Calculer des modules et des noyaux avec scicoslab

Formation sur les Systèmes à Événements Discrets (SED)

2e édition
Mars 2025
Nantes

$$A \otimes x = B \otimes x$$

$$x \in \operatorname{Im} M,$$

$$\operatorname{Im} M = \bigcap_{i=1\ldots n} \operatorname{Im} M_i,$$

$$A_i x = B_i x\ .$$

$$C \otimes x \leq D \otimes x$$

1. Cônes, modules, noyaux
   - L'algorithme de X. Allamigeon
   - Calculer une image, un moyau
2. Deux ou trois exercices d'application
   - Un problème de contraintes temporelles
   - Un problème de synchronisation
   - Un problème d'observation
3. Conclusion

L'algorithme de X. Allamigeon en quelques mots

```
function Gout=UmaLineaAxLeqBx(A,B,Gin)
//G=UmaLineaAxLeqBx(A,B,Gin)
//A, B, et Gin donnees, Gout est une matrice dont les colonnes sont des generateurs des solutions de Ax<=Bx.
//Si l'ensemble est vide, Gout est une matrice nulle (\varepsilon partout). Gin est
//l'ensemble des generateurs a tester.
//Cette procedure provient de l'algorithme de Xavier Allamigeon 'Double sided method' 2010.
//

[r,n]=size(Gin);
new=%F;
if (full(A) > full(B)) then Gout=%zeros(r,1);
elseif Gin==%zeros(r,1) then Gout=%zeros(r,1);
else
  p=0;q=0; G=%zeros(r,1); H=%zeros(r,1);GG=%zeros(r,1);GGG=%zeros(r,1);
  for i=1:n
     x=Gin(:,i);
     if (full(A*x)<=full(B*x)) then q=q+1; G(:,q)=x;
     else p=p+1;H(:,p)=x;
     end
  end;
  if (p>0) then
    k=0;
    for i=1:p;for j=1:q;
    k=k+1;
    GG(:,k)=(((A*H(:,i))*G(:,j))+((B*G(:,j))*H(:,i)));
    end;end;
    j=0;
    G2=G(:,1:q);
    for i=1:k;
      mb=-max(plustimes(GG(:,i)));b=(GG(:,i)*mb);
      if (~includespan(full(G2),full(b))) then
        j=j+1;
      GGG(:,j)=b;
      G2=[G(:,1:q) GGG];
      new=%T;
      end
    end
    end
    if (new) then Gout=[G(:,1:q) GGG];
    else Gout=G;
    end
end
endfunction
```

L'algorithme de X. Allamigeon en quelques mots

```
function Gout=UmaLineaAxLeqBx(A,B,Gin)
//G=UmaLineaAxLeqBx(A,B,Gin)
//A, B, et Gin donnees, Gout est une matrice dont les colonnes sont des generateurs des solutions de Ax<=Bx.
//Si l'ensemble est vide, Gout est une matrice nulle (\varepsilon partout). Gin est
//l'ensemble des generateurs a tester.
//Cette procedure provient de l'algorithme de Xavier Allamigeon 'Double sided method' 2010.
//

[r,n]=size(Gin);
new=%F;
if (full(A) > ful
elseif Gin==%zero
else
  p=0;q=0; G=%zer
  for i=1:n
     x=Gin(:,i);
     if (full(A*x)
     else p=p+1;H(:,p)=x;
     end
  end;
  if (p>0) then
    k=0;
    for i=1:p;for j=1:q;
    k=k+1;
    GG(:,k)=(((A*H(:,i))*G(:,j))+((B*G(:,j))*H(:,i)));
    end;end;
    j=0;
    G2=G(:,1:q);
    for i=1:k;
      mb=-max(plustimes(GG(:,i)));b=(GG(:,i)*mb);
      if (~includespan(full(G2),full(b))) then
        j=j+1;
      GGG(:,j)=b;
      G2=[G(:,1:q) GGG];
      new=%T;
      end
    end
    end
    if (new) then Gout=[G(:,1:q) GGG];
    else Gout=G;
    end
end
endfunction
```

1/ On trie les générateurs :

- dans G, ceux qui vérifient l'inégalité ;
- dans H, les autres.

