cases}$

Dans **(S4)**, en considérant la première ligne, nous déduisons la condition **(C4)** sur la variable x_l :

(C4) $I = Max\left(A^l_{neg}x' - b^l_{neg}\right) \leq x_l \leq S = +\infty$

Le nouveau système **(S'4)** lors de l'élimination de la variable x_l est :

(S'4) $A^l_{zer}x' \leq b^l_{zer}$

- **Forme5 ou système (S5)**

  On considère que les éléments d'indices *zer* et *neg* dans la forme générale **(S)** sont nuls, c'est-à-dire que I_0 et I_- n'existent pas dans **(S)** d'où **(S)** devient :

  (S5) $a^l_{pos}x_l + A^l_{pos}x' \leq b^l_{pos}, I_+$

  La condition **(C5)** sur la variable x_l lors de l'élimination est :

  (C5) $I = -\infty \leq x_l \leq S = Min\left(b^l_{pos} - A^l_{pos}x'\right)$

  Dans ce cas, pas de nouveau système et la procédure d'élimination s'arrête. L'ensemble des autres variables c'est-à-dire le vecteur x' est quelconque.

- **Forme6 ou système (S6)**

  On considère que les éléments d'indices *zer* et *pos* dans la forme générale **(S)** sont nuls, c'est-à-dire que I_0 et I_+ n'existent pas dans **(S)** d'où **(S)** devient :

  (S6) $-a^l_{neg}x_l + A^l_{neg}x' \leq b^l_{neg}$

  La condition **(C6)** sur la variable x_l lors de l'élimination est :

  (C6) $Max\left(A^l_{neg}x' - b^l_{neg}\right) \leq x_l$

  Dans ce cas, pas de nouveau système et la procédure d'élimination s'arrête. L'ensemble des autres variables c'est-à-dire le vecteur x' est quelconque.

2.5.2 Test d'existence de solution

On s'intéresse à la fin de la phase de descente, c'est à dire la détermination de la condition sur la dernière variable x_n :

Lors de la détermination de la condition sur la dernière variable x_n, on peut tomber sur un système ayant la forme générale (S), ou la forme (S1), (S2), (S3), (S4), (S5) ou (S6). Mais cette fois l'ensemble des autres variables formé par le vecteur x' sera considéré comme étant nul. C'est-à-dire qu'il n'existe plus de variable au-delà de x_n.

Nous allons étudier le test d'existence de solution selon chaque forme :

- **Forme générale (S)**

$$(S)\begin{cases} a^n_{pos}x_n & \le & b^n_{pos} \\ -a^n_{neg}x_n & \le & b^n_{neg} \\ a^n_{zer}x_n & \le & b^n_{zer} \end{cases} \Leftrightarrow \begin{cases} I = Max\left(-b^n_{neg}\right) \le x_n \le S = Min\left(b^n_{pos}\right) \\ a^n_{zer}x_n \le b^n_{zer} \end{cases}$$

Le système d'inéquations linéaires $\boldsymbol{Ax \le b}$ admet une solution réelle si et seulement si

- $S \ge I$ et
- Tous les éléments du vecteur b^n_{zer} sont positifs. C'est à dire qu'on se trouve dans un cas où on a $0.x_n \le b^n_{zer,i}$, $(i = 1 \text{ à } nzer)$ qui est une solution triviale. Elle sera vérifiée seulement pour $b^n_{zer,i} \ge 0$, $(i = 1 \text{ à } nzer)$.

- **Forme1 (S1)**

$$(S1)\begin{cases} a^n_{pos}x_n \le b^n_{pos} \ , \ I_+ \\ -a^n_{neg}x_n \le b^n_{neg} \ , I_- \end{cases} \Leftrightarrow \ I = Max\left(-b^n_{neg}\right) \le x_n \le S = Min\left(b^n_{pos}\right)$$

Le système d'inéquations linéaires $\boldsymbol{Ax \le b}$ admet une solution réelle si et seulement si

- $S \ge I$

- **Forme2 (S2)**

(S2) : $a^n_{zer}x_n \le b^n_{zer}$ Le système d'inéquations linéaires $\boldsymbol{Ax \le b}$ admet une solution réelle si et seulement si :

- Tous les éléments de b^n_{zer} sont positifs. C'est à dire qu'on se trouve dans un cas où on a $0.x_n \le b^n_{zer,i}$, $(i = 1 \text{ à } nzer)$ qui est une solution triviale. Elle sera vérifiée seulement pour $b^n_{zer,i} \ge 0$, $(i = 1 \text{ à } nzer)$.

- **Forme3 (S3)**

$$(S3)\begin{cases} a^n_{pos}x_n \le b^n_{pos} \ , \ I_+ \\ a^n_{zer}x_n \le b^n_{zer} \ , I_0 \end{cases} \Leftrightarrow \begin{cases} I = -\infty \le x_n \le Min\left(b^n_{pos}\right) \\ a^n_{zer}x_n \le b^n_{zer} \ , I_0 \end{cases}$$

Le système d'inéquations linéaires $\boldsymbol{Ax \le b}$ admet une solution réelle si et seulement si :

- Tous les éléments de b^n_{zer} sont positifs. C'est à dire qu'on se trouve dans un cas où on a $0.x_n \le b^n_{zer,i}$, $(i = 1 \text{ à } nzer)$ qui est une solution triviale. Elle sera vérifiée seulement pour $b^n_{zer,i} \ge 0$, $(i = 1 \text{ à } nzer)$.

- **Forme4 (S4)**

$$(S4)\begin{cases} -a^n_{neg}x_n \le b^n_{neg} \ , \ I_- \\ a^n_{zer}x_n \le b^n_{zer} \ , I_0 \end{cases} \Leftrightarrow \begin{cases} I = Max\left(-b^n_{neg}\right) \le x_n \le S = +\infty \\ a^n_{zer}x_n \le b^n_{zer} \ , I_0 \end{cases}$$

Le système d'inéquations linéaires $\boldsymbol{Ax \le b}$ admet une solution réelle si et seulement si :

- Tous les éléments de b^n_{zer} sont positifs. C'est à dire qu'on se trouve dans un cas où on a $0.x_n \leq b^n_{zer,i}$, $(i = 1 \text{ à } nzer)$ qui est une solution triviale. Elle sera vérifiée seulement pour $b^n_{zer,i} \geq 0,\ (i = 1 \text{ à } nzer)$.

