NI405 - Modélisation des systèmes répartis

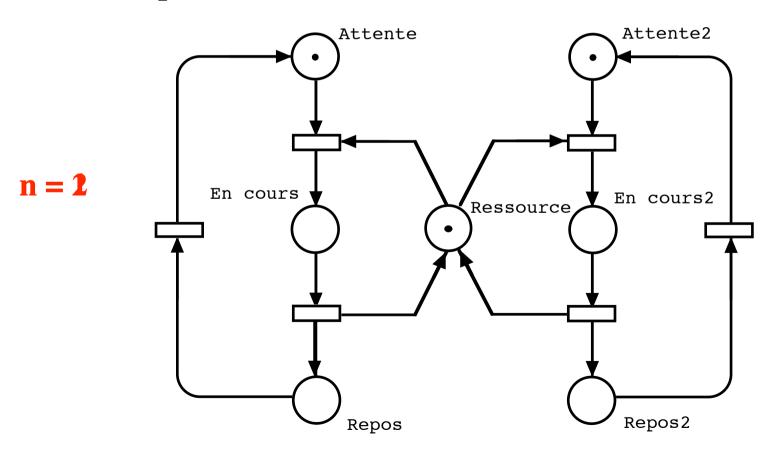
Introduction aux réseaux de Petri de haut niveau (les réseaux colorés)

1 - Modélisation

Pourquoi les réseaux de haut niveau?

• Problème:

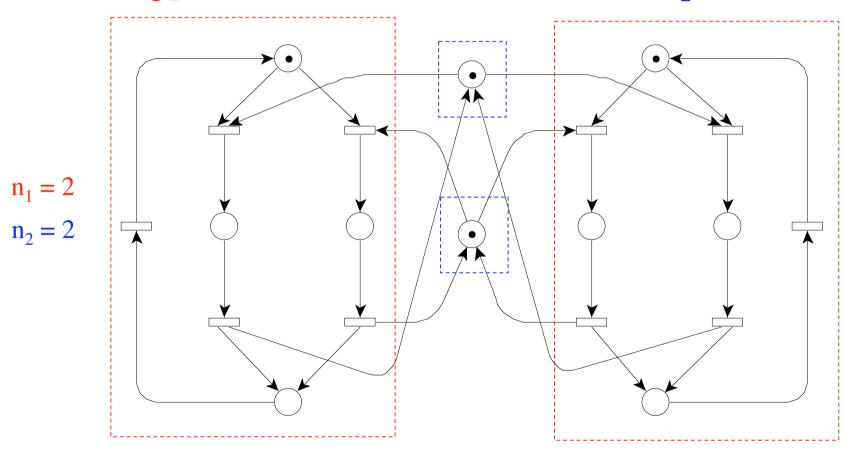
n processus en exclusion mutuelle sur 1 ressource



Pourquoi les réseaux de haut niveau?

• Problème 2:

n₁ processus en exclusion mutuelle sur n₂ ressources



Pourquoi les réseaux de haut niveau?

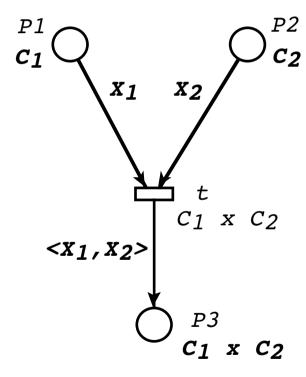
- Les réseaux de Petri ordinaires
 - ne capturent pas les symétries d'un problème
 - ne permettent pas d'associer des informations aux jetons
 - Ne permettent pas de paramétrer la solution d'un problème
- → Utiliser une notation concise et paramétrée des réseaux de Petri :

les réseaux de haut-niveau

Les réseaux de Petri colorés

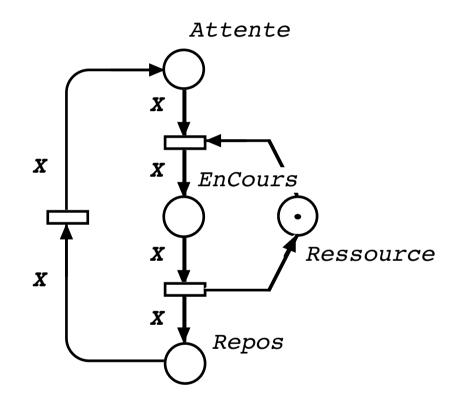
Définition informelle

- Chaque place p est caractérisée par un domaine de couleur C(p)
- Un jeton d'une place p est un élément de C(p)
- Chaque transition t est caractérisée par un domaine de couleur C(t)
- Le domaine de couleur d'une transition caractérise la signature de cette transition
- Les fonctions de couleur sur les arcs déterminent les instances de jetons nécessaires, consommées et produites lors du franchissement d'une transition