L'algorithme de X. Allamigeon en quelques mots

```
function Gout=UmaLineaAxLeqBx(A,B,Gin)
//G=UmaLineaAxLeqBx(A,B,Gin)
//A, B, et Gin donnees. Gout est une matrice dont les colonnes sont des generateurs des solutions de Ax<=Bx.
//Si l'ensemble es
//l'ensemble des
//Cette procedure
//

[r,n]=size(Gin);
new=%F;
if (full(A) > ful
elseif Gin==%zero
else
  p=0;q=0; G=%zeros(r,1); H=%zeros(r,1);GG=%zeros(r,1);GGG=%zeros(r,1);
  for i=1:n
     x=Gin(:,i);
     if (full(A*x)<=full(B*x)) then q=q+1; G(:,q)=x;
     else p=p+1;H(
     end
  end;
  if (p>0) then
    k=0;
    for i=1:p;for
    k=k+1;
    GG(:,k)=(((A*
    end;end;
    j=0;
    G2=G(:,1:q);
    for i=1:k;
      mb=-max(plustimes(GG(:,i)));b=(GG(:,i)*mb);
      if (~includespan(full(G2),full(b))) then
        j=j+1;
      GGG(:,j)=b;
      G2=[G(:,1:q) GGG];
      new=%T;
      end
    end
    end
    if (new) then Gout=[G(:,1:q) GGG];
    else Gout=G;
    end
end
endfunction
```

1/ On trie les générateurs :

- dans G, ceux qui vérifient l'inégalité ;
- dans H, les autres.

2/ On fabrique de nouveaux générateurs :

$$g_k = A \otimes h_i \otimes g_j \oplus B \otimes g_j \otimes h_i$$

L'algorithme de X. Allamigeon en quelques mots

```
function Gout=Um
//G=UmaLineaAxLe
//A, B, et Gin c                                    sont des gen
//Si l'ensemble                                     lon partout).
//l'ensemble des
//Cette procedure provient de l'algorithme de Xavier Allamigeon 'Double sid
//

[r,n]=size(Gin);
new=%F;
if (full(A) > fu
elseif Gin==%zer
else
  p=0;q=0; G=%zeros(r,1); H=%zeros(r,1);GG=%zeros(r,1);GGG=%zeros(r,1);
  for i=1:n
     x=Gin(:,i);
     if (full(A*
     else p=p+1;
     end
  end;
  if (p>0) then
    k=0;
```

1/ On trie les générateurs :

- dans G, ceux qui vérifient l'inégalité ;
- dans H, les autres.

2/ On fabrique de nouveaux générateurs :

$$g_k = A \otimes h_i \otimes g_j \oplus B \otimes g_j \otimes h_i$$

3/ On garde les générateurs vraiment nouveaux.

Linear Algebra and its Applications 435 (2011) 1758–1768

Contents lists available at ScienceDirect

Linear Algebra and its Applications

journal homepage: www.elsevier.com/locate/laa

Basic solutions of systems with two max-linear inequalities ☆

Sergeĭ Sergeev [a,*], Edouard Wagneur [b]

[a] University of Birmingham, School of Mathematics, Watson Building, Edgbaston B15 2TT, UK
[b] GERAD and Département de Mathématiques et Génie Industriel École Polytechnique de Montréal, PQ, Canada

ARTICLE INFO

Article history:
Available online 25 March 2011

Submitted by J.J. Loiseau

ABSTRACT

We give an explicit description of the basic solutions of max-linear systems $A \otimes x \leqslant B \otimes x$ with two inequalities.

4.1. A simple example

To illustrate the sets of generators constructed in the paper on a simple example, we consider the following system of two inequalities with four variables:

$$\begin{aligned} 4 \otimes x_3 \oplus 2 \otimes x_4 &\leqslant x_1 \oplus 2 \otimes x_2, \\ 3 \otimes x_1 \oplus x_3 &\leqslant x_2. \end{aligned} \tag{29}$$

Calculer une image

4.1. A simple example

To illustrate the sets of generators constructed in the paper on a simple example, we consider the following system of two inequalities with four variables:

$$\begin{aligned} 4 \otimes x_3 \oplus 2 \otimes x_4 &\leqslant x_1 \oplus 2 \otimes x_2, \\ 3 \otimes x_1 \oplus x_3 &\leqslant x_2. \end{aligned} \tag{29}$$

```
-->help mpsolve

-->A=#([%0 %0 4 2; 3 %0 0 %0]);

-->B=#([0 2 %0 %0;%0 0 %0 %0]);

-->AB=A+B
 AB  =


!0   2   4   2   !
!                !
!3   0   0   .   !

-->X=mpsolve(AB,B);
```

```
-->X=mpsolve(AB,B);

-->full(X)
 ans  =


!-3   .   .   .    !
!                  !
!0    0   0   0    !
!                  !
!.    .   .   -2   !
!                  !
!.    .   0   .    !

-->
```

