L'EXCLUSION MUTUELLE DANS UN ENVIRONNEMENT DISTRIBUÉ

SAIDOUNI Djamel Eddine

Université Constantine 2 - Abdelhamid Mehri
Faculté des Nouvelles Technologies de l'Information et de la Communication
Département d'Informatique Fondamentale et ses Applications

Laboratoire de Modélisation et d'Implémentation des Systèmes Complexes

Djamel.saidouni@univ-constantine2.dz
saidounid@hotmail.com
Tel: 0559082425

ALGORITHME DE CHANDY ET MISRA

ALGORITHME DE CHANDY ET MISRA

But de l'algorithme: Concevoir un algorithme d'exclusion mutuelle:

1) Qui soit adaptatif: Un site qui a donné ses permissions et qui n'est pas intéressé par l'accès à la SC ne doit pas être sollicité.
2) Qui n'utilise que des variables de taille bornée.

Remarque:

- L'algorithme de Ricart et Agrawala peut satisfaire la contrainte (2) si on implémente les horloges modulo $2 * n - 1$ et que celui de Carvalho et Roucairol satisfait la contrainte (1).
- Les seules variables non bornées dans l'algorithme de Carvalho et Roucairol sont les horloges logiques. Ces dernières sont utilisées pour définir la priorité entre les requêtes des processus demandeurs.
- Et si on pouvait s'en passer des horloges logiques pour ordonner les requêtes tout en gardant le principe des permissions utilisé dans l'algorithme de Carvalho et Roucairol ? Cela donnera un algorithme adaptatif avec des variables bornées.

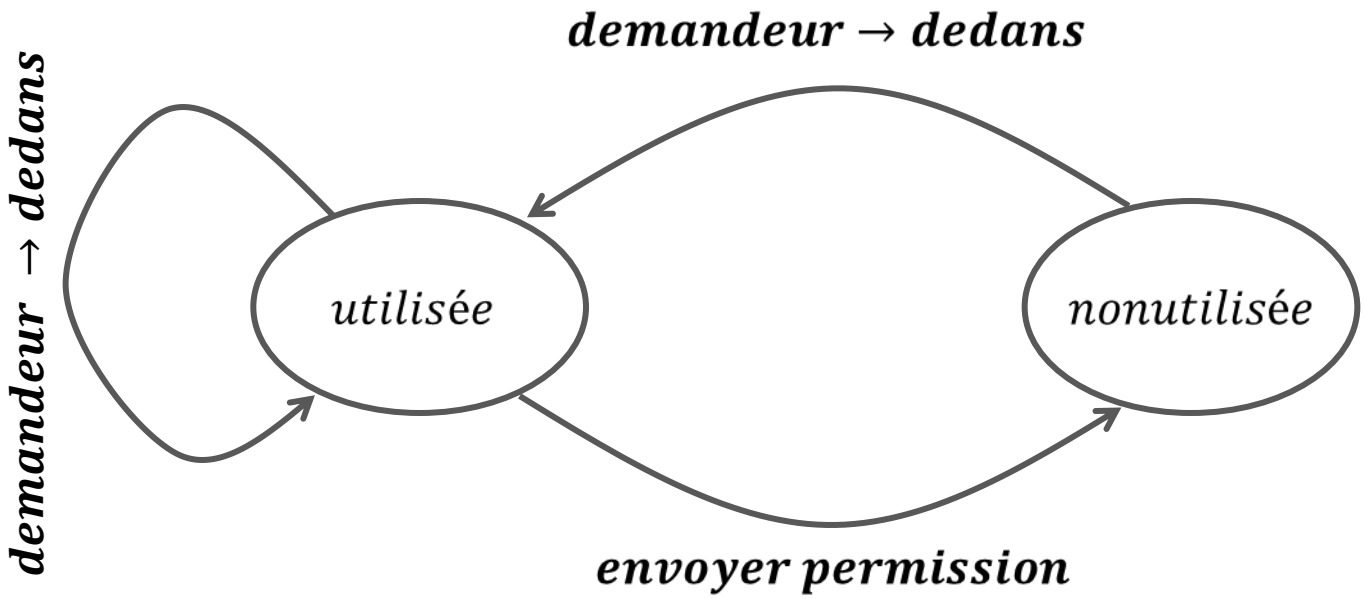
ALGORITHME DE CHANDY ET MISRA

Principe:

- Associer un état à toute permission $perm_{ij}$ liant deux processus P_i et P_j.
- L'état d'une permission est soit $utilisée$ soit $nonutilisée$.
- Si le processus P_i reçoit la permission $perm_{ij}$, l'état de celle-ci est $nonutilisée$. Son état devient et reste $utilisée$ dès qu'il rentre dans sa section critique.
- Si le processus P_i détient la permission $perm_{ij}$, reçoit la requête $requête(j)$ alors :
  - Il se déclare prioritaire s'il est dedans ou bien demandeur et l'état de la permission $perm_{ij}$ est $nonutilisée$. Dans ce dernier cas, cela signifie qu'il n'a pas encore utilisé cette permission pour rentrer dans sa section critique depuis qu'il l'a demandée, il cédera pas donc la permission.
  - Il se déclare non prioritaire et cédera cette permission dans les autres cas.

ALGORITHME DE CHANDY ET MISRA

Règles de changement d'état d'une permission: Elles sont définies par l'automate suivant.



A l'état initial toutes les permissions sont dans l'état $utilisée$

L'ALGORITHME

Variables locales pour un processus P_i

- $\boldsymbol{R_i} = \{j \mid le\ site\ i\ ne\ possède\ pas\ perm_{ij}\}$
- $\boldsymbol{Etat_i}: \{dehors, demandeur, dedans\}$
- $\boldsymbol{Priorité_i}: Booléen$
- $\boldsymbol{Différés_i}: ensemble\ de\ sites\ init$ à $\emptyset$

L'ALGORITHME

PROCÉDURES DU PROCESSUS P_i

Lors d'un appel à acquérir

$Etat_i = demandeur$;

$\forall j \in R_i$: ***envoyer*** $requête(i)$ à j ;

attendre$(R_i = \emptyset)$;

$Etat_i = dedans$;

$\forall j \in \{1..n\} - \{i\}: perm_{ij}.état = utilisée$;

L'ALGORITHME

PROCÉDURES DU PROCESSUS P_i (SUITE)

Lors d'un appel à libérer

$$Etat_i = dehors\ ;$$

$$\forall j \in Différés_i : \boldsymbol{envoyer}\ perm_{ij}\ \text{à}\ j\ ;$$

$$R_i = Différés_i\ ;$$

$$Différés_i = \emptyset\ ;$$

Lors de la réception de $\boldsymbol{permission(j)}$

$$R_i = R_i - \{j\}$$

L'ALGORITHME

PROCÉDURES DU PROCESSUS P_i (SUITE)

Lors de la réception de $requête(j)$

Si $j \in R_i$ **Alors** $Priorité_i = True$ * Le message $perm_{ij}$ est en transit vers P_i

Sinon

$$Priorité_i = \begin{cases} Etat_i = dedans \\ \textbf{ou} \\ (Etat_i = demandeur) \textbf{ et } perm_{ij}.état = nonutilisée \end{cases}$$

Finsi

Si $Priorité_i$ **Alors** $Différés_i = Différés_i \cup \{j\}$

Sinon { ***envoyer*** $perm_{ij}$ à j ;

$R_i = R_i \cup \{j\}$;

Si $état_i = demandeur$ **Alors** ***envoyer*** $requête(i)$ à j **Finsi** }

Finsi

PREUVE DE L'ALGORITHME

Sûreté: L'unicité du jeton liant chaque deux sites résout le conflit entre ces deux sites, d'où la vérification de la propriété d'exclusion mutuelle.

Vivacité: Elle dépend des conditions initiales précises qui définissent un graphe acyclique. Ce sont ces conditions initiales qui sont le germe à partir duquel une asymétrie peut être maintenue.

Graphe des priorités: $i \rightarrow j : i$ n'est pas prioritaire sur j ssi la permission partagée entre $i \; et \; j$ est:

