Réseaux de Petri

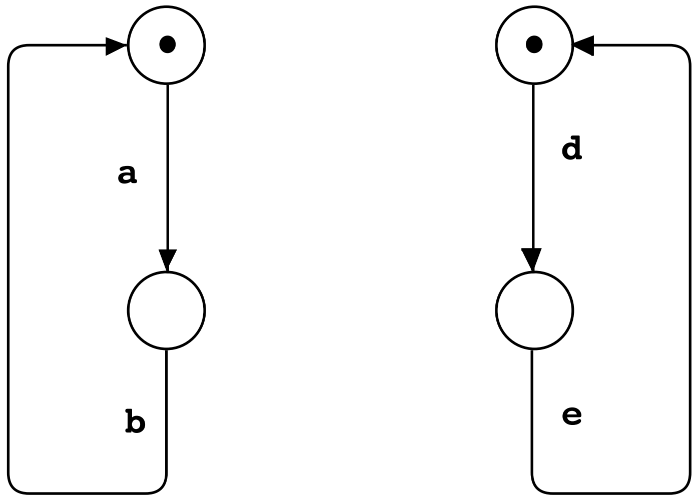
1 - Modélisation de systèmes finis

Des automates aux réseaux de Petri

- Problème : modéliser un système producteur/consommateur asynchrone avec tampon infini
  - Lorsque le tampon est non vide, le consommateur peut consommer indépendamment de l’état du producteur

a : produire un message

b : déposer un message



Producteur **Consommateur**

d : consommer un message

e : traiter le message

E. Encrenaz-Tiphène / C. Dutheillet

Sémantique du modèle par automates

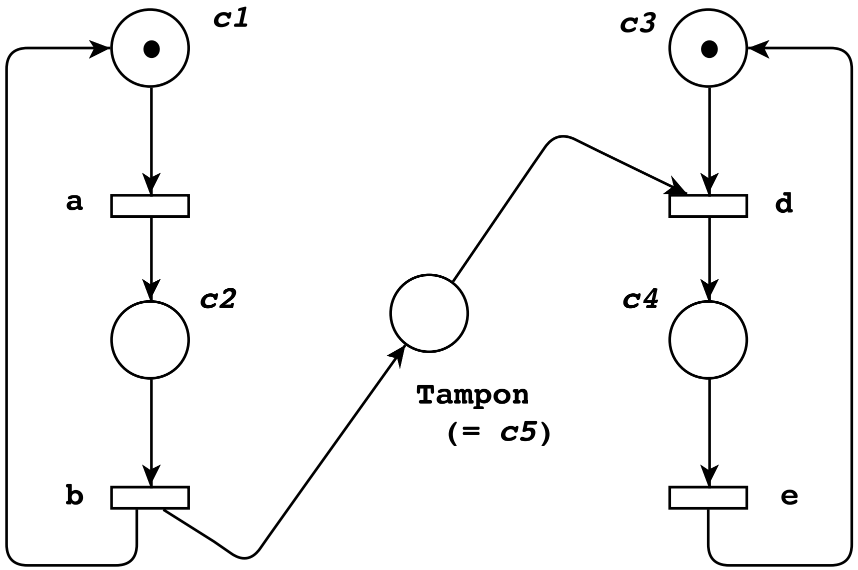
- Séquences possibles :
  - Ici, la séquence (de)* est acceptable (on considère que tout état est potentiellement terminal).
  - Le consommateur peut s'exécuter indéfiniment avant que le producteur ait déposé quelque chose dans le tampon.

- Nécessité de conditionner l'exécution de certaines actions
  - ajout de rectangles sur les arcs pour pouvoir "synchroniser" les conditions d'exécution d'une action
  - représentation des éléments de l'environnement qui conditionnent l'exécution : ici, le tampon.

Système condition / événement

- 2 types de nœuds :
  - Les conditions représentées par des cercles
  - Les événements représentés par des rectangles

- Modèle du producteur / consommateur



Notations

- On note :
  - •n l'ensemble des prédécesseurs du nœud n
  - n• l'ensemble des successeurs du nœud n
- Si n est un événement, on appelle
  - •n l'ensemble des pré-conditions de n
  - n• l'ensemble des post-conditions de n
- Exemples :
  - •d = {c3, c5} les pré-conditions de d sont c_3 et c_5
  - b• = {c1, c5} les post-conditions de b sont c_1 et c_5
  - •c_4 = {d} c_1• = {a}

Sémantique d'un système condition / événement

- Un pas du système C/E est défini par : $C_1 \xrightarrow{\lambda} C_2$

où C_1 et C_2 sont des sous-ensembles de conditions et où λ est l'étiquette d'un événement e qui vérifie :

1) $\bullet e \subseteq C_1$

autrement dit, toutes les pré-conditions sont remplies

2) $C_2 = (C1 \setminus \bullet e) \cup e\bullet$

autrement dit, le nouvel état est obtenu en annulant les pré-conditions et en validant les post-conditions

Sémantique d'un système C/E (suite)

- Une séquence du système C/E est définie par un mot :

$$\mathbf{w = \lambda_1\, \lambda_2 \dots \lambda_n}$$

tel qu'il existe des états $C_1, \dots, C_n$ vérifiant

$$\mathbf{C_{i-1} \xrightarrow{\lambda_i} C_i} \qquad \text{est un pas du système C/E}$$

- C_0 est un sous-ensemble de conditions représentant l'état initial

- Exemple : système C/E du producteur / consommateur

$$C_0 = \{ c_1, c_3 \} \qquad \bullet a = \{ c_1 \} \qquad a\bullet = \{ c_2 \}$$

$$\bullet a \subseteq C_0 \quad \Rightarrow \quad C_0 \xrightarrow{a} C_1 \quad \text{avec } C_1 = (C_0 - \bullet a) \cup a\bullet$$

$$C_1 = (\{ c_1, c_3 \} \setminus \{ c_1 \}) \cup \{ c_2 \} = \{ c_2, c_3 \}$$

Exemple de séquence

- **`ab`** est une séquence du système C/E modélisant le producteur/consommateur

$$C_0 \xrightarrow{a} C_1 \qquad \text{avec} \quad C_1 = \{ c_2 , c_3 \}$$

$$\bullet b = \{ c_2 \} \qquad \bullet b \subseteq C_1 \qquad \Rightarrow \qquad C_1 \xrightarrow{b} C_2$$

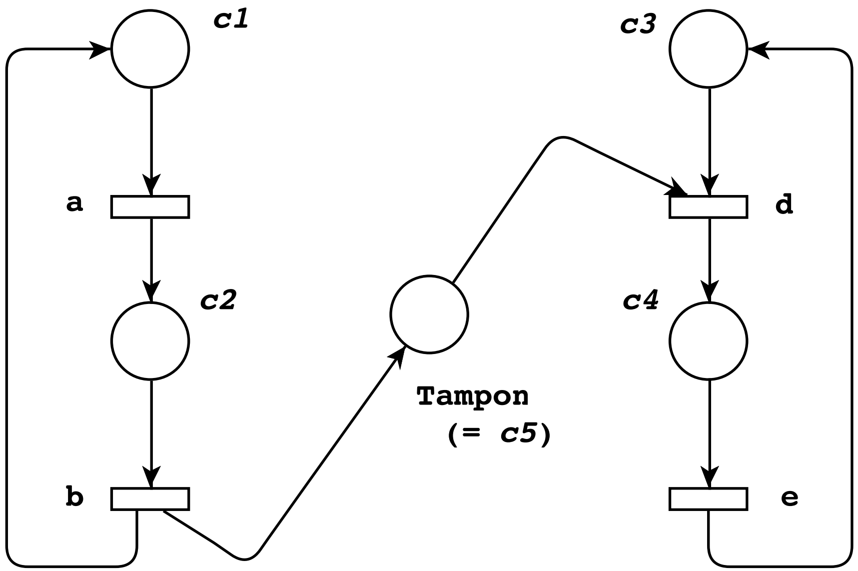
$$b\bullet = \{ c_1 , c_5 \} \qquad C_2 = (C_1 - \bullet b) \cup b\bullet = \{ c_1 , c_3 , c_5 \}$$

$$C_0 \xrightarrow{ab} C_2$$

- C_2 : une fois que le producteur a créé et stocké un message, il revient dans son état initial, le tampon est non vide, et le consommateur prêt à consommer.