In this example, the basis consists of four generators in S_1, S_{2A1}, S_{2A2} and S_{2B}: e_2, $e_2 \oplus e_4$, $3e_2 \oplus e_1$ and $2e_2 \oplus e_3$. Indeed, the remaining two generators in S_3 are redundant: (1) $5e_2 \oplus 2e_1 \oplus e_4$ (S_{3B2})

Calculer une image

[11] E. Wagneur, L. Truffet, F. Faye, M. Thiam. Tropical cones defined by max-linear inequalities, in: G.L. Litvinov, S.N. Sergeev (Eds.), Tropical and Idempotent Mathematics, Contemporary Mathematics, vol. 495, AMS, Providence, 2009, pp. 351–366.

4.2. An example from [11]

To compare our results with the approach of [11], we consider [11], Example 4.2, which is a system of two inequalities with seven variables:

$$\begin{aligned} &x_4 \oplus 4 \otimes x_5 \oplus 2 \otimes x_6 \oplus 6 \otimes x_7 \leqslant x_1 \oplus 1 \otimes x_2 \oplus 5 \otimes x_3, \\ &5 \otimes x_2 \oplus 6 \otimes x_3 \oplus 2 \otimes x_7 \leqslant 3 \otimes x_1 \oplus x_4 \oplus 2 \otimes x_5 \oplus 4 \otimes x_6. \end{aligned} \tag{30}$$

4.2. An example from [11]

To compare our results with the approach of [11], we consider [11], Example 4.2, which is a system of two inequalities with seven variables:

$$\begin{aligned} x_4 \oplus 4 \otimes x_5 \oplus 2 \otimes x_6 \oplus 6 \otimes x_7 &\leqslant x_1 \oplus 1 \otimes x_2 \oplus 5 \otimes x_3, \\ 5 \otimes x_2 \oplus 6 \otimes x_3 \oplus 2 \otimes x_7 &\leqslant 3 \otimes x_1 \oplus x_4 \oplus 2 \otimes x_5 \oplus 4 \otimes x_6. \end{aligned} \tag{30}$$

```
-->A=#([%0 %0 %0 0 4 2 6; %0 5 6 %0 %0 %0 2]);

-->A
 A  =


!.   .   .   0   4   2   6   !
!                             !
!.   5   6   .   .   .   2   !

-->B=#([%1 1 5 %0 %0 %0 %0; 3 %0 %0 0 2 4 %0])
 B  =


!0   1   5   .   .   .   .   !
!                             !
!3   .   .   0   2   4   .   !
```

```
-->X=mpsolve(AB,B);

-->full(X)
 ans  =


!0   0   0    0    0    0    .    .    .    0    0    .    0    .    .    0    !
!                                                                              !
!.   .   .    .    -2   .    2    .    .    .    .    .    .    .    .    .    !
!                                                                              !
!.   .   .    .    .    -3   0    1    1    -3   -3   1    -3   1    0    .    !
!                                                                              !
!.   0   .    .    .    .    .    .    .    .    .    6    2    .    .    .    !
!                                                                              !
!.   .   -4   .    .    .    .    2    .    -2   .    .    .    .    .    .    !
!                                                                              !
!.   .   .    -2   .    .    3    3    3    .    .    3    .    3    3    .    !
!                                                                              !
!.   .   .    .    .    -4   .    .    0    .    .    .    .    .    .    -6   !

-->
```

Thus the basis consists of e_1, 8 combinations of two unit vectors and 7 combinations of three unit vectors.

The two-combinations are: $e_1 \oplus e_4$, $4e_1 \oplus e_5$ and $2e_1 \oplus e_6$ (S_{2A1}), $2e_1 \oplus e_2$ and $3e_1 \oplus e_3$ (S_{2A2}), $6e_1 \oplus e_7$ (S_{2B}), $2e_6 \oplus e_3$ and $e_3 \oplus 3e_6$ (S_{2C}).

The three-combinations are: $3e_1 \oplus e_3 \oplus 5e_4$, $3e_1 \oplus e_3 \oplus 1e_5$ (S_{3B2}), $4e_1 \oplus 1e_3 \oplus e_7$ (S_{3C2}), $2e_2 \oplus e_3 \oplus 3e_6$ (S_{3D1}), $2e_6 \oplus e_3 \oplus 5e_4$ and $2e_6 \oplus e_3 \oplus 1e_5$ (S_{3D2}), $3e_6 \oplus 1e_3 \oplus e_7$ (S_{3E}).