Un exemple

- n processus appartenant à une classe de processus C = {1, ..., n}, en exclusion mutuelle sur une ressource non signée
- Un processus est soit dans l'état Attente, soit dans l'état EnCours, soit dans l'état Repos.
- Pour passer de Attente à EnCours, un processus a besoin de la ressource.



$$C(Attente) = C(EnCours) = C(Repos) = C$$

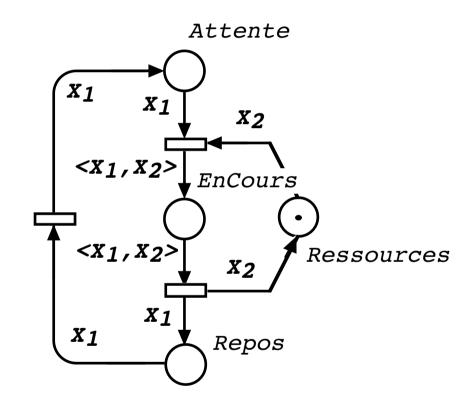
$$C(Ressource) = \{\epsilon\}$$

$$X : Bag(C) \rightarrow Bag(C)$$

$$c \rightarrow c$$

Un autre exemple

- n₁ processus appartenant à une classe de processus C₁ = {1, ..., n₁}, en exclusion mutuelle sur n₂ ressources appartenant à C₂ = {1, ..., n₂}
- Un processus X₁ est soit dans l'état Attente, soit dans l'état EnCours, soit dans l'état Repos.
- Pour passer de Attente à EnCours, un processus X₁ a besoin d'une ressource X₂.



$$C(Attente) = C(Repos) = C_1$$

$$C(EnCours) = C_1 \times C_2 \qquad M_0(Repos) = C_1 \cdot All$$

$$C(Ressource) = C_2 \qquad M_0(Ressource) = C_2 \cdot All$$

Multi-ensemble

- Soit A un ensemble fini et non vide.
- Un multi-ensemble a sur A est une fonction de A vers IN.
- On note

$$a = \sum_{x \in A} a(x) x$$

où a(x) désigne le nombre d'occurrences de x dans a.

• Bag(A) désigne l'ensemble des multi-ensembles sur A

Notations fonctionnelles

```
f_1: Bag(C_1) \rightarrow Bag(C_2)
f_2: Bag(C'_1) \rightarrow Bag(C'_2)
g: Bag(C) \rightarrow Bag(C_1)
\langle \mathbf{f}_1, \mathbf{f}_2 \rangle : \operatorname{Bag}(C_1) \times \operatorname{Bag}(C'_1) \rightarrow \operatorname{Bag}(C_2) \times \operatorname{Bag}(C'_2)
                                                  (x, y) \rightarrow \langle f_1(x), f_2(y) \rangle
 f_1 \circ g : Bag(C) \rightarrow Bag(C_2)
                            x \rightarrow f_1(g(x))
```

Notations vectorielles

- Soit F un vecteur sur A indexé par I $(F(i) \in A)$
- F(i) désigne la ième composante de F
- On note F par une somme formelle :

$$F = \sum_{i \in I} F(i)i$$

Définition formelle : la structure

• Un réseau de Petri coloré est un 6-uplet

$$\langle P, T, C, W^-, W^+, M_0 \rangle$$

- P est l'ensemble des places, T est l'ensemble des transitions $(P \cap T = \emptyset, P \cup T \neq \emptyset)$
- C définit pour chaque place et chaque transition son domaine de couleur
- W (= Pré) (resp. W = Post), indexée sur P x T, est la matrice d'incidence arrière (resp. avant) du réseau
- W⁻(p, t) et W⁺(p, t) sont des fonctions linéaires de couleurs définies de Bag(C(t)) dans Bag(C(p))

Compléments à la définition

- M_0 est le marquage initial du réseau ; c'est un vecteur indexé par P et $M_0(p)$ est un élément de Bag(C(p))
- Les transitions peuvent être gardées par une fonction $Bag(C(t)) \rightarrow \{0, 1\}$
- Les domaines de couleurs sont généralement des produits cartésiens

Définition formelle : la dynamique

Soit $CN = \langle P, T, C, W^-, W^+, M_0 \rangle$ un réseau coloré.