Limites de la modélisation par système C/E

- Producteur / Consommateur :



$C_0 \xrightarrow{ab} C_2$

avec $C_2 = \{c_1 , c_3 , c_5 \}$

$C_2 \xrightarrow{ab} C_3$

avec $C_3 = \{c_1 , c_3 , c_5 \}$

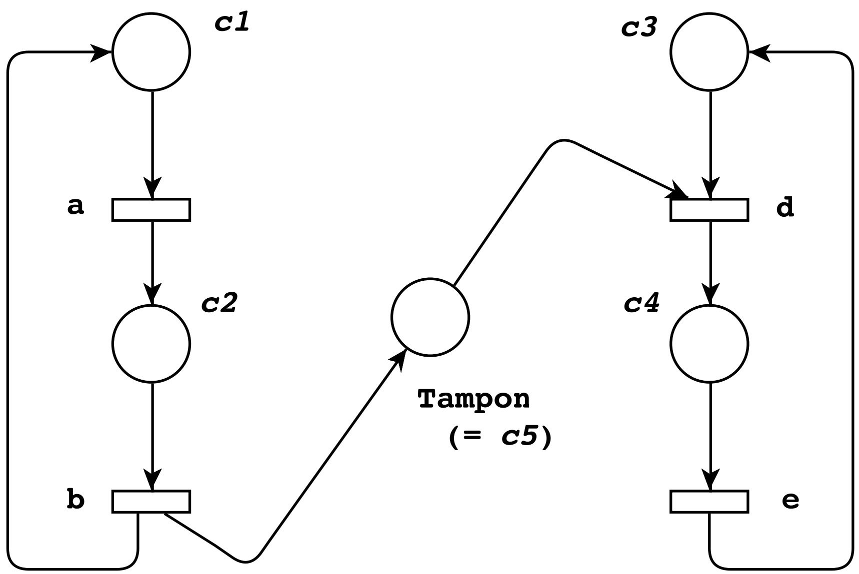
$C_3 \xrightarrow{de} C_4$ mais $C_4 = \{c_1 , c_3 \}$, donc $C_4 \cancel{\xrightarrow{d}}$

1 seule consommation possible, alors qu'il y a eu 2 productions !

Utiliser un modèle non booléen : les réseaux de Petri

Modèle RdP du producteur / consommateur

- Représentation :



Conditions → places

Evénements → transitions

- Etat d'un réseau de Petri = marquage M = $\sum_{p \in P} M(p).p$

  où M(p) est le nombre de jetons (marques) dans la place p.

Ex : après l'exécution de la séquence abab $\mathbf{M = c_1 + c_3 + 2.\ c_5}$

Modèle réseau de Petri

Définition :

Un réseau de Petri **R** est un quadruplet <P, T, Pré, Post> tel que :

P : ensemble fini de places ($P \neq \emptyset$)

T : ensemble fini de transitions ($T \neq \emptyset$, $P \cap T = \emptyset$)

Pré : $P \times T \rightarrow \mathbb{N}$

$Pré(p, t) = n \ (n > 0) \Leftrightarrow$

le franchissement de t est conditionné par la présence de *n* ressources dans p

Post : $P \times T \rightarrow \mathbb{N}$

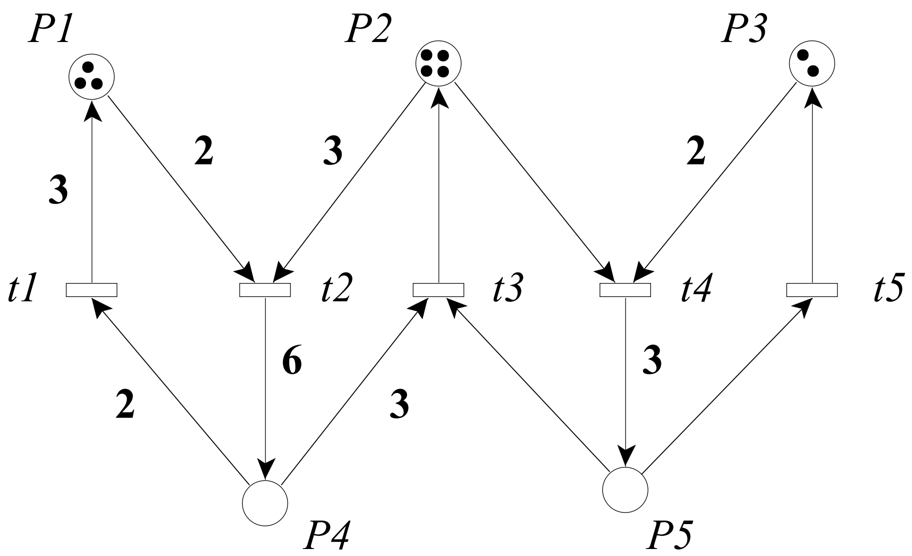
$Post(p, t) = n \ (n > 0) \Leftrightarrow$

le franchissement de t produit *n* ressources dans p

Un réseau **marqué** <R, M_0> possède en outre un marquage initial $M_0 \in \mathbb{N}^P$

Réseau de Petri - Exemple

Le marquage initial est M_0



$$M_0 = \begin{bmatrix} 3 \\ 4 \\ 2 \\ 0 \\ 0 \end{bmatrix}$$

Franchissement de transition

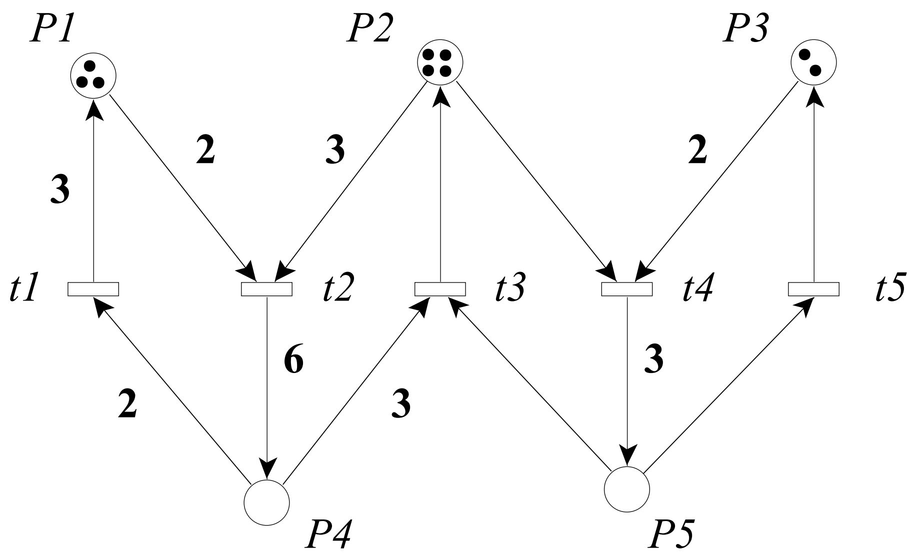
Règle de franchissement :

- $t \in T$ est franchissable à partir de M ssi $\forall\ p \in P$, $M(p) \geq Pré(p,t)$.

- si t est franchissable, alors son franchissement mène à l'état M' :
  $\forall\ p \in P,\ M'(p) = M(p) - Pré(p,t) + Post(p,t)$

Le franchissement de t est noté : M [t> M'.