Calculer un noyau

```
-->D=%zeros(4,4)
 D  =


(    4,    4) zero sparse matrix


-->D(1,1)=-3
 D  =


(    4,    4) sparse matrix

(    1,    1)      - 3.

-->D(2,:)=%ones(1,4);

-->D(3,4)=-2;

-->D(4,3)=0
 D  =


(    4,    4) sparse matrix

(    1,    1)      - 3.
(    2,    1)        0.
(    2,    2)        0.
(    2,    3)        0.
(    2,    4)        0.
(    3,    4)      - 2.
(    4,    3)        0.
```

Calculer un noyau

```
-->D=%zeros(4,4)
 D  =


(    4,    4) zero sparse matrix


-->D(1,1)=-3
 D  =


(    4,    4) sparse matrix

(    1,    1)      - 3.

-->D(2,:)=%ones(1,4);

-->D(3,4)=-2;

-->D(4,3)=0
 D  =


(    4,    4) sparse matrix

(    1,    1)      - 3.
(    2,    1)        0.
(    2,    2)        0.
(    2,    3)        0.
(    2,    4)        0.
(    3,    4)      - 2.
(    4,    3)        0.
```

```
-->[K,V]=mpkernel(full(D))
 V  =


!0  .  -2  .  .  .  0  .  .  -4  .  !
!                                   !
!.  .  .   .  0  .  .  0  0  -4  .  !
!                                   !
!0  .  .   0  0  .  .  .  .  .   0  !
!                                   !
!.  0  .   .  .  0  .  0  .  .   .  !
 K  =


!0  .  -2  .  .  .  0  .  .  -4  .  !
!                                   !
!0  0  .   0  .  .  0  .  0  .   .  !
!                                   !
!0  .  .   0  0  .  .  .  .  .   0  !
!                                   !
!.  0  .   .  .  0  .  0  .  .   .  !

-->D*K==D*V
 ans  =

  T T T T T T T T T T
  T T T T T T T T T T
  T T T T T T T T T T
  T T T T T T T T T T

 -->
```

Calculer un noyau, bis

```
-->[K,V]=mpkernel(full(D))
 V  =


!0  .  -2  .  .  .  0  .  .  -4  .  !
!                                   !
!.  .  .  .  0  .  .  0  0  -4  .  !
!                                   !
!0  .  .  0  0  .  .  .  .  .  0  !
!                                   !
!.  0  .  .  .  0  .  0  .  .  .  !
 K  =


!0  .  -2  .  .  .  0  .  .  -4  .  !
!                                   !
!0  0  .  0  .  .  0  .  0  .  .  !
!                                   !
!0  .  .  0  0  .  .  .  .  .  0  !
!                                   !
!.  0  .  .  .  0  .  0  .  .  .  !

-->D*K==D*V
 ans  =

  T T T T T T T T T T T
  T T T T T T T T T T T
  T T T T T T T T T T T
  T T T T T T T T T T T

-->
```

```
-->P=full([D %zeros(4,4)]);

-->Q=full([%zeros(4,4) D]);

-->R=mpsolve(P,Q);

-->R
 R  =


!0  .  -2  .  .  .  0  .  .  -4  .  !
!                                   !
!0  0  .  0  .  .  0  .  0  .  .  !
!                                   !
!0  .  .  0  0  .  .  .  .  .  0  !
!                                   !
!.  0  .  .  .  0  .  0  .  .  .  !
!                                   !
!0  .  -2  .  .  .  0  .  .  -4  .  !
!                                   !
!.  .  .  .  0  .  .  0  0  -4  .  !
!                                   !
!0  .  .  0  0  .  .  .  .  .  0  !
!                                   !
!.  0  .  .  .  0  .  0  .  .  .  !

-->equalspan(R,[K;V])
 ans  =

  T

-->
```

1. Cônes, modules, noyaux
   - L'algorithme de X. Allamigeon
   - Calculer une image, un moyau

2. Deux ou trois exercices d'application
   - Un problème de contraintes temporelles
   - Un problème de synchronisation
   - Un problème d'observation

3. Conclusion

Quelques exercices d'application

Un problème de contraintes temporelles

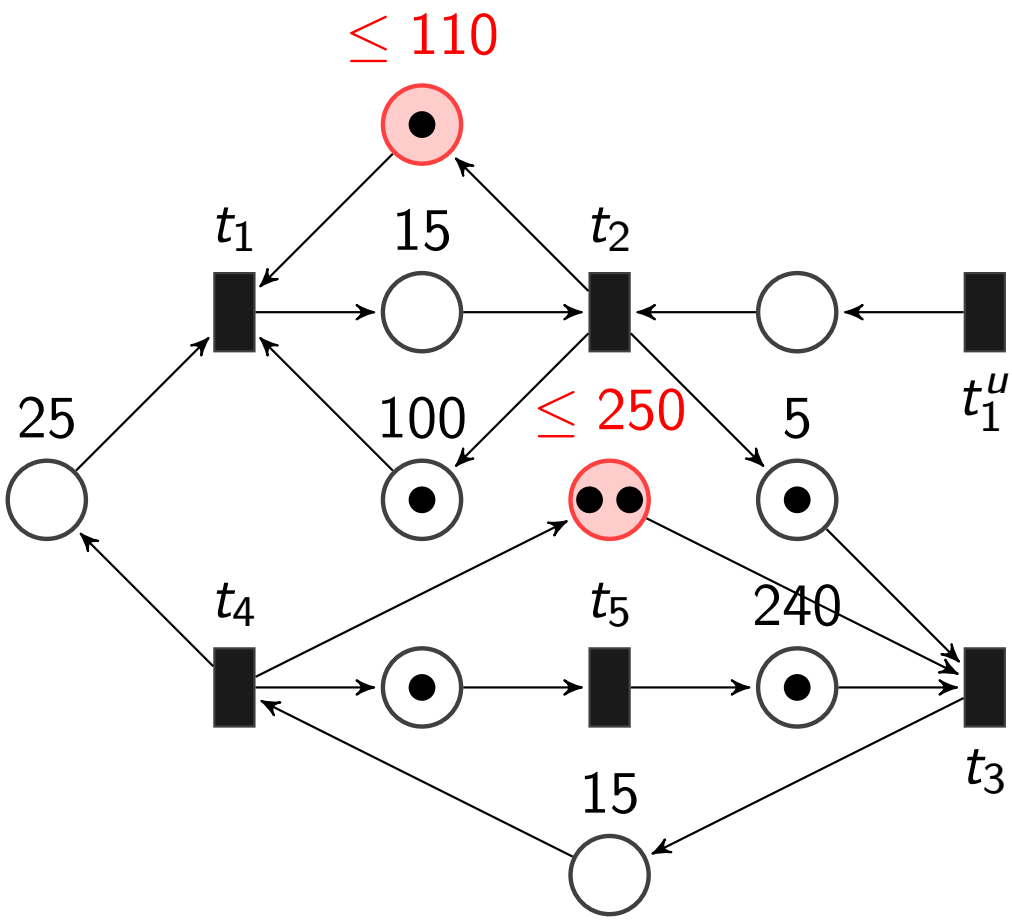


Figure – Un graphe d'événements temporisé, les durées de séjour des jetons dans les places en rouge sont soumises à une contrainte de temps.

Un problème de contraintes temporelles

Le comportement dynamique du graphe d'événements temporisés représenté dans la figure (1) s'obtient avec le système linéaire en max-plus suivant. Cet exemple provient de [4], il a été également étudié dans [2], [3] et [1].

$$x(k+1) = \quad A \otimes x(k) \oplus B \otimes u(k), \tag{1}$$

avec les matrices définies ainsi :

$$A = \begin{pmatrix} \cdot & 100 & \cdot & \cdot & 280 \\ \cdot & 115 & \cdot & \cdot & 295 \\ \cdot & 5 & \cdot & \cdot & 240 \\ \cdot & 20 & \cdot & \cdot & 255 \\ \cdot & \cdot & \cdot & 0 & \cdot \end{pmatrix} B = \begin{pmatrix} \cdot \\ 0 \\ \cdot \\ \cdot \\ \cdot \end{pmatrix}.$$

Ls contraintes de temps de séjour, illustrées par des places rouges dans la figure (1), sont respectées quand le vecteur d'état vérifie les inégalités suivantes :

$$x_1(k) \leq 110 \otimes x_2(k-1) \tag{2}$$

$$x_3(k) \leq 250 \otimes x_4(k-2). \tag{3}$$

Afin de mettre les contraintes sous la forme :

$$C_x \otimes x \leq x, \tag{4}$$

le système (1) est étendu :

$$A_e = \begin{pmatrix} . & 100 & . & . & 280 & . & . \\ . & 115 & . & . & 295 & . & . \\ . & 5 & . & . & 240 & . & . \\ . & 20 & . & . & 255 & . & . \\ . & . & . & 0 & . & . & . \\ . & 0 & . & . & . & . & . \\ . & . & . & 0 & . & . & . \\ . & . & . & . & . & 0 & . \end{pmatrix} \quad B_e = \begin{pmatrix} . \\ 0 \\ . \\ . \\ . \\ . \\ . \end{pmatrix}.$$

La matrice C_x contenant ε partout sauf $C_x(6,1) = -110$ et $C_x(8,3) = -250$. Le module $\mathcal{K} = \mathrm{Im}\, C_x^{\star}$ n'est pas invariant controlé. Il convient alors de rechercher un module $\mathcal{R} \subset \mathcal{K}$, qui vérifie l'égalité $A_e R \oplus B_e U = RW$.

Un problème de synchronisation

Une clef du Lean Management : le Tackt Time

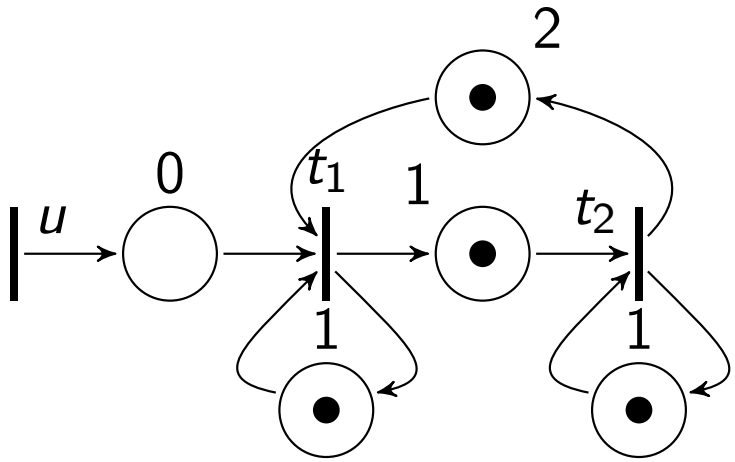
The Plant

$$(\Sigma_1) \quad \left\{ x(k+1) = A \otimes x(k) \oplus B \otimes u(k) \right.$$

The Reference

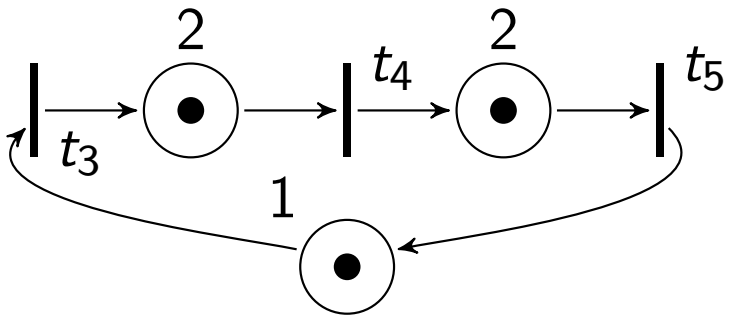
$$(\Sigma_2) \quad \left\{ w(k+1) = A_r w(k) \right.$$

La chaine de production



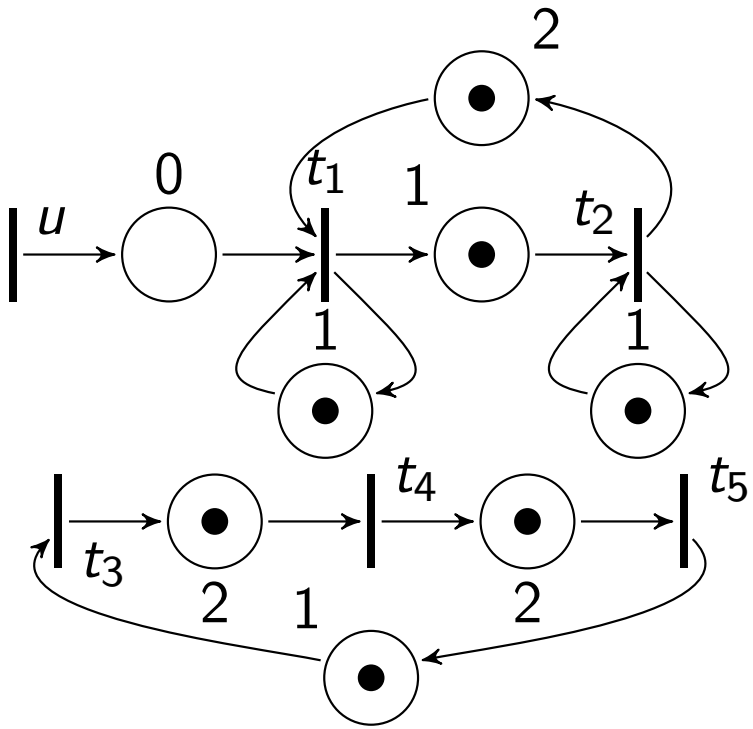
$$\sum_1 : A = \begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix}, B = \begin{pmatrix} 0 \\ \epsilon \end{pmatrix}$$