- **Forme5 (S5)**
  (S5) $a^n_{pos} x_n \leq b^n_{pos} \Leftrightarrow I = -\infty \leq x_n \leq S = Min(b^n_{pos})$
  Le système d'inéquations linéaires $\boldsymbol{Ax} \leq \boldsymbol{b}$ admet une solution car les bornes inférieures *I* et supérieures *S* définissent un intervalle quelque soit la valeur de *S.*

- **Forme6 (S6)**
  (S6) $-a^n_{neg} x_n \leq b^n_{neg} \Leftrightarrow I = Max(-b^n_{neg}) \leq x_n \leq S = +\infty$
  Le système d'inéquations linéaires $\boldsymbol{Ax} \leq \boldsymbol{b}$ admet une solution car les bornes inférieures *I* et supérieures *S* définissent un intervalle quelque soit la valeur de *I.*

2.5.3 Synthèse

<u>*Rappels*</u> :

La forme générale est notée (S)

$$(S)\begin{cases} a^l_{pos} x_l + A^l_{pos} x' \leq b^l_{pos}\ ,\ I_+ \\ -a^l_{neg} x_1 + A^l_{neg} x' \leq b^l_{neg}\ ,\ I_- \\ a^l_{zer} x_1 + A^l_{zer} x' \leq b^l_{zer}\ ,\ I_0 \end{cases}$$

I_+ : correspond au sous système d'inéquations d'écrit par la première ligne de (S)
I_- : correspond au sous système d'inéquations d'écrit par la deuxième ligne de (S)
I_0 : correspond au sous système d'inéquations d'écrit par la troisième ligne de (S)
npos : le nombre d'inéquations formé par le système I_+
nneg : le nombre d'inéquations formé par le système I_-
nzer : le nombre d'inéquations formé par le système I_0

A^l_{pos} : matrice de dimension *npos* x $(n-l)$ $\quad$ A^l_{neg} : matrice de dimension *nneg* x $(n-l)$
a^l_{pos} : vecteur unitaire de dimension *npos* x 1 , a^l_{neg} : vecteur unitaire de dimension *nneg* x 1
b^l_{pos} : vecteur de dimension *npos* x 1 $\quad$ b^l_{neg} : vecteur de dimension *nneg* x 1
A^l_{zer} : matrice de dimension *nzer* x $(n-l)$
a^l_{zer} : vecteur nul de dimension *nzer* x 1
b^l_{zer} : vecteur de dimension *nzer* x 1

$A^l_{pos,i}$, $A^l_{neg,i}$, $A^l_{zer,i}$ respectivement la $i^{ème}$ ligne des matrices A^l_{pos} , A^l_{neg} , A^l_{zer}

$b^l_{pos,i}$, $b^l_{neg,i}$, $b^l_{zer,i}$ respectivement le $i^{ème}$ élément des vecteurs b^l_{pos} , b^l_{neg} , b^l_{zer}

Tableau de synthèse sur la phase de descente

forme	Système avec x_l	$I \le x_l \le S$	Système suivant avec la variable x_l éliminée	*npos*	*nneg*	*nzer*
(S)	$\begin{cases} I_+ \\ I_- \\ I_0 \end{cases}$	$I = Max\left(A^l_{neg}x^{'} - b^l_{neg}\right)$ $S = Min\left(b^l_{pos} - A^l_{pos}x^{'}\right)$	$\begin{cases} \left(A^l_{neg,j} + A^l_{pos,i}\right)x^{'} \le \left(b^l_{neg,j} + b^l_{pos,i}\right) \; ; (i = 1 \text{ à } npos;\; j = 1 \text{ à } nneg) \\ A^l_{zer}x^{'} \le b^l_{zer} \end{cases}$	$\neq 0$	$\neq 0$	$\neq 0$
1	$\begin{cases} I_+ \\ I_- \end{cases}$	$I = Max\left(A^l_{neg}x^{'} - b^l_{neg}\right)$ $S = Min\left(b^l_{pos} - A^l_{pos}x^{'}\right)$	$\left(A^l_{neg,j} + A^l_{pos,i}\right)x^{'} \le \left(b^l_{neg,j} + b^l_{pos,i}\right), (i = 1 \text{ à } npos\,;\, j = 1 \text{ à } nneg)$	$\neq 0$	$\neq 0$	$= 0$
2	I_0	$I = -\infty$ $S = +\infty$	$A^l_{zer}x^{'} \le b^l_{zer}$	$= 0$	$= 0$	$\neq 0$
3	$\begin{cases} I_+ \\ I_0 \end{cases}$	$I = -\infty$ $S = Min\left(b^l_{pos} - A^l_{pos}x^{'}\right)$	$A^l_{zer}x^{'} \le b^l_{zer}$	$\neq 0$	$= 0$	$\neq 0$
4	$\begin{cases} I_- \\ I_0 \end{cases}$	$I = Max\left(A^l_{neg}x^{'} - b^l_{neg}\right)$ $S = +\infty$	$A^l_{zer}x^{'} \le b^l_{zer}$	$= 0$	$\neq 0$	$\neq 0$
5	I_+	$I = -\infty$ $S = Min\left(b^l_{pos} - A^l_{pos}x^{'}\right)$	Pas de nouveau système (la phase de l'élimination s'arrête)	$\neq 0$	$= 0$	$= 0$
6	I_-	$I = Max\left(A^l_{neg}x^{'} - b^l_{neg}\right)$ $S = +\infty$	Pas de nouveau système (la phase de l'élimination s'arrête)	$= 0$	$\neq 0$	$= 0$

Si on trouve que les bornes de la variable x_n définissent un intervalle vide Alors pas de solution pour le système d'inéquations linéaires $\boldsymbol{Ax \le b}$. Autrement dit la borne inférieure I n'est pas inférieure à la borne supérieure S de la dernière variable x_n.
S'il existe des inégalités trivialement non vérifiées, alors il n'y a pas de solution.