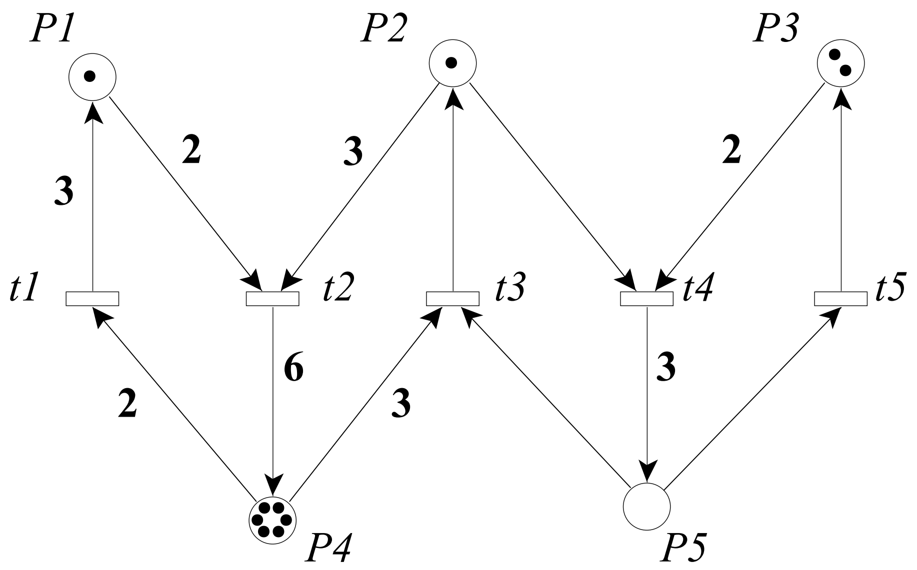
Franchissabilité - exemple



$\boldsymbol{t_2}$ et $\boldsymbol{t_4}$ sont franchissables à partir de M_0

On note $M_0\ [t_2>$ et $M_0\ [t_4>$

Franchissement - exemple

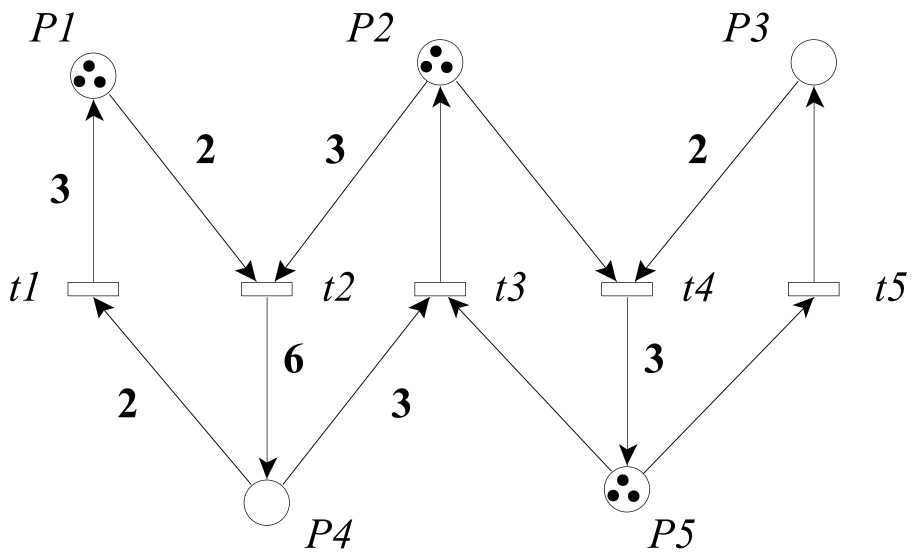


Le franchissement de $\boldsymbol{t_2}$ à partir de M_0 mène dans le marquage M_1

$$M_1 = \begin{bmatrix} 1 \\ 1 \\ 2 \\ 6 \\ 0 \end{bmatrix}$$

on note $M_0\ [t_2 > M_1$

Franchissement - exemple



Le franchissement de $\boldsymbol{t_4}$ à partir de M_0 mène dans le marquage M_2

$$M_2 = \begin{bmatrix} 3 \\ 3 \\ 0 \\ 0 \\ 3 \end{bmatrix}$$

on note $M_0\ [t_4 > M_2$

Séquence de franchissements

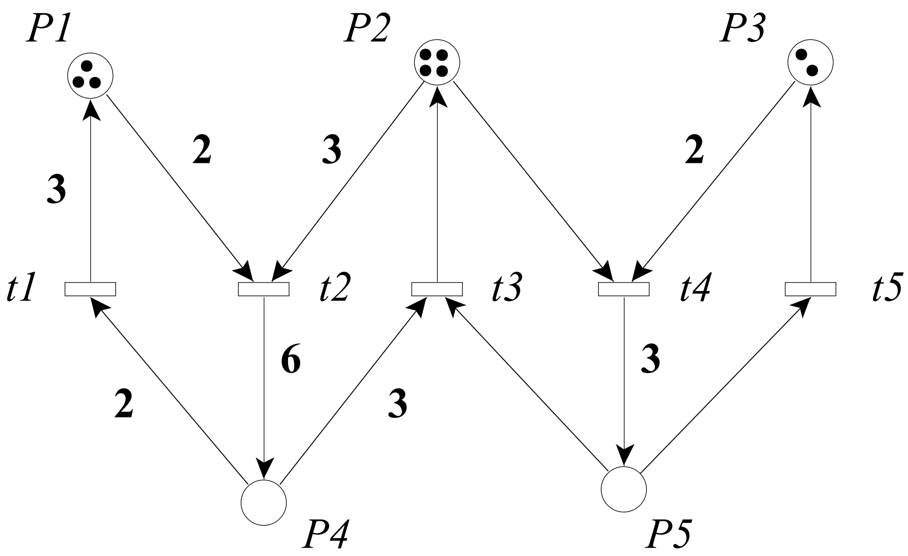
Définition :

Une séquence de franchissements de M_0 à M_n est un mot $\boldsymbol{t_0 \ldots t_{n-1}}$ tel qu'il existe des marquages $M_1,\ldots,M_{n-1}$ vérifiant

$$M_0\ [t_0{>}\ M_1\ \ldots\ M_{n-1}\ [t_{n-1}{>}\ M_n$$

E. Encrenaz-Tiphène / C. Dutheillet

Séquence de franchissements - Exemple



$\boldsymbol{t_2\, t_4\, t_3}$ est une séquence de franchissements à partir de M_0.

$$M_0\ [t_2 > M_1\ [t_4 > M_3\ [t_3 > M_4$$

$$M_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 6 \\ 3 \end{bmatrix} \qquad M_4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 3 \\ 2 \end{bmatrix}$$