- Un marquage M de CN est un vecteur indexé par P, avec M(p) ∈ Bag(C(p))
- Une transition t est franchissable pour une instance $c_t \in C(t)$ et un marquage M si et seulement si :
 - Soit t est non gardée, soit la garde vaut 1 (= vrai) pour c_t
 - $\forall p \in P, M(p) \ge W^{-}(p, t)(c_t)$

La dynamique (suite)

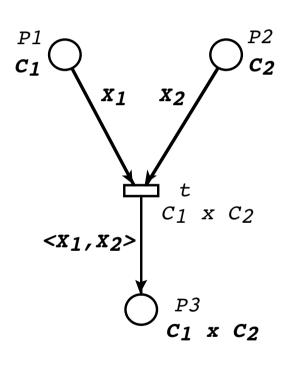
 Le marquage M' atteint après le franchissement de t pour une instance c_t à partir du marquage M est défini par :

$$\forall p \in P, M'(p) = M(p) - W'(p, t)(c_t) + W'(p, t)(c_t)$$

On note:
$$M[(t, c_t)>M']$$

$$M \stackrel{(t, c_t)}{\rightarrow} M'$$

Exemple de franchissement



• Soit $x_1 \in C_1, x_2 \in C_2$

• t est franchissable pour (x_1, x_2) ssi

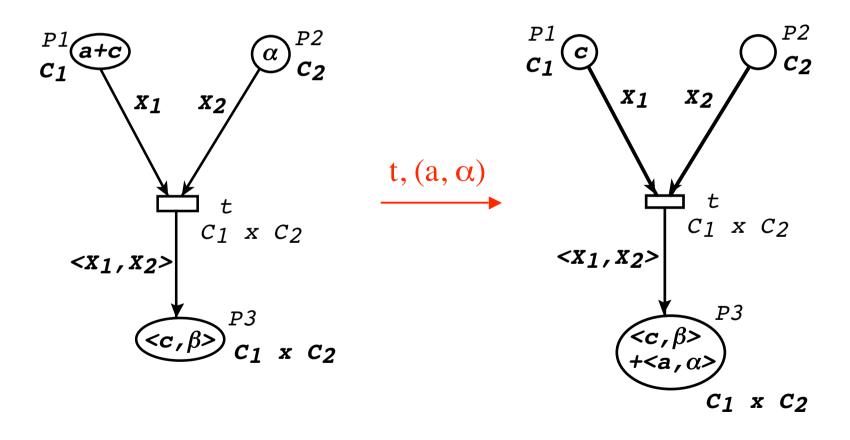
- 1) P1 contient au moins un jeton de couleur x_1 $(X_1(x_1, x_2) = x_1)$
- 2) P2 contient au moins un jeton de couleur x₂

• Si on franchit t pour (x_1, x_2) alors

- 1) Un jeton de couleur x₁ est retiré de P1
- 2) Un jeton de couleur x₂ est retiré de P2
- 3) Un jeton de couleur $\langle x_1, x_2 \rangle$ est produit dans P3: $\langle X_1, X_2 \rangle$ ($x_1, x_2 \rangle$) = $\langle x_1, x_2 \rangle$

Exemple (suite)

$$C_1 = \{a, b, c\}$$
 $C_2 = \{\alpha, \beta\}$



Fonctions de couleur de base

$$C = \prod_{i=1}^{n} \prod_{j=1}^{e_i} C_i$$

un domaine de couleur construit par produit cartésien des classes de couleurs, dans lequel la classe C_i apparaît e_i fois.

Dans la pratique, C est le domaine d'une transition.

$$c = \langle c_1^{\ l}, c_1^{\ 2}, ..., c_1^{\ el}, ..., c_n^{\ l}, c_n^{\ 2}, ..., c_n^{\ en} \rangle \in C$$

• Identité/Projection :

- Notée par une variable : X, Y, ou X_1 , ou X_1^1 , ou p, q, ...

$$X_i^j(c) = c_i^j \qquad Y(\langle x, y \rangle) = y$$
$$q(\langle p, q, r \rangle) = q$$

Fonctions (suite)

• Successeur (sur C_i ordonnée circulairement)

Notée
$$X_i$$
++ ou $(X_i \oplus 1)$ ou X_i !