Tableau de synthèse sur le test d'existence de solution

forme	Système avec x_n	$I \le x_l \le S$	Existence de solution
(S)	$\begin{cases} a^n_{pos}\, x_n \le b^n_{pos} \\ -a^n_{neg}\, x_n \le b^n_{neg} \\ a^n_{zer}\, x_n \le b^n_{zer} \end{cases}$	$I = Max\left(-b^n_{neg}\right)$ $S = Min\left(b^n_{pos}\right)$	si $S \ge I$ et $b^n_{zer,i} \ge 0$, $(i = 1 \text{ à } nzer)$ Alors il existe une solution
1	$\begin{cases} a^n_{pos}\, x_n \le b^n_{pos} \\ -a^n_{neg}\, x_n \le b^n_{neg} \end{cases}$	$I = Max\left(-b^n_{neg}\right)$ $S = Min\left(b^n_{pos}\right)$	si $S \ge I$ Alors il existe une solution
2	$a^n_{zer} x_n \le b^n_{zer}$	$I = -\infty$ $S = +\infty$	Il existe une solution
3	$\begin{cases} a^n_{pos}\, x_n \le b^n_{pos} \\ a^n_{zer}\, x_n \le b^n_{zer} \end{cases}$	$I = -\infty$ $S = Min\left(b^n_{pos}\right)$	si $b^n_{zer,i} \ge 0$, $(i = 1 \text{ à } nzer)$ Alors il existe une solution
4	$\begin{cases} -a^n_{neg}\, x_n \le b^n_{neg} \\ a^n_{zer}\, x_n \le b^n_{zer} \end{cases}$	$I = Max\left(-b^n_{neg}\right)$ $S = +\infty$	si $b^n_{zer,i} \ge 0$, $(i = 1 \text{ à } nzer)$ Alors il existe une solution
5	$a^n_{pos} x_n \le b^n_{pos}$	$I = -\infty$ $S = Min\left(b^n_{pos}\right)$	Il existe une solution
6	$-a^n_{neg} x_n \le b^n_{neg}$	$I = Max\left(-b^n_{neg}\right)$ $S = +\infty$	Il existe une solution

Exemple3

Considérons le système $\boldsymbol{Ax \le b}$ suivant :

$$(a)\begin{cases} -x_1 - 2x_2 - 2x_3 - x_4 \le -8 \\ x_1 - 2x_2 + x_3 + 6x_4 \le 3 \\ 2x_1 + 8x_2 + 5x_3 - 3x_4 \le 28 \end{cases} \quad \text{avec} \quad A = \begin{bmatrix} -1 & -2 & -2 & -1 \\ 1 & -2 & 1 & 6 \\ 2 & 8 & 5 & -3 \end{bmatrix} \text{ et } b = \begin{bmatrix} -8 \\ 3 \\ 28 \end{bmatrix}$$

Nous appliquons la méthode d'élimination des variables de Fourier-Motzkin :

$$(a) \Leftrightarrow \begin{cases} x_1 \ge 8 - 2x_2 - 2x_3 - x_4 \\ x_1 \le 3 + 2x_2 - x_3 - 6x_4 \\ x_1 \le 14 - 4x_2 - 2.5x_3 + 1.5x_4 \end{cases} , \text{ forme1 ou (S1)}$$

La condition (C1) sur la variable x_1 est :

(1) $\left(8 - 2x_2 - 2x_3 - x_4\right) \le x_1 \le Min\left(3 + 2x_2 - x_3 - 6x_4 , 14 - 4x_2 - 2.5x_3 + 1.5x_4\right)$

Le nouveau (S'1) système est :

(b) $\begin{cases} 8 & -2x_2 & -2x_3 & -x_4 & \leq & 3 & +2x_2 & -x_3 & -6x_4 \\ 8 & -2x_2 & -2x_3 & -x_4 & \leq & 14 & -4x_2 & -2.5x_3 & -1.5x_4 \end{cases} \Leftrightarrow$

(c) $\begin{cases} x_2 & \geq & 1.25 & -0.25x_3 & +1.25x_4 \\ x_2 & \leq & 3 & -0.25x_3 & +1.25x_4 \end{cases}$, forme1 ou (S1)

La condition (C1) sur la variable x_2 est :

(2) $(1.25 - 0.25x_3 + 1.25x_4) \leq x_2 \leq (3 - 0.25x_3 + 1.25x_4)$

Le nouveau système (S'1) est :

(d) $1.25 - 0.25x_3 + 1.25x_4 \leq 3 - 0.25x_3 + 1.25x_4 \Leftrightarrow$

(3) $0x_3 + 0x_4 \leq 1.75$, forme2 (S2) : solution triviale.

Cette dernière inégalité (3) trivialement vérifiée nous indique qu'il existe une solution au système. Pour en trouver une explicitement, on commence par poser par exemple $x_3 = x_4 = 0$, puis on remonte. On utilise (2), qui nous indique que $1.25 \leq x_2 \leq 3$. On choisit par exemple $x_2 = 2$, et pareil avec (1) pour choisir $x_1 = 4$. On a donc au final la solution $x = [4\ 2\ 0\ 0]^T$.

2.6 Guide de utilsateur

Nous avons développé un programme Scilab « Optim » basé sur l'algorithme de Fourier-Motzkin. Il traite toutes les parties développées dans ce chapitre. Il comprend trois parties : un sous programme « elimination » qui traite la phase de descente, un sous programme « remontee » qui traite la phase de remontée et une fonction d'affichage « affichage » qui oriente l'utilisateur dans ses choix.

- L'utilisateur lance le programme principale « Optim » qui lui donne la possibilité de tester l'existence de solution et d'en trouver une quand elle existe en tapant sur 1. En tapant sur 2 ou 3, l'utilisateur peut décide ainsi de traiter un problème d'optimisation. Tous les programmes sont en **annexe** de ce rapport.

2.7 Conclusion

La méthode d'élimination de Fourier-Motzkin est une méthode de résolution exacte dans IR des systèmes d'inéquations linéaires $\boldsymbol{Ax \leq b}$. Elle sert à la fois de procédure de décision (test d'existence de solution) et de technique de preuve (trouver une solution). Maintenant que nous disposons d'un programme qui sait résoudre les systèmes d'inégalités linéaires $\boldsymbol{Ax \leq b}$ dans IR, nous allons l'appliquer aux graphes d'Événements Temporisés pour le tracer des trajectoires temporelles. Une étape préalable est de modéliser les graphes d'Événements Temporisés sous la forme $\boldsymbol{Ax \leq b}$.

Chapitre 3

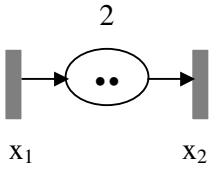
Application aux systèmes à événements discrets.

Application aux systèmes à événements discrets.

3.1 Modèle algébrique

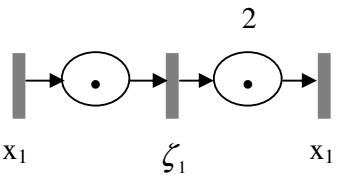
Dans cette partie, nous allons décrire les équations aux dateurs d'un graphe d'Événements Temporisés dans l'algèbre ordinaire sous forme : $\boldsymbol{Ax \leq b}$.
En général, pour les graphes d'Événements, certaines places contiennent plus d'un jeton.
Afin de rendre plus simple la manipulation de ses inéquations, nous allons dupliquer les places qui contiennent plus d'un jeton pour disposer enfin d'un graphe équivalent avec des places ayant au plus un jeton. Cela se fait au prix d'une extension du vecteur d'état *x(k)*. Pour obtenir cette forme récurrente, le graphe d'Événements Temporisés devra satisfaire :

- Toute place située entre deux transitions internes doit contenir au maximum un jeton,



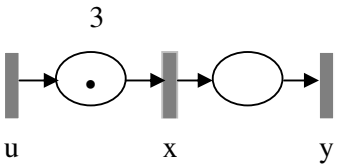
, Avec $x = [x_1\ x_2]^T$

Après transformation nous avons le graphe suivant :

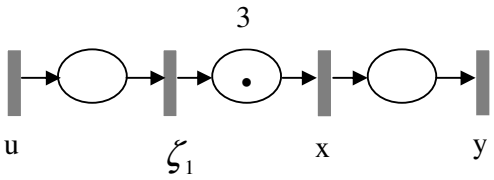


Avec $x = [x_1\ \zeta_1\ x_2]^T$

- Toute place située entre une transition source et une transition interne doit être sans jeton,



Après transformation nous avons le graphe suivant :



- Toute place située entre une transition interne et une transition puits doit être sans jeton.

3.1.1 Modélisation des graphes d'Événements sous la forme $Ax \leq b$

Nous allons partir d'un exemple de graphe d'Événements Temporisés pour déduire le modèle algébrique sous sa forme générale $\boldsymbol{Ax \leq b}$.

Exemple

On considère le graphe d'Événements Temporisés suivent :

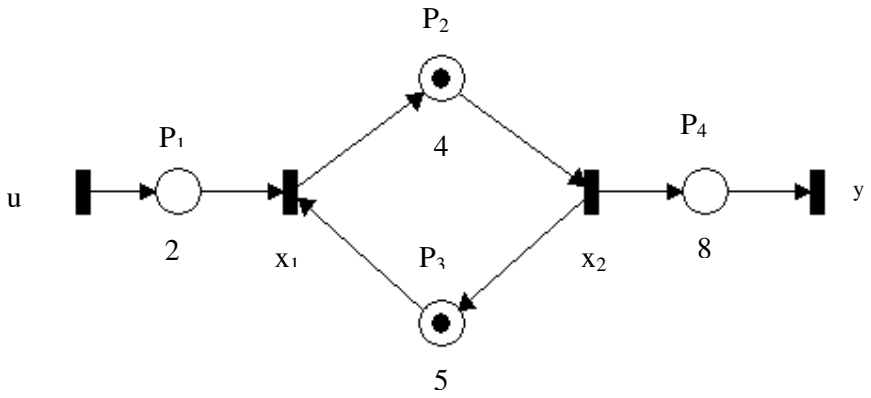


Figure1 : Graphe d'événements temporisés

Description aux dateurs :

$$\begin{cases} x_1(k) \geq 2+u(k) \\ x_1(k) \geq 5+x_2(k-1) \\ x_2(k) \geq 4+x_1(k-1) \\ y(k) \geq 8+x_2(k) \end{cases} \quad \textbf{(3.1)} \quad \Leftrightarrow$$

$$\begin{cases} \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} x(k) \geq \begin{bmatrix} 2 \\ 5 \\ 4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} x(k-1) + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u(k) \\ \\ y(k) \geq 8 + \begin{bmatrix} 0 & 1 \end{bmatrix} x(k) \end{cases} \Leftrightarrow$$

$$\begin{cases} \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} x(k) \geq \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} x(k-1) + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u(k) \begin{bmatrix} 2 \\ 5 \\ 4 \end{bmatrix} \\ \\ 1.y(k) \geq \begin{bmatrix} 0 & 1 \end{bmatrix} x(k) \quad 0.u(k) + 8 \end{cases} \quad \textbf{(3.2)}$$

En posant $A_x^{'} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}$; $A_x^{''} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$; $A_1 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$; $A_u = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$; $B_x = \begin{bmatrix} 2 \\ 5 \\ 4 \end{bmatrix}$

$$A_y = 1 \; ; \; C = \begin{bmatrix} 0 & 1 \end{bmatrix} ; \; D = 0 \; ; \quad B_y = 8$$