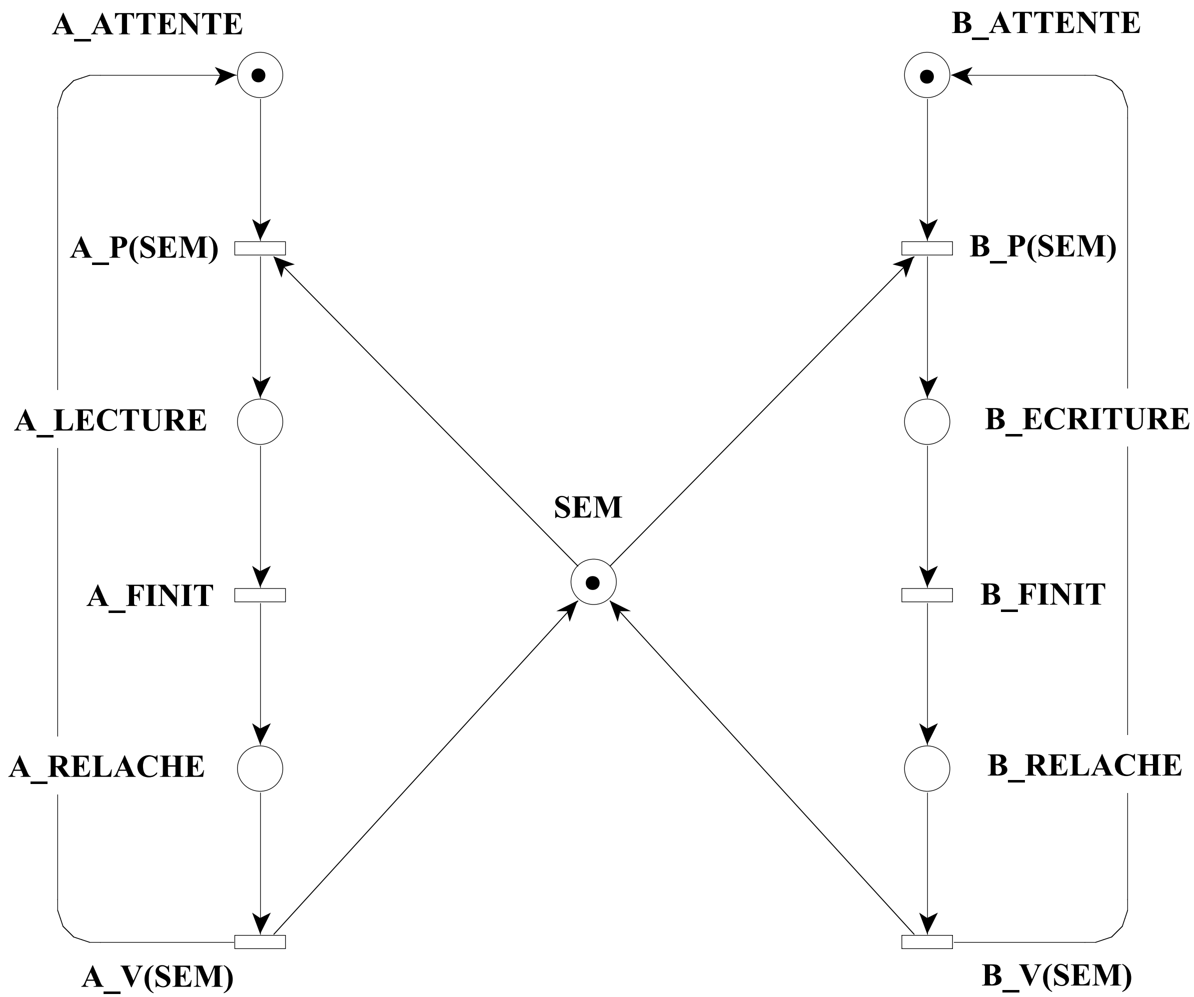
Exemple

partage de fichier : 1 Lecteur - 1 Ecrivain

```
Proc A{
      P(sem)
      lire_fichier
      V(sem)
      }
```

```
Proc B{
      P(sem)
      écrire_fichier
      V(sem)
      }
```

Exemple



A_ATTENTE
B_ATTENTE
A_P(SEM)
B_P(SEM)
A_LECTURE
B_ECRITURE
SEM
A_FINIT
B_FINIT
A_RELACHE
B_RELACHE
A_V(SEM)
B_V(SEM)

Exemple

partage de fichier : 2 Lecteurs - 1 Ecrivain

```
Proc A {
      P(sem)
      lire_fichier
      V(sem)
      }
```

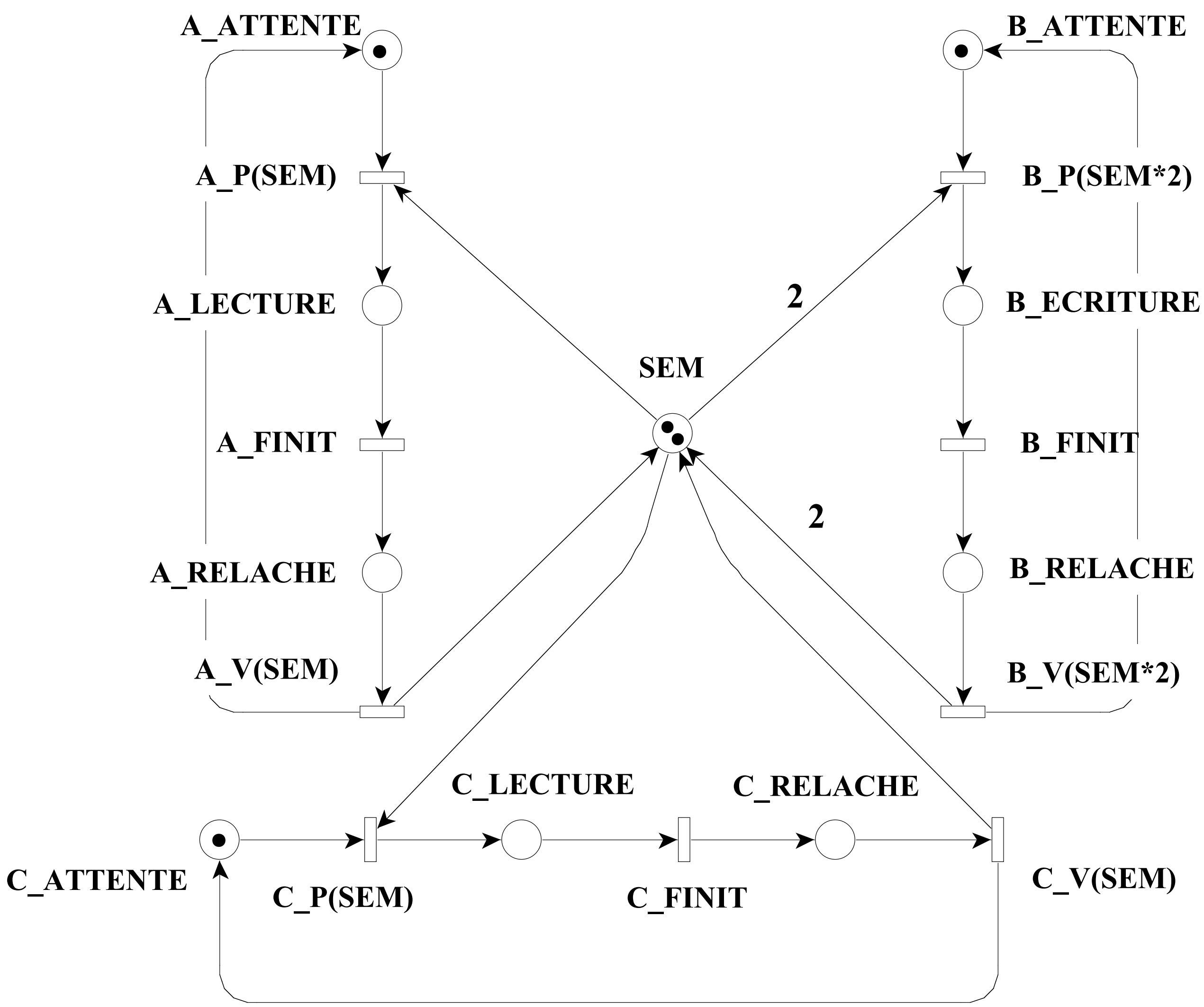
```
Proc B {
      P(sem,2)
      écrire_fichier
      V(sem,2)
      }
```

```
Proc C {
      P(sem)
      lire_fichier
      V(sem)
      }
```

Comment modéliser cette synchronisation pour 10 lecteurs et 3 écrivains (indifférenciés) ?

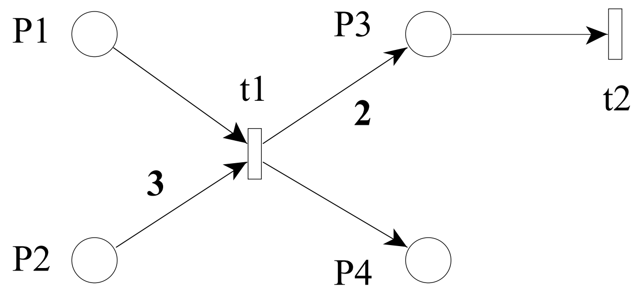
Comment généraliser à n lecteurs et p écrivains ?

Exemple


A_ATTENTE
B_ATTENTE
A_P(SEM)
B_P(SEM*2)
A_LECTURE
2
B_ECRITURE
SEM
A_FINIT
B_FINIT
2
A_RELACHE
B_RELACHE
A_V(SEM)
B_V(SEM*2)
C_LECTURE
C_RELACHE
C_ATTENTE
C_P(SEM)
C_FINIT
C_V(SEM)

Représentation matricielle

- Représentation graphique



- Représentation matricielle

$$\text{Pré} = \begin{array}{c} \\ P1 \\ P2 \\ P3 \\ P4 \end{array} \begin{array}{c} \begin{array}{cc} t1 & t2 \end{array} \\ \begin{bmatrix} 1 & 0 \\ 3 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \end{array} \qquad \text{Post} = \begin{array}{c} \\ P1 \\ P2 \\ P3 \\ P4 \end{array} \begin{array}{c} \begin{array}{cc} t1 & t2 \end{array} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 2 & 0 \\ 1 & 0 \end{bmatrix} \end{array}$$

Matrice d'incidence

Définition :

Soit R un réseau de Petri.