La référence (Tackt Time)



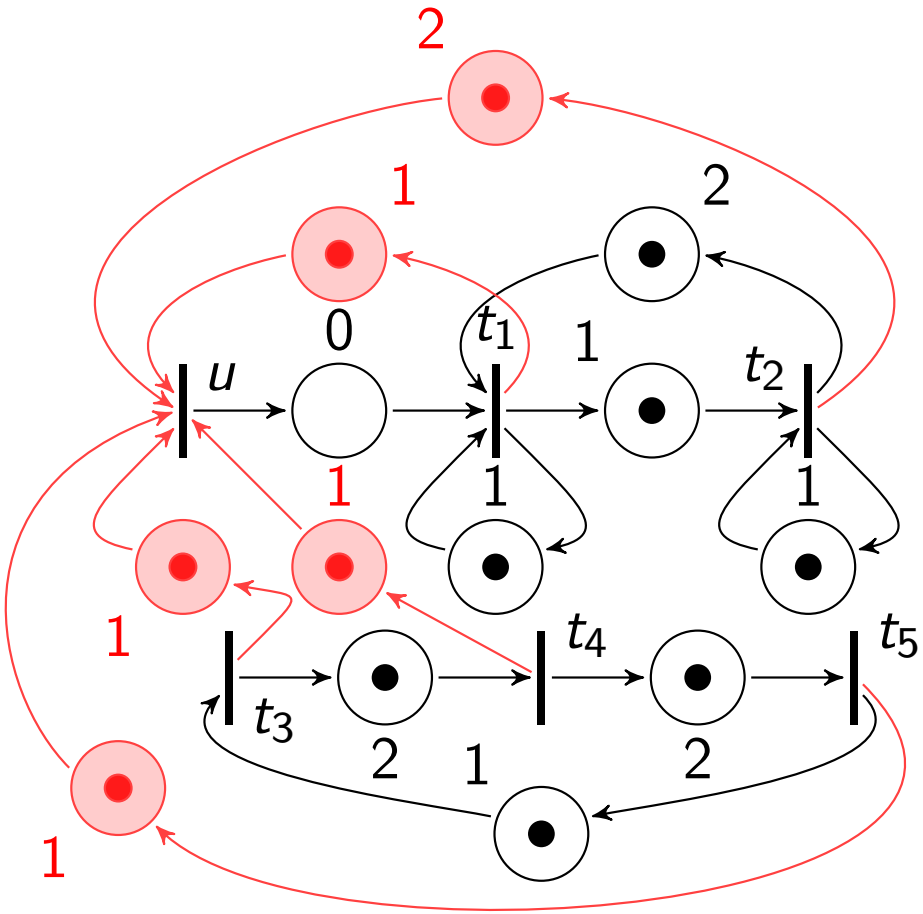
$$\sum_{2} : A_r = \begin{pmatrix} \epsilon & \epsilon & 1 \\ 2 & \epsilon & \epsilon \\ \epsilon & 2 & \epsilon \end{pmatrix} .$$

Le système étendu



$$A_e = \begin{pmatrix} 1 & 2 & \epsilon & \epsilon & \epsilon \\ 1 & 1 & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & 1 \\ \epsilon & \epsilon & 2 & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & 2 & \epsilon \end{pmatrix}, Be = \begin{pmatrix} 0 \\ \epsilon \\ \epsilon \\ \epsilon \\ \epsilon \end{pmatrix}.$$