Le système **(3.2)** peut s'écrire :

$$\begin{cases} A_x^{'} x(k) \geq A_x^{''} x(k) + A_1 x(k-1) + A_u u(k) + B_x \\ A_y y(k) \geq Cx(k) + D.u(k) + B_y \end{cases} \quad \textbf{(3.3)}$$

D'où la forme générale :

$$\begin{cases} \left(A_x^{''} - A_x^{'}\right) x(k) \leq -A_1 x(k-1) - A_u u(k) - B_x \\ -A_y y(k) \leq -Cx(k) - D.u(k) - B_y \end{cases} \quad \textbf{(3.4)}$$

Avec :

$x(k)$: vecteur de dimension nx1 avec n le nombre de transitions internes

$A_x^{'}$, $A_x^{''}$, A_1 : matrices de dimension m x n avec m le nombre d'inéquations ne contenant pas les éléments du vecteur $y(k)$

$u(k)$: Vecteur d'entrée de dimension n_e x 1 avec n_e le nombre d'entrée du système

A_u : matrice de dimension m x n_e

B_x : vecteur de temporisation de dimension m x 1

$y(k)$: vecteur de sortie de dimension n_s x 1 avec n_s le nombre de sortie

D : matrice de dimension m' x n_e avec m' le nombre d'inéquations contenant les éléments de y (k)

A_y , C : matrices de dimension m' x n_s

B_y : vecteur de temporisation de dimension m' x1

En posant $\begin{cases} A = \left(A_x^{''} - A_x^{'}\right) \quad \text{et} \quad b = -A_1 x(k-1) - A_u x(k) - B_x \\ A^{'} = -A_y \quad \text{et} \quad b^{'} = -Cx(k) - D.u(k) - B_y \end{cases}$ le système **(3.4)** devient :

$$\begin{cases} Ax(k) \le b & \text{(a)} \\ A^{'} y(k) \le b^{'} & \text{(b)} \end{cases} \quad \textbf{(3.5)}$$

Connaissant *u(k),* on peut donc appliquer Fourier-Motzkin au sous système **(3.5.a)** pour déterminer *x(k)*, une fois *x(k)* connue, elle sera remplacée dans le sous système **(3.5.b)** pour calculer la sortie correspondante toujours par Fourier-Motzkin.

3.2 Systèmes monotones [6][7]

Définition1
Un système d'inéquations linéaires de la forme $\boldsymbol{Ax \le b}$ est dit **sup-monotone** (respectivement **inf-monotone**) si chaque ligne de la matrice $\boldsymbol{A}$ à au maximum un élément strictement positif (respectivement au maximum un élément négatif).

Exemple Considérons le système $\boldsymbol{Ax \le b}$ suivant :

$$A = \begin{bmatrix} -3 & -2 & 1 \\ 1 & -1 & -1 \\ 1 & 0 & -1 \end{bmatrix} \text{ et } b = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}$$

On remarque que sur chaque ligne de $\boldsymbol{A}$, on a au maximum un élément positif par ligne donc le système est sup-monotone donc monotone.

Définition2
Un système d'inéquations linéaires $\boldsymbol{Ax \le b}$ est dit monotone s'il est inf-monotone ou sup-monotone.

Définition3
Un système d'inéquations linéaires de la forme $\boldsymbol{Ax \le b}$ est dit **bimonotone** s'il est inf-monotone et sup-monotone.

Soient $\boldsymbol{Ax \le b}$ un système inf-monotone (respectivement sup-monotone) et E l'ensemble de ses solutions.

Théorème1

- L'ensemble E est un demi-treillis inférieur (respectivement demi-treillis supérieur)
- Si x et y sont deux éléments de E, le minimum entre x et y (respectivement le maximum) appartient à E.

Propriété1 : L'ensemble E d'un système bimonotone $\boldsymbol{Ax \le b}$ est un treillis.

Théorème 2 : L'ensemble E a un plus grand élément (respectivement plus petit élément) si l'ensemble E est non vide et à un majorant (respectivement un minorant).

Propriété remarquable

Un graphe d'Événements Temporisés modélisé sous la forme $\boldsymbol{Ax \leq b}$ (à travers sa description aux dateurs) est un système inf-monotone.

Démonstration

$$\begin{cases} Ax(k) \leq b & \text{(a)} \\ A^{'} y(k) \leq b^{'} & \text{(b)} \end{cases}$$ (voir pargraphe **3.1.1**)

Avec $A = A_x^{''} - A_x^{'}$, $A^{'} = -A_y$

D'une part

$A_x^{'}$ est une matrice dont chaque ligne possède un et un seul élément égal à 1 et 0 pour tous les autrs. Donc la matrice $-A_x^{'}$ possède sur chaque ligne un et un seul élément égal à -1 et 0 pour les aures.

D'autre part

Chaque ligne de la matrice $A_x^{''}$ possède des éléments positifs ou nuls. Donc $A = A_x^{''} - A_x^{'}$ possède au maximum un élément négatif par ligne d'où le système $\boldsymbol{Ax(k) \leq b}$ est inf-monotone.

Un raisonnement similaire pour le système $A^{'} y(k) \leq b^{'}$ nous montre également qu'il est inf-monotone du fait que $A^{'} = -A_y$ a au maximum un élément négatif par ligne.

3.3 Trajectoires monotones

Une trajectoire issue d'un graphe d'Événement Temporisés est nécessairement monotone croissante : la date d'occurrence *x(k)* est nécessairement supérieure à la date d'occurrence *x(k-1)*.