On définit C la matrice d'incidence de R par :

$$\mathbf{C = Post - Pré}$$

Franchissement :

Soit M[t>M'. On a :

$$\mathbf{M'(p)} = M(p) + Post(p, t) - Pré(p, t) = \mathbf{M(p) + C(p, t)}$$

Vecteur et équation caractéristiques

Définition : *vecteur caractéristique (vecteur de Parikh)*

Soit s une séquence de franchissements.

Le vecteur caractéristique $\underline{s}$ de la séquence s est un vecteur d'entiers indexé par les transitions.

La composante relative à la transition t représente le nombre d'occurrences de t dans s.

$$s = t_1\ t_2\ t_1\ t_4 \quad \Rightarrow \quad \underline{s} = \begin{bmatrix} 2 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

Définition : *équation caractéristique*

Soit **M [s> M'**, alors M' peut se déduire directement de M par application de l'équation caractéristique :

$$\mathbf{M' = M + C.\underline{s}}$$

Graphe d'accessibilité

Définition : Le graphe d'accessibilité d'un réseau marqué $<R, M_0>$ est un système de transitions $< Q, \Delta, \lambda, q_0 >$ tel que :

Q l'ensemble des marquages accessibles dans R à partir de M_0
$Q = \{ M \mid M \in \mathbb{N}^P \text{ et } \exists \sigma \in T^* \text{ tq } M_0 [\sigma\rangle M \}$

Δ l'ensemble des arcs reliant deux marquages de R accessibles à partir de M_0

$$\{(q_1, q_2) \in Q \times Q \mid t \in T, q_1[t\rangle q_2\}$$

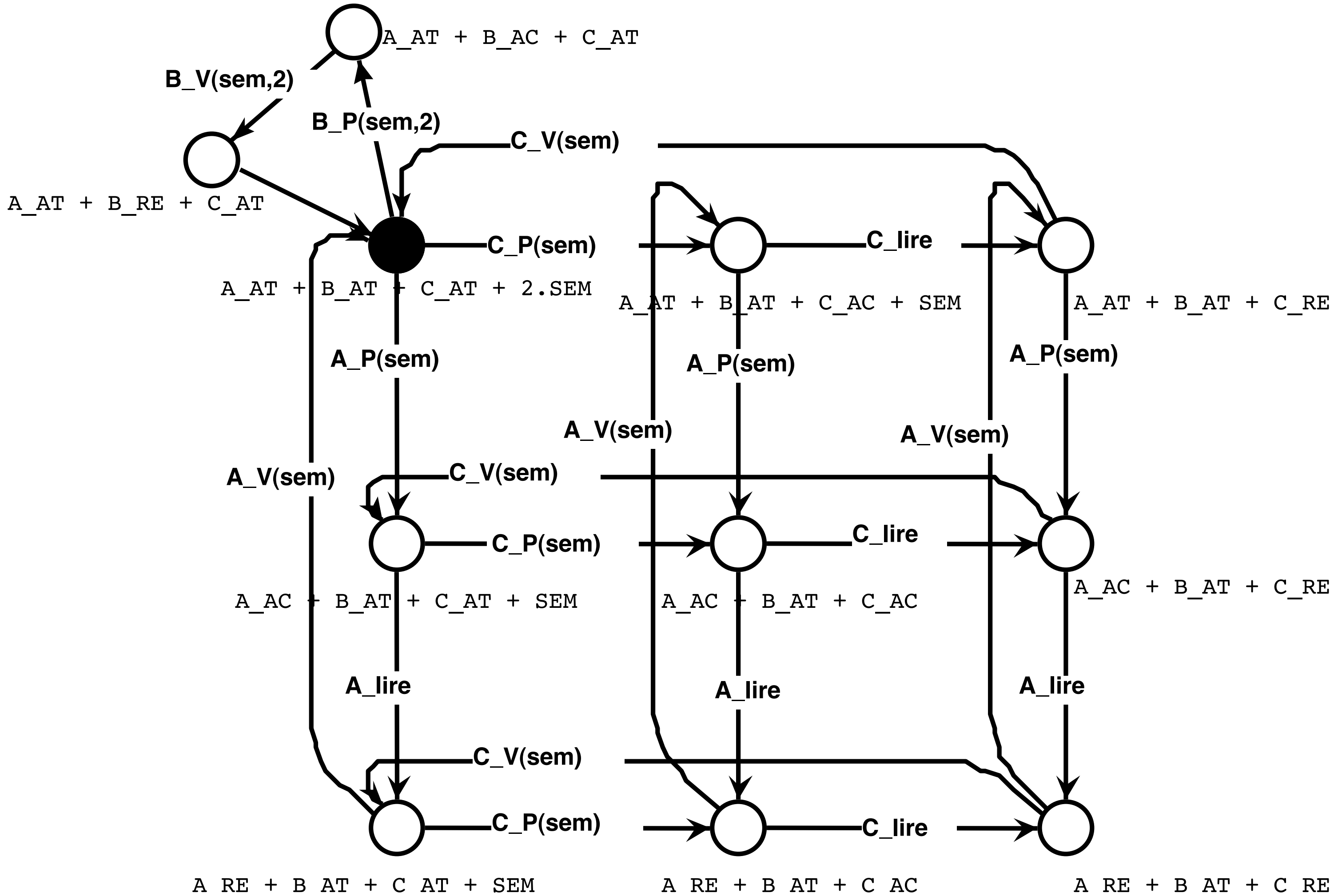
λ étiquette les arcs du graphe par le nom de la transition de R qui a été franchie

$\mathbf{q_0} = M_0$

Graphe d'accessibilité

```
nouveaux_états = Mo
graphe = < {Mo},ø,id,Mo >
Tq nouveaux_états <> ø faire
   état_en_cours = un élément de nouveaux_états
   nouveaux_états = nouveaux_états \ état_en_cours
   Pour toute transition t de T faire :
       Si état_en_cours[t> alors
         état_en_cours [t> état_futur
         Si état_futur est nouveau alors
             créer le noeud état_futur et
             l'ajouter à l'ensemble Q des noeuds du graphe
             ajouter état_futur à l'ensemble nouveaux_états
         FinSi
         ajouter au graphe état_en_cours -> état_futur
       FinSi
   FinPour
FinTq
Retourner graphe
```

Graphe d'accessibilité - exemple



E. Encrenaz-Tiphène / C. Dutheillet

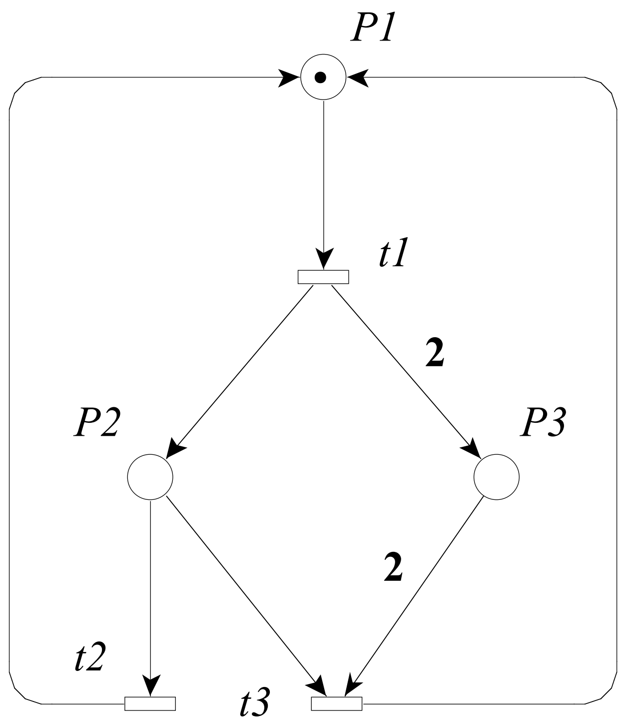
Graphe d'accessibilité - remarques

- Le graphe d'accessibilité dépend de R et de Mo.

- Un graphe fini peut contenir des séquences infinies.
  - ➤ existence de cycles

- Le graphe peut être infini.