$$X_i^j + +(c) = successeur(c_i^j)$$

La relation successeur est définie par l'ordre d'énumération des éléments de la classe C_i

• Diffusion / Synchronisation (sur C_i)

Notée C_i.All ou S_{Ci}

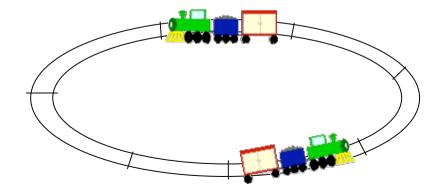
$$C_i. All(c) = \sum_{x \in C_i} x$$

La valeur est indépendante de l'élément choisi dans C_i

Modélisation du problème des trains

Problème:

- n₁ trains sont répartis sur une voie circulaire décomposée en n₂ sections
- Pour des raisons de sécurité, un train ne peut entrer dans une section que si cette section et la suivante sont libres (distance de sécurité)



Modèle(s)?

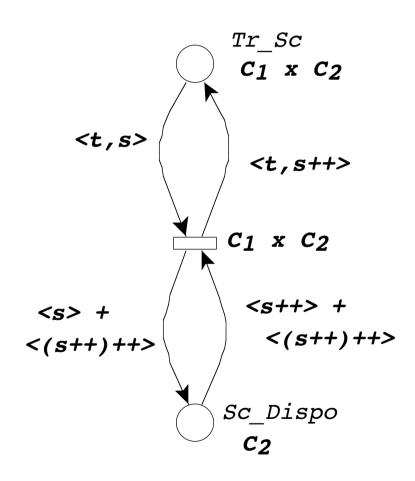
• Domaines de couleurs :

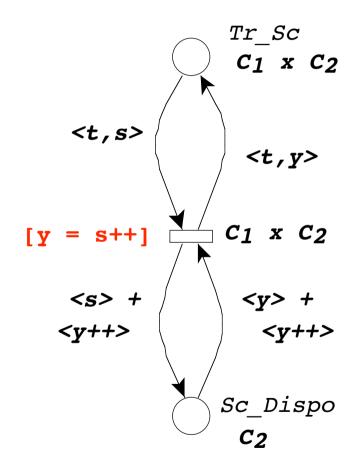
```
- C<sub>1</sub> = {tr_1, ..., tr_n<sub>1</sub>}
- C<sub>2</sub> = {sc_1, ..., sc_n<sub>2</sub>}
```

• Dynamique:

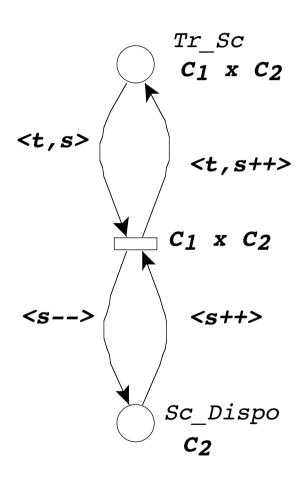
- L'état du système est donné par un ensemble d'associations <n° train, n° section> → place Tr_Sc
- Une section disponible est une ressource permettant le mouvement d'un train → place Sc_Dispo
- Une transition permet de représenter la progression d'un train

Des solutions...





Ou encore...



- s représente maintenant la section demandée
- Attention au marquage initial!

Dépliage de réseau

- Pour obtenir un réseau ordinaire ayant le même comportement que le réseau coloré
 - Pour chaque place ou transition, on crée autant d'instances que son domaine de couleur contient d'éléments
 - Les connexions sont obtenues en « dépliant » les fonctions de couleur
- Parfois le seul moyen d'obtenir des résultats sur le modèle
 - Mais on ne sait pas exprimer les résultats sur le modèle d'origine
 interprétation parfois difficile
- Facilement automatisable

Dépliage de réseau

Soit $CN = \langle P, T, C, W^-, W^+, M_0 \rangle$ un réseau coloré. Le réseau ordinaire sous-jacent (déplié) est le réseau

$$CN_d = \langle P_d, T_d, C_d, W_d^-, W_d^+, M_{0d} \rangle$$
 où

- $\mathbf{P_d} = \bigcup_{p \in P, c_p \in C(p)} (p, c_p)$ est l'ensemble des places
- $T_d = \bigcup_{t \in T, c_t \in C(t)} (t, c_t)$ est l'ensemble des transitions
- $M_{0d}(p, c_p) = M_0(p)(c_p)$ est le marquage initial

. . ./ . . .