3.3.1 Application de l'algorithme de Fourier-Motzkin aux graphes d'événements Temporisés

Exemple

Considérons l'exemple du paragraphe **3.1.1**

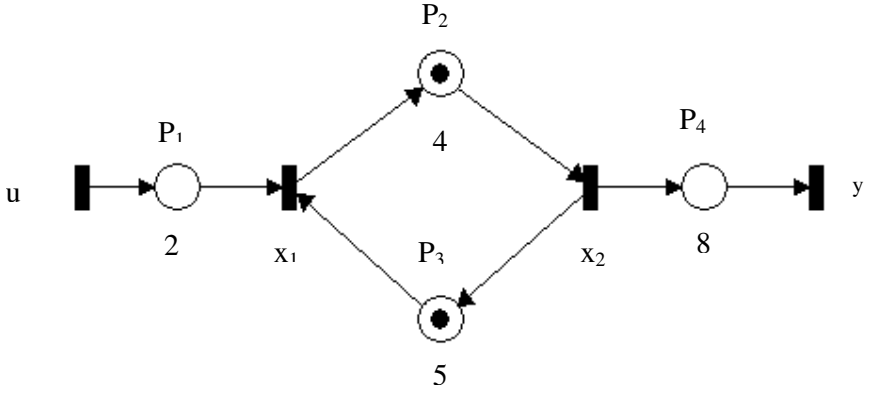


Figure1 : Graphe d'événements temporisés

Description aux dateurs :

$$\begin{cases} x_1(k) \geq 2+u(k) \\ x_1(k) \geq 5+x_2(k-1) \\ x_2(k) \geq 4+x_1(k-1) \\ y(k) \geq 8+x_2(k) \end{cases} \Leftrightarrow \begin{cases} -x_1(k) + 0.x_2(k) \leq -2-u(k) \\ -x_1(k) + 0.x_2(k) \leq -5-x_2(k-1) \\ 0.x_1(k) - x_2(k) \leq -4-x_1(k-1) \\ y(k) \geq 8+x_2(k) \end{cases}$$

Ce dernier système est équivalent à la **forme4** d'où nous avons les conditions suivantes :

$$\begin{cases} I = Max(2+u(k), 5+x_2(k-1)) \leq x_1(k) \leq S = +\infty \\ I = (4+x_1(k-1)) \leq x_2(k) \leq S = +\infty \\ I = (8+x_2(k)) \leq y(k) \leq S = +\infty \end{cases}$$

On va représenter la de sortie $y(k)$, $k = \{1,10\}$
Pour cela, on prend l'entrée $u(k)$ sur un horizon [1 10]

k	1	2	3	4	5	6	7	8	9	10
$u(k)$	28	28	67	76	76	115	124	124	172	$+\infty$

$$u(1) = 28 \ ; \ x(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{cases} I = Max(2+u(1), 5+x_2(0)) \leq x_1(1) \leq S = +\infty \\ I = (4+x_1(0)) \leq x_2(1) \leq S = +\infty \\ I = (8+x_2(1)) \leq y(1) \leq S = +\infty \end{cases} \Leftrightarrow \begin{cases} I = 30 \leq x_1(1) \leq S = +\infty \\ I = 4 \leq x_2(1) \leq S = +\infty \\ I = 12 \leq y(1) \leq S = +\infty \end{cases}$$

Une solution est :

$$x(1) = \begin{bmatrix} 30 \\ 4 \end{bmatrix} \ , \quad y(1) = 12$$

$$u(2) = 28 \ ; \ x(1) = \begin{bmatrix} 30 \\ 4 \end{bmatrix}$$

Une solution est :

$$\begin{cases} I = Max(2+u(2), 5+x_2(1)) \leq x_1(2) \leq S = +\infty \\ I = (4+x_1(1)) \leq x_2(2) \leq S = +\infty \\ I = (8+x_2(2)) \leq y(2) \leq S = +\infty \end{cases} \Leftrightarrow \begin{cases} I = 30 \leq x_1(2) \leq S = +\infty \\ I = 34 \leq x_2(2) \leq S = +\infty \\ I = 42 \leq y(2) \leq S = +\infty \end{cases}$$

$$x(2) = \begin{bmatrix} 30 \\ 34 \end{bmatrix} \ , \quad y(2) = 42$$

$$u(3) = 67 \;;\; x(2) = \begin{bmatrix} 30 \\ 34 \end{bmatrix}$$

$$\begin{cases} I = Max(2 + u(3), 5 + x_2(2)) \le x_1(3) \le S = +\infty \\ I = Max(4 + x_1(2)) \le x_2(3) \le S = +\infty \\ I = (8 + x_2(3)) \le y(3) \le S = +\infty \end{cases} \Leftrightarrow \begin{cases} I = 69 \le x_1(3) \le S = +\infty \\ I = 34 \le x_2(3) \le S = +\infty \\ I = 42 \le y(3) \le S = +\infty \end{cases}$$

Une solution est :

$$x(3) = \begin{bmatrix} 69 \\ 34 \end{bmatrix} \;,\quad y(3) = 42$$

En répétant la procédure, nous avons le résultat final :

k	1	2	3	4	5	6	7	8	9	10
$x_1(k)$	30	30	69	78	78	117	126	126	174	$+\infty$
$x_2(k)$	4	34	34	73	82	82	121	130	130	178

Une solution est :