Réseau dont le graphe est infini

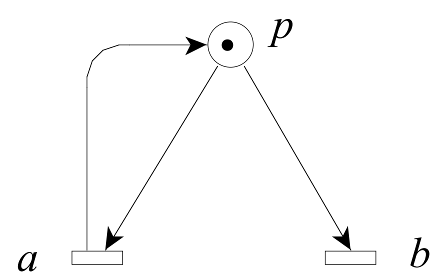


Propriétés fondamentales (1)

Séquence infinie :

<R, M_0> admet une séquence infinie s, où s est un mot infini sur T, si et seulement si pour tout s' préfixe fini de s, s' est une séquence de franchissement de <R, M_0>. Autrement dit, si $s = t_1.t_2. \ldots .t_n. \ldots$, alors pour tout i, si $s_i = t_1.\ldots.t_i$ on a : $M_0[s_i>$

<R, M_0> :

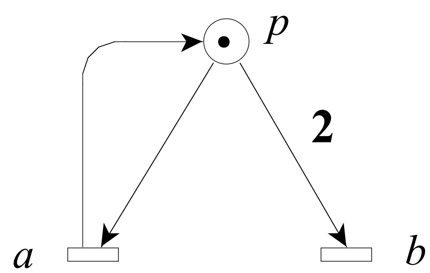


Propriétés fondamentales (2)

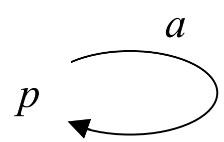
Pseudo-vivacité

<R, M_0> est pseudo-vivant si pour tout M appartenant à GMA(R, M_0) il existe une transition t telle que M[t>.

<R, M_0> :



GMA(R, M_0) :



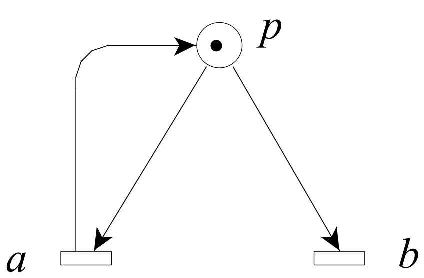
E. Encrenaz-Tiphène / C. Dutheillet

Propriétés fondamentales (3)

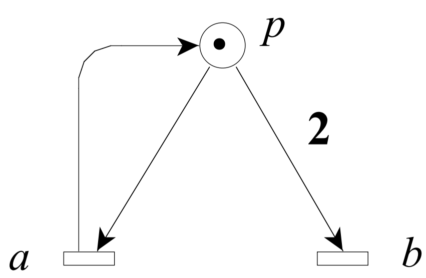
Quasi-vivacité

<R, M_0> est quasi-vivant si, pour toute transition t, il existe un marquage M appartenant à GMA(R, M_0) tel que M[t>.

<R, M_0> :



<R’, M_0> :

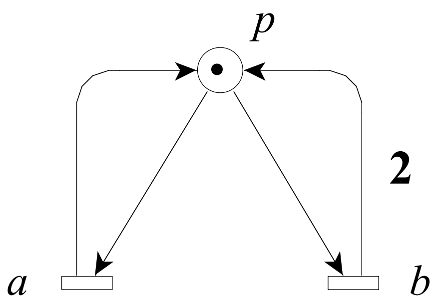


Propriétés fondamentales (4)

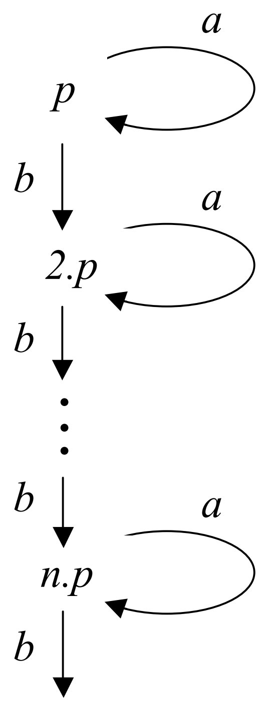
Vivacité

<R, M_0> est vivant si, pour tout marquage M appartenant à GMA(R, M_0), <R, M> est quasi-vivant.

<R, M_0> :



GMA(R, M_0) :



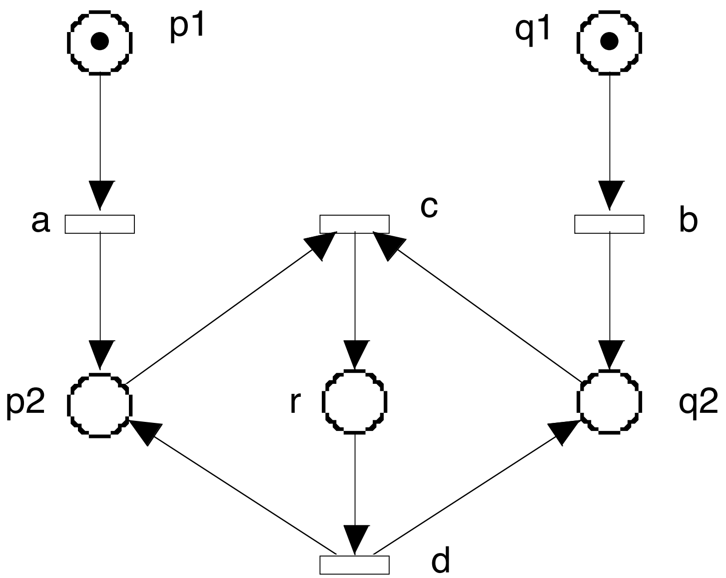
E. Encrenaz-Tiphène / C. Dutheillet

Propriétés fondamentales (5)

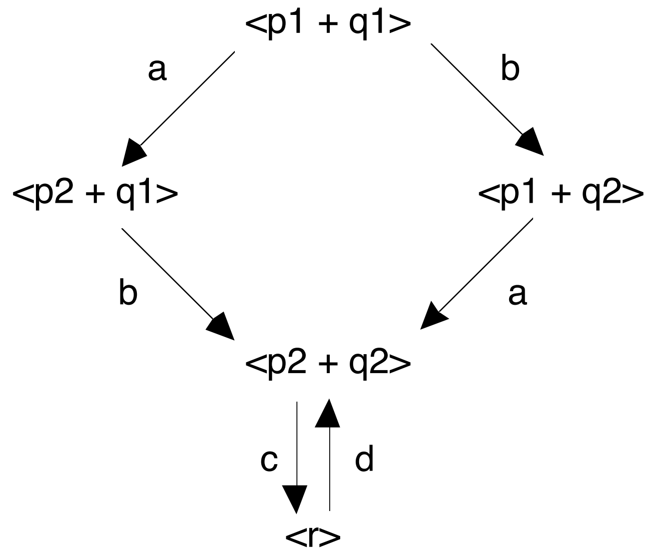
Etat d'accueil

<R, M_0> admet un état d'accueil M_a si, pour tout marquage M appartenant à GMA(R, M_0), il existe une séquence s telle que M[s>M_a.

(R)



G(R)



E. Encrenaz-Tiphène / C. Dutheillet

Propriétés fondamentales (6)

Caractère borné

<R, M_0> est non borné si, pour tout entier n, il existe un marquage M appartenant à A(R, M_0) et une place p tels que M(p) > n.

Le graphe d'accessibilité est infini

Trouver un moyen de représenter les états accessibles

Relations entre propriétés

- Si $<R, M_0>$ est pseudo-vivant ou non borné, alors
  $<R, M_0>$ admet une séquence infinie.

- Si $<R, M_0>$ est vivant, alors
  $<R, M_0>$ est quasi-vivant et pseudo-vivant.

- Si $<R, M_0>$ est quasi-vivant et admet M_0 pour marquage d'accueil, alors
  $<R, M_0>$ est vivant.

Monotonie des propriétés

- Soit Π une propriété de réseau de Petri.