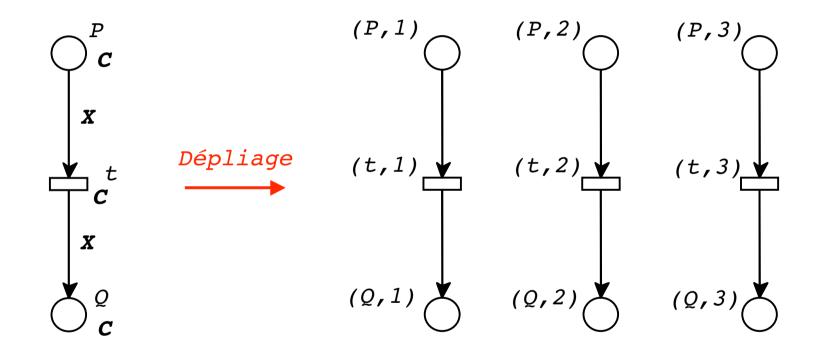
Dépliage (suite)

- $W_d^-(p, c_p)(t, c_t) = W^-(p, t)(c_t)(c_p)$ est la fonction d'incidence arrière
- $W_d^+(p, c_p)(t, c_t) = W_t^+(p, t)(c_t)(c_p)$ est la fonction d'incidence avant

Proposition:

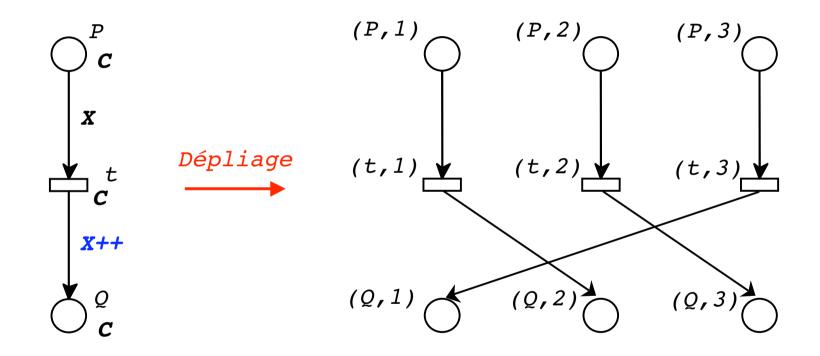
$$M [(t, c_t) >_{CN} M' \Leftrightarrow M_d [(t, c_t) >_{CNd} M'_d$$
où $M_d(\mathbf{p}, \mathbf{c}) = M(\mathbf{p})(\mathbf{c})$

Exemples de dépliage (1)



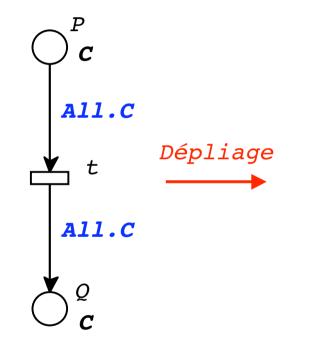
$$C = \{1, 2, 3\}$$

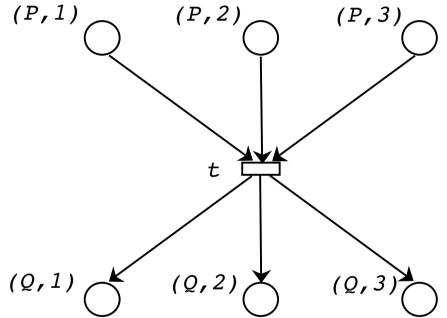
Exemples de dépliage (2)



$$C = \{1, 2, 3\}$$

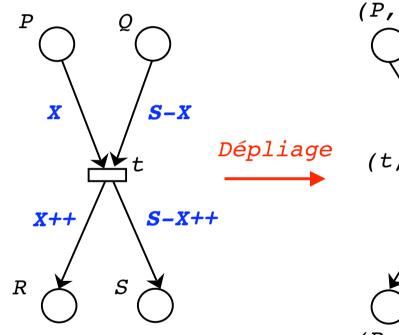
Exemples de dépliage (3)