k	1	2	3	4	5	6	7	8	9	10
$y(k)$	12	42	42	81	90	90	129	138	138	186

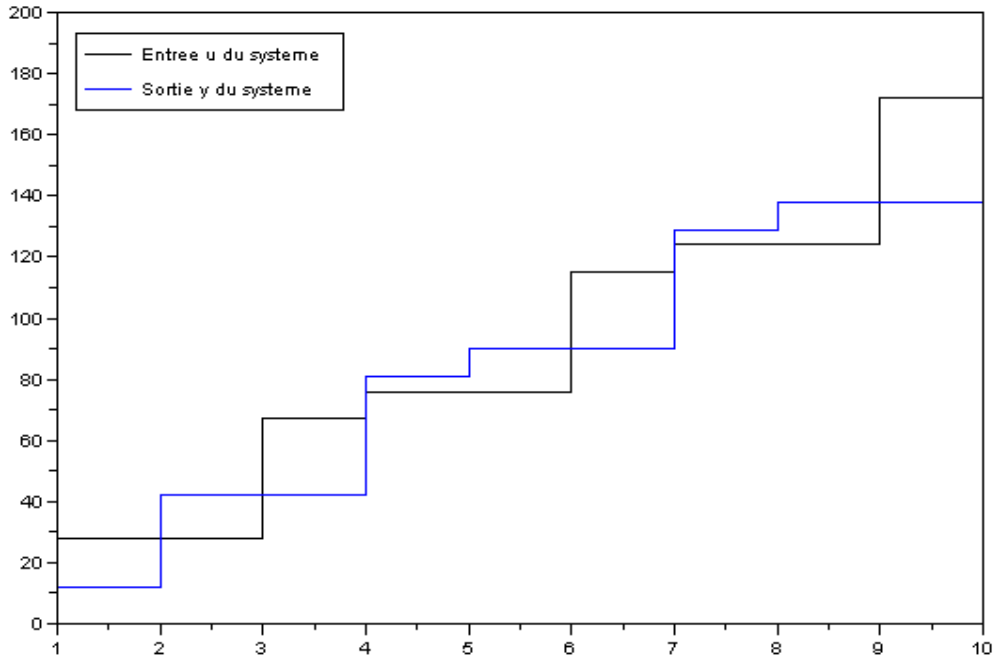


Figure2 : Trajectoire temporelle monotone

En abscisse de la figure2, nous avons le numéro de tire des transitions et en ordonnée les dates de tire. La sorte y représente la sortie du système qui est une trajectoire monotone et u l'entrée du système.

3.4 Conclusion

Dans ce chapitre, nous avons vu que les graphes d'Événements Temporisés peuvent être modélisés dans l'algèbre ordinaire sous la forme $\boldsymbol{Ax \leq b}$ au moyen de la description des inéquations aux dateurs. Nous avons également démontré que le modèle $\boldsymbol{Ax \leq b}$ représentatif des graphes d'Événements Temporisés est un système inf-monotone.

Nous avons développé également le programme Scilab « Optim » qui peut être utilisé pour tracer les trajectoires temporelles des systèmes à Événements Discrets.

Chapitre 4

Conclusion et perspectives

Dans ce rapport, nous avons pu mettre en place un programme développé sous Scilab qui réalise l'algorithme de Fourier-Motzkin. Ainsi à travers ce programme, nous disposons à la fois d'un outil d'aide à la décision et de résolution numérique pour les systèmes d'inéquations linéaires de la forme $\boldsymbol{Ax \leq b}$**.** Ce programme sait résoudre également les problèmes d'optimisation en programme linéaire comme l'algorithme de simplexe . Son temps de calcul est naturellement moins bon.

La modélisation des graphes d'Événements Temporisés sous la forme $\boldsymbol{Ax \leq b}$ nous permet d'exploiter le programme Scilab développé dans ce document pour calculer les réponses en sortie $y(k)$ des systèmes à Événements Discrets suite à une série d'entrée $u(k)$.

Les résultats trouvés dans ce rapport peuvent être étendus sur plusieurs points :

- Modélisation des graphes d'Événements Temporisés sous la forme algébrique $\boldsymbol{Ax \leq b}$ en utilisant la description aux compteurs.

- Calcul du taux de production pour les systèmes à événements discrets.

- Commande des systèmes à événements discrets

Bibliographie

[1] P. R. David, and H. Alla (1989). *Du grafcet aux réseaux de Petri. Hermès, Paris.*

[2] Proth.J.M, and XieX (1994). *Les réseaux de Petri pour la conception et la gestion des systems de production. Masson.*

[3] Chrijever. (1986). *Theorie of Linear and Integer Programming, Page155*

[4] P. Declerck. *Introduction à l'optimisation. Polycopié de cours donné à l'ISTIA*

[5] F. Thomasset (2001). *Calcul de dépendance dans les boucles, méthode d'élimination de Fourier-Motzkin*

[6] G. Nariboni (2001). *Un cas remarquable des systèmes linéaires : les systèmes monotones. Thèse Ecole Normale Supérieure de Cachan (2001).*

[7] G. Abdelhak, P. Declerck and J.L Boimond (2008). *From monotone inequalities to model predictive control.*

Annexe

Programme Principal « optim »

```
clear , clc
disp('##########################################################')
disp('##   METHODE D''ELIMINATION DE FOURIER-MOTZKIN :     ##')
disp('##   A.X < b  ou  Max(Min) Z=Cx sous A.X < b         ##')
disp('##   TAPEZ 1 POUR :                                  ##')
disp('##   - TROUVER UNE SOLUTION QUELCONQUE POUR A.X < b  ##')
disp('##   TAPEZ 2 POUR :                                  ##')
disp('##   - MAXIMISATION DE Z=C.X                         ##')
disp('##     SOUS A.X < b                                  ##')
disp('##   TAPEZ 3 POUR :                                  ##')
disp('##   - MINIMISATION DE Z=C.X                         ##')
disp('##     SOUS A.X < b                                  ##')
disp('##########################################################')

function temp=sortmat(A,b)
[m,n]=size(A);
temp=[A,b];
temp1=temp;
temp=[];
a1=temp1(:,1);
[s k]=sort(a1);
for i=1:m
   temp=[temp;temp1(k(i),:)];
end
temp;
endfunction

a_pos=list();
b_pos=list();
a_neg=list();
b_neg=list();
a_zer=list();
b_zer=list();
x=list();

mprintf('\n')
choix1=input(' TAPEZ VOTRE CHOIX ====> '); clc
mprintf('\n')
A=input(' DONNEZ LA MATRICE A =  ');clc
mprintf('\n')
b=input(' DONNER LE VECTEUR b =  ');
[m n]=size(A);clc
nvar=n;

if (choix1~=1) then
  mprintf('\n')
  C=input(' DONNER LE VECTEUR C / Z=CX  ');
  if (choix1==2) then
    Z=[-C 1];
  elseif (choix1==3) then
    Z=[C 1];
  end
  A(:,n+1)=zeros(m,1);
  A=[A;Z];
  [m n]=size(A);
  b(m)=0;
  nvar=n;
end
getf('F:\Fourier_Motzkin\affichage.sci');
exec('F:\Fourier_Motzkin\elimination.sce');
exec('F:\Fourier_Motzkin\remontee.sce');
```