  Π est dite monotone si et seulement si :

  $$\forall <R, M_0>, \quad \Pi(R, M_0) \quad \Rightarrow \quad \forall M'_0 \geq M_0, \quad \Pi(R, M'_0)$$

- Lemme de monotonie :

  $\forall <R, M_0>, \quad \forall M_1$ accessible à partir de M_0 :

  $$M_1 [s> M_2 \text{ et } M'_1 \geq M_1 \quad \Rightarrow \quad M'_1 [s> M'_2 \text{ et } M'_2 \geq M_2$$

Propriétés non monotones

- <R, M_0> est pseudo-vivant
- <R, M_0> est vivant
- <R, M_0> admet un état d'accueil

Propriétés monotones

- <R, M_0> admet une séquence infinie
- <R, M_0> est quasi-vivant
- <R, M_0> est non borné

Exemple de modélisation : l'algorithme de Peterson

- Algorithme de Peterson : exclusion mutuelle entre 2 processus
  - Les deux processus sont symétriques
  - Il existe une mémoire partagée contenant les variables `last`, $\texttt{dem}_p$ et $\texttt{dem}_q$
- Code du processus p :

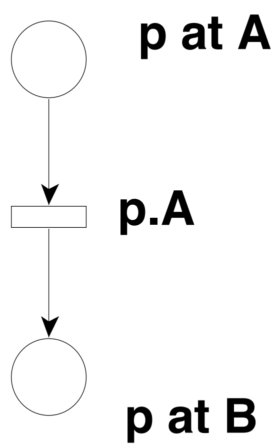
```
A :    dem_p = vrai
B :    last = p
C :    Attendre (last == q || dem_q  == faux)
D :    < Section critique >
E :    dem_p = faux; goto A
```

- Initialement : $\texttt{dem}_p = \texttt{dem}_q =$ faux
- $\texttt{last} = \text{p}$ ou $\texttt{last} = \text{q}$ indifféremment

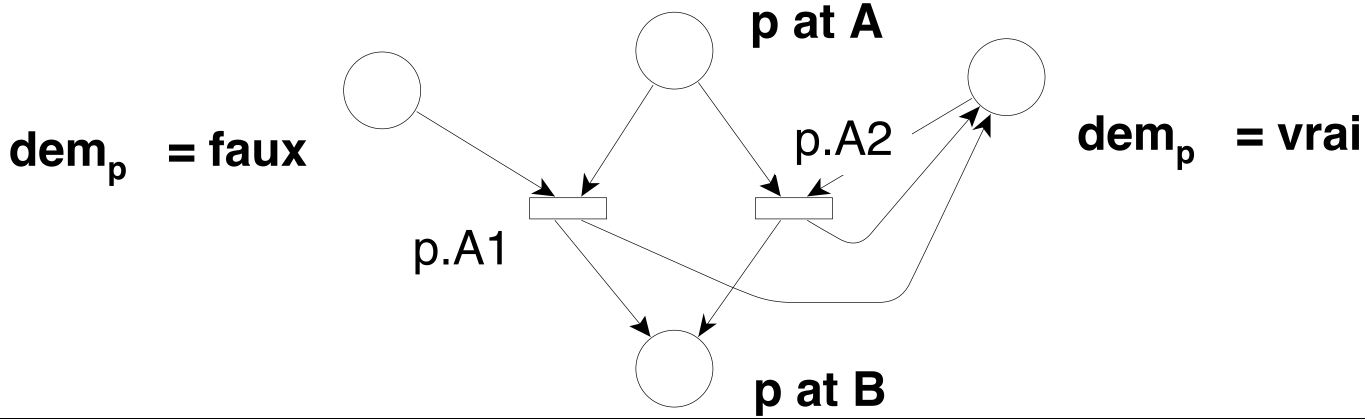
Exécution de la première instruction

`A :    demp  = vrai`

fait passer le processus p de l'état A à l'état B :



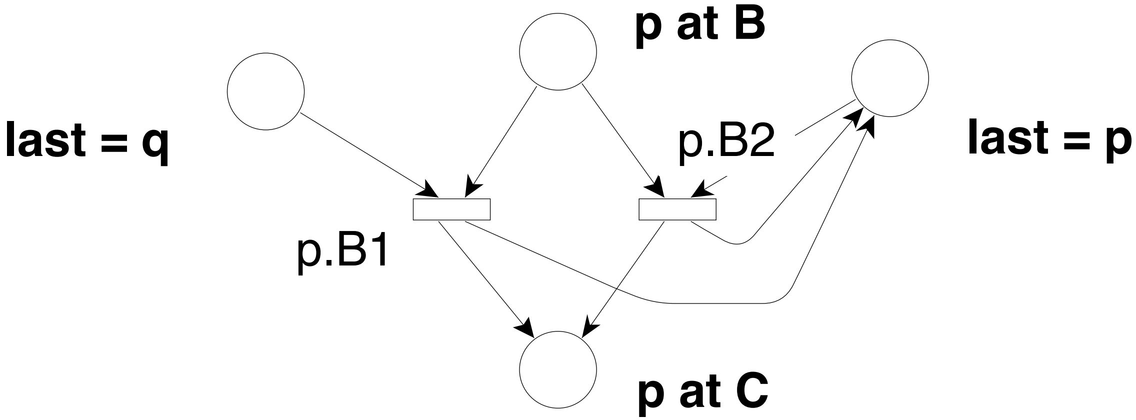
modifie (éventuellement) la valeur de la variable $\texttt{dem}_p$:



E. Encrenaz-Tiphène / C. Dutheillet

Exécution de la deuxième instruction

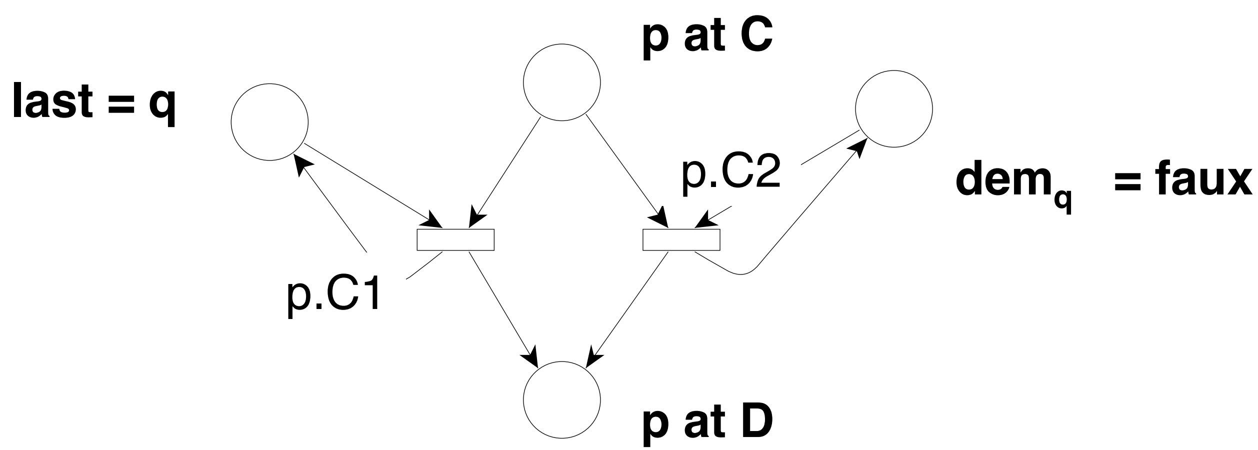
`B : last = p`



E. Encrenaz-Tiphène / C. Dutheillet

Modélisation de l'attente

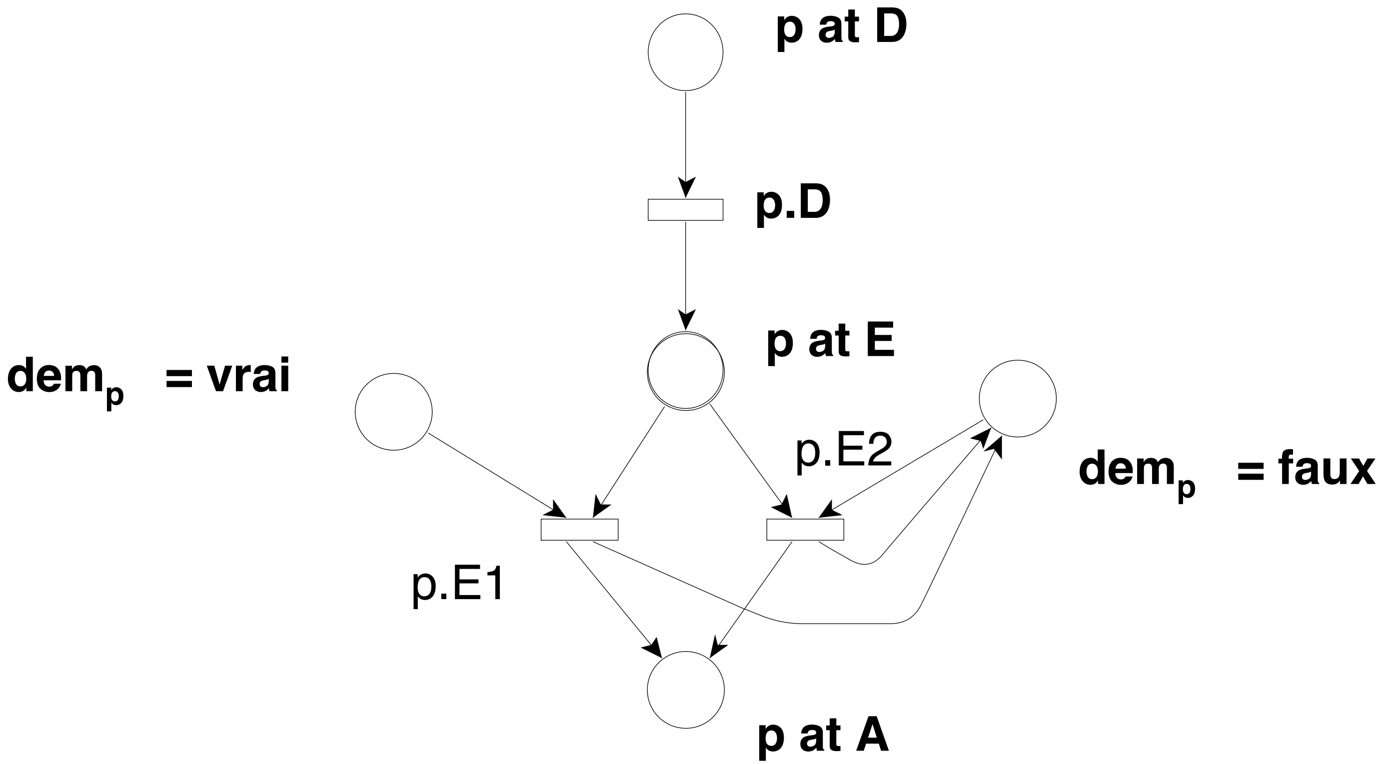
```
C :    Attendre (last == q || dem_q == faux)
```



E. Encrenaz-Tiphène / C. Dutheillet

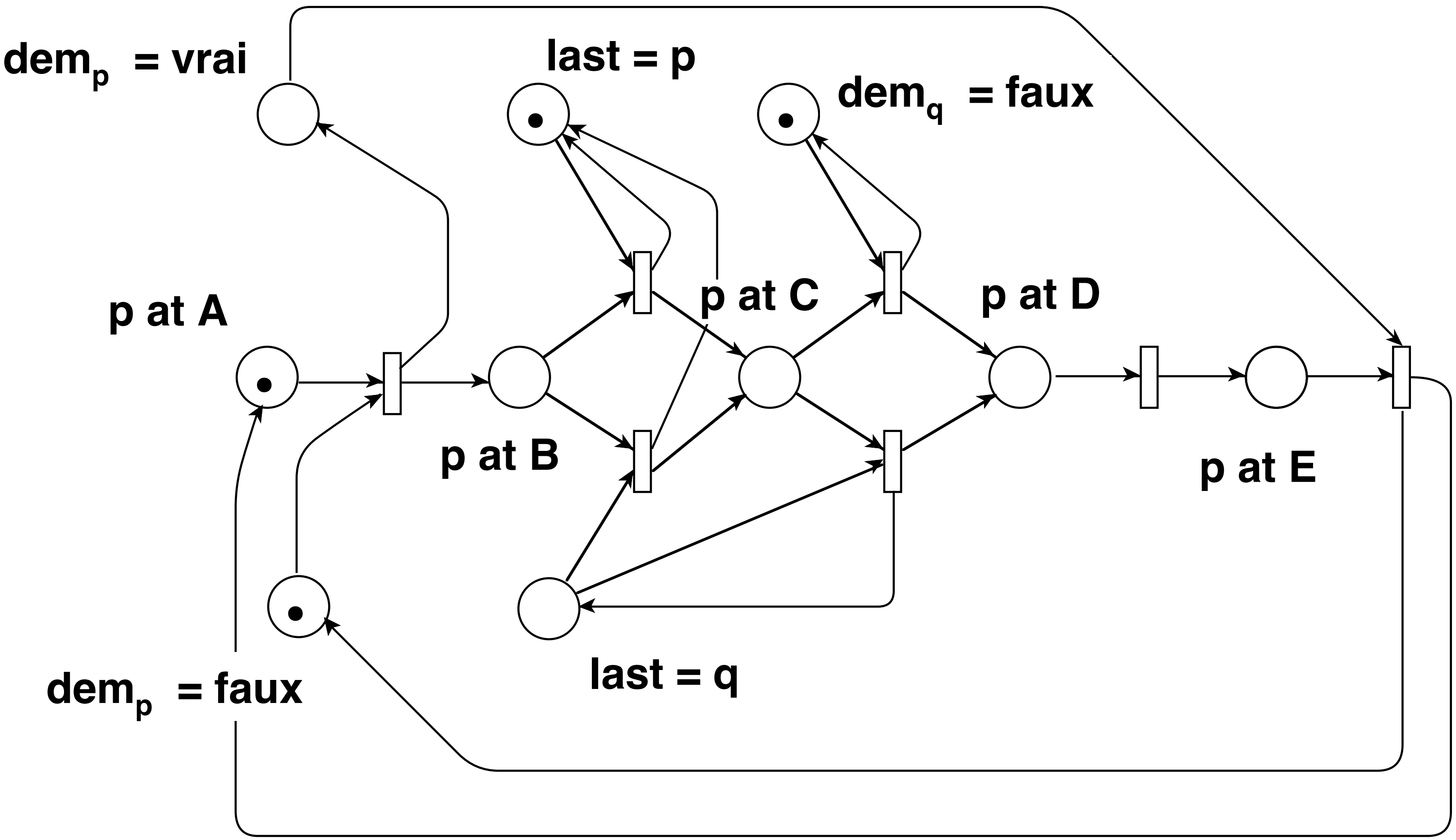
Sortie de section critique

```
D :    < Section critique >
E :    dem  = faux; goto A
```



Modèle global du processus p

On peut simplifier le modèle si on démontre la propriété : **p at A** $\Leftrightarrow$ **dem$_p$ = faux**



Exercice 1 : **Les trains**

Soit un rail circulaire de 6 sections sur lequel circulent deux trains dans le même sens et qui suivent les règles suivantes :

(i) Initialement le premier train est sur la première section et le deuxième sur la troisième section.

(ii) Un train ne pénètre dans une nouvelle section qui si ni cette section ni la suivante ne sont occupées par l'autre train.

1) Modéliser ce système à l'aide d'un réseau de Petri ordinaire.

2) Construire le graphe d'état et vérifier que la règle (ii) est bien respectée

Exercice 2 : **les philosophes**

- Soit une table circulaire sur laquelle sont posées cinq assiettes et cinq baguettes entre les assiettes. Cinq philosophes assis à cette table mangent et pensent alternativement. Pour manger les philosophes doivent s'emparer des deux baguettes de part et d'autre de leur assiette. Lorsqu'ils se remettent à penser, ils reposent leurs deux baguettes. Initialement tous les philosophes pensent.

- Les philosophes prennent d'abord la baguette gauche puis la droite. Modéliser ce système à l'aide d'un réseau de Petri.

Les philosophes (suite)

- Modifier votre modélisation pour que les philosophes prennent leurs baguettes l'une après l'autre, dans un ordre indifférent.
- Montrer que les cinq philosophes peuvent se retrouver simultanément en attente d'une baguette.
- Proposer une stratégie pour éviter l'interblocage et la modéliser.
- Exhiber sur ce dernier modèle une séquence infinie où quatre philosophes mangent alternativement sans que le cinquième ait jamais ses deux baguettes simultanément posées sur la table (sauf au départ).