Une solution au problème de synchronisation



$$F = \begin{pmatrix} 1 & 2 & 1 & 1 & 1 \end{pmatrix}$$

Un problème d'observation

ELSEVIER

Automatica

Volume 107, September 2019, Pages 103-111

On Max-plus linear dynamical system theory: The observation problem ☆

Vinicius Mariano Gonçalves [a], Carlos Andrey Maia [a], Laurent Hardouin [b]

Show more

https://doi.org/10.1016/j.automatica.2019.05.026 ↗

On a un système d'équations :

$$\begin{cases} x(k+1) = A \otimes x(k) \oplus B \otimes u(k) \\ y(k) = C \otimes x(k), \\ z(k) = D \otimes x(k), \end{cases}$$

On souhaite reconstruire $z(k)$ à partir des mesures $y(k)$ et d'un observateur.

$$\left\{\begin{array}{lcl} x(k+1) & = & A \otimes x(k) \\ y(k) & = & C \otimes x(k), \\ z(k) & = & D \otimes x(k), \end{array}\right.$$

avec

$$A = \begin{pmatrix} 0 & -10 & -10 & 1 \\ 0 & 0 & -10 & 2 \\ -10 & 0 & 0 & 3 \\ -10 & -10 & 0 & 4 \end{pmatrix}, \quad C = \begin{pmatrix} \ldots 0 \end{pmatrix}, \quad D = \begin{pmatrix} 0 \ldots 0 \end{pmatrix},$$

On souhaite reconstruire $z(k)$ à partir des mesures $y(k)$ et de l'observateur dynamique :

$$\left\{ w(k+1) = W \otimes w(k) \oplus Y \otimes y(k) \right. \tag{5}$$

Dans cet exemple, $\mathrm{Ker}D$, le noyau de D n'est pas invariant conditionnel :

$$A(\mathrm{Ker}D \cap \mathrm{Ker}C) \nsubseteq \mathrm{Ker}D$$

.

On s'intéresse à la plus petite congruence (C, A)-invariante contenant I_n :

$$\mathcal{S}^{k+1} = \mathrm{Ker} I_n \oplus A\left(\mathcal{S}^k \cap \mathrm{Ker} C\right) \ , k \in \mathbb{N} \ , \ \mathcal{S}^0 = \mathrm{Ker} I_n \ ,$$

En effet, on souhaite vérifier que :

$$\mathcal{S}^{\star}_{\mathrm{Ker} I_n}(C, A) \subset \mathrm{Ker} D$$

```
-->
 A=


!0    -10   -10   1   !
!                     !
!0    0     -10   2   !
!                     !
!-10  0     0     3   !
!                     !
!-10  -10   0     4   !

 B=


!0   !
!    !
!0   !
!    !
!0   !
!    !
!0   !
```

```
 C=


[More (y or n) ?]

!.   .   .   0   !

 D=


!0   .   .   0   !

 Is A(KerC \Cap KerD) \subset KerD ?

  F

 Is S1 \subset KerD ?

  T
```

1. Cônes, modules, noyaux
   - L'algorithme de X. Allamigeon
   - Calculer une image, un moyau
2. Deux ou trois exercices d'application
   - Un problème de contraintes temporelles
   - Un problème de synchronisation
   - Un problème d'observation
3. Conclusion

Conclusion

Conclusion

On pourrait aussi...

Références bibliographiques

Références bibliographiques

- Cárdenas, C., Loiseau, J. J., and Martinez, C. (2017). Invariance par retour d'état sur le demi-anneau max-plus. In MSR 2017, Modélisation des Systèmes Réactifs, pages 1 – 8, Marseille.
- V. M. Gonçalves, Carlos Andrey Maia, Laurent Hardouin, *On max-plus linear dynamical system theory : The regulation problem*, Automatica, Volume 75, January 2017, pp 202-209.
- R. Jacob and S. Amari, *Output feedback control of discrete processes under time constraint : application to cluster tools*, International Journal of Computer Integrated Manufacturing, Vol.30, 2017, pp. 880-894.
- C. Kim, T.-E. Lee Feedback control of cluster tools for regulating wafer delays IEEE Transactions on Automation Science and Engineering, 13 (2) (2016), pp. 1189–1199
- V. M. Gonçalves, Carlos Andrey Maia, Laurent Hardouin, *On Max-plus linear dynamical system theory : The observation problem*,

Remerciements et crédits

Équipe pédagogique

Auteur.rice.s : Jean Jacques Loiseau, Claude Martinez.

Intervenant.e.s : Jean Jacques Loiseau, Claude Martinez.

Crédits