Sous programme « elimination »

```
cas=zeros(6,nvar);
for i=1:nvar

  nvarLimite=i;
  temp=sortmat(A,b);

  A1=[]; b1=[];
  [m,n]=size(A);

  nneg=sum(temp(:,1)<0);
  nzer=sum(temp(:,1)==0);
  npos=sum(temp(:,1)>0);

  temp=[temp(1:npos,:);temp(npos+nzer+1:m,:);temp(npos+1:npos+nzer,:)];
  temp(1:npos+nneg,:)=temp(1:npos+nneg,:)./abs(temp(1:npos+nneg,1)*ones(1,n+1));

  a_pos(i)=temp(1:npos,2:n);
  b_pos(i)=temp(1:npos,n+1);

  a_neg(i)=temp(npos+1:npos+nneg,2:n);
  b_neg(i)=temp(npos+1:npos+nneg,n+1);

  a_zer(i)=temp(npos+nneg+1:m,2:n);
  b_zer(i)=temp(npos+nneg+1:m,n+1);
  A1=[]; b1=[];


  if (npos*nneg~=0) then
    for I1=1:npos  // (voir Sch86a : page 155 , (19))
      for J=1:nneg
        A1=[A1;a_pos(i)(I1,:)+a_neg(i)(J,:)];
        b1=[b1;b_pos(i)(I1)+b_neg(i)(J)];
      end
    end
    // nouveau système correspondant à l'étape i+1
    A=[A1;a_zer(i)];
    b=[b1;b_zer(i)];
    cas(1,i)=1;
  elseif (npos*nneg==0)&(nzer~=0) then
    A=a_zer(i); b=b_zer(i);
    if (npos==0)&(nneg==0) then
      cas(2,i)=1;
    elseif (npos~=0)&(nneg==0) then
      cas(3,i)=1;
    elseif (npos==0)&(nneg~=0) then
      cas(4,i)=1;
    end
  elseif (npos*nneg==0)&(nzer==0) then
    if (npos~=0) then
      cas(5,i)=1;
      break
    elseif (nneg~=0) then
      cas(6,i)=1;
      break
    end
  end
end
```

Sous programme « remontee »

```
R=1;
while R~=0
  X=[];

  if (nvarLimite~=nvar) then
    for i=nvarLimite+1:nvar
      clc

       mprintf('\n')
       disp('  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
       disp('  %                                        %')
       disp('  %      RESOLUTION PAR LA REMONTEE        %')
       disp('  %                                        %')
       disp('  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
       mprintf('\n')

      mprintf('\n')
      mprintf(' UN ENCADREMENT DE LA VARIABLE NUMERO : %.0f EST : [-Inf  Inf]\n',i);
      mprintf('\n')
      x(i)=input('DONNEZ UNE VALEUR POUR CETTE VARIABLE ====> ');clc
      X=[X;x(i)];
    end
  end



  for j=1:nvarLimite
    Numero=nvarLimite-j+1;
    b2=b_zer(nvarLimite-j+1)-a_zer(nvarLimite-j+1)*X;
    b3=sum(b2<0);
    I=max(a_neg(nvarLimite-j+1)*X-b_neg(nvarLimite-j+1));
    S=min(b_pos(nvarLimite-j+1)-a_pos(nvarLimite-j+1)*X);

       mprintf('\n')
       disp('  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
       disp('  %                                        %')
       disp('  %      RESOLUTION PAR LA REMONTEE        %')
       disp('  %                                        %')
       disp('  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
       mprintf('\n')

    if cas(1,Numero)==1 then // cas sympa

      if (b3~=0)|(S<I) then
        clc
        X=[]; break
      elseif (b3==0)&(S>=I) then

        affichage(1,I,S,Numero);
        x(Numero)=input('DONNEZ UNE VALEUR POUR CETTE VARIABLE ====> ');clc
        X=[x(Numero);X];
      end

    elseif cas(1,Numero)==0 then  // cas non sympa

      if (b3~=0) then
        clc,
        X=[]; break
      elseif (b3==0) then
        for i=2:6
          if cas(i,Numero)==1 then

           affichage(i,I,S,Numero);
           x(Numero)=input('DONNEZ UNE VALEUR POUR CETTE VARIABLE ====> ');clc
```

```
          X=[x(Numero);X];
         end
       end
     end
   end
  end

    mprintf('\n')
    disp('       #####    #####    #       ##   ##    ######    ##   ####    #   # ')
    disp('       #        #   #    #       ##   ##      ##      ##   #  #    ##  # ')
    disp('       #####    #   #    #       ##   ##      ##      ##   #  #    # # # ')
    disp('           #    #   #    #       ##   ##      ##      ##   #  #    #  ## ')
    disp('       #####    #####    #####   #######     ##      ##   ####    #   # ')
    mprintf('\n')
      X
   mprintf('\n')
   R=input(' POUR TROUVER UNE NOUVELLE SOLUTION TAPEZ 1 SINON O POUR SORTIR ====>  ');clc
end
```

Sous programme « affichage »

```
function affichage(i,I,S,Numero)
    if i==1 then
      mprintf('\n')
      mprintf('  UN ENCADREMENT DE LA VARIABLE NUMERO : %.0f EST :[%.5f %.5f]\n',Numero,I,S)
      mprintf('\n')

    elseif i==2 then
      mprintf('\n')
      mprintf(' UN ENCADREMENT DE LA VARIABLE NUMERO : %.0f EST : [-Inf  Inf]\n',Numero);
      mprintf('\n')

    elseif (i==3)|(i==5) then
      mprintf('\n')
      mprintf('  UN ENCADREMENT DE LA VARIABLE NUMERO : %.0f EST :[-Inf %.5f]\n',Numero,S)
      mprintf('\n')

    elseif (i==4)|(i==6) then
      mprintf('\n')
      mprintf('  UN ENCADREMENT DE LA VARIABLE NUMERO : %.0f EST :[%.5f Inf]\n',Numero,I)
      mprintf('\n')
    end
endfunction
```