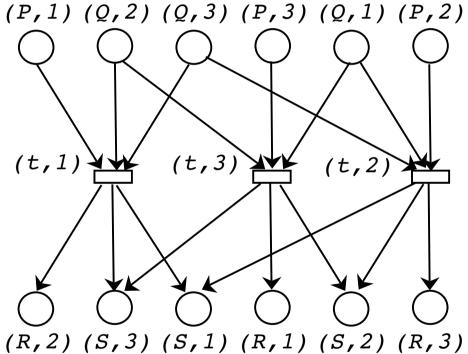


$$C = \{1, 2, 3\}$$

Exemples de dépliage (4)

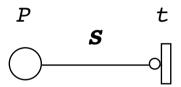


$$C = \{1, 2, 3\}$$
 $C(P) = C(Q) = C(R) = C(S) = C$



Arc inhibiteur coloré

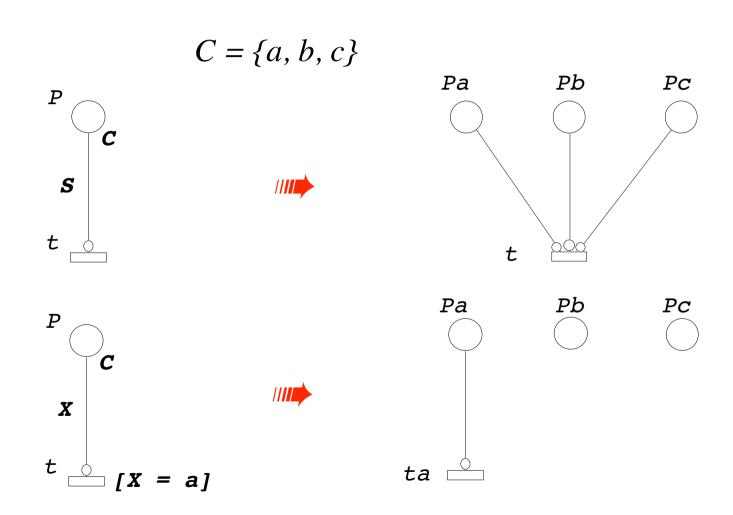
• Pour tester qu'une place est vide



• Pour tester qu'une place ne contient pas de jeton de couleur a

- Les ensembles de couleur sont finis
 - pas de modélisation du type lecteur/écrivain avec un nombre infini de lecteurs

Dépliage d'arc inhibiteur



Modélisation : les philosophes

- N philosophes sont assis autour d'une table et pensent.
- De temps en temps, ils mangent dans un bol de riz posé devant eux
- <u>Problème</u>: il n'y a que N baguettes, et chaque philosophe a besoin de 2 baguettes pour manger...

Version 1 : il y a N baguettes posées au milieu de la table.

Les philosophes (suite)

Version 2 : les baguettes sont disposées entre les philosophes. Pour manger, un philosophe doit prendre la baguette à sa gauche et celle à sa droite.

- a) Un philosophe prend ses 2 baguettes en même temps
- b) Il prend d'abord celle de droite, puis celle de gauche
- c) Il prend l'une puis l'autre

Modélisation : l'algorithme de Peterson

But: réaliser l'exclusion mutuelle de 2 processus au moyen de variables partagées

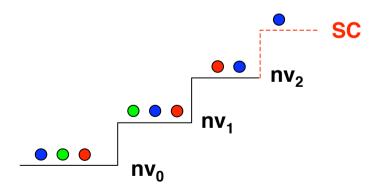
```
Processus x (x == 1 ou x == 2)
                                   Flag[x] == 1 : x demande la section
Flag[x] = 0;
                                   critique
While (1) {
                                   Turn: identité du dernier processus
  Flag[x] = 1;
                                   ayant demandé la section critique
  Turn = x;
  wait until
      ((Flag[x++] = 0) | | (Turn = x++));
  Section critique
  Flag[x] = 0;
```

Généralisation à N processus

• Principe:

- « escalier » à (N-1) niveaux
- Un processus peut passer du niveau j au niveau j+1 si :
 - Soit il n'est pas le dernier arrivé au niveau j,
 - Soit il est seul au niveau j et tous les niveaux supérieurs sont libres
- → Un seul processus peut aller au delà du niveau N-1
 - > section critique





L'algorithme pour N processus

```
Processus x (x == 1 ... N-1)
Flag[x] = 0;
While (1) {
  For (j=1; j<N; j++) {
       Flag[x] = j;
       Turn[j] = x;
       wait until
           ((\forall y \neq x, (Flag[y] < j)) \mid | (Turn[j] \neq x))
  Section critique
  Flag[x] = 0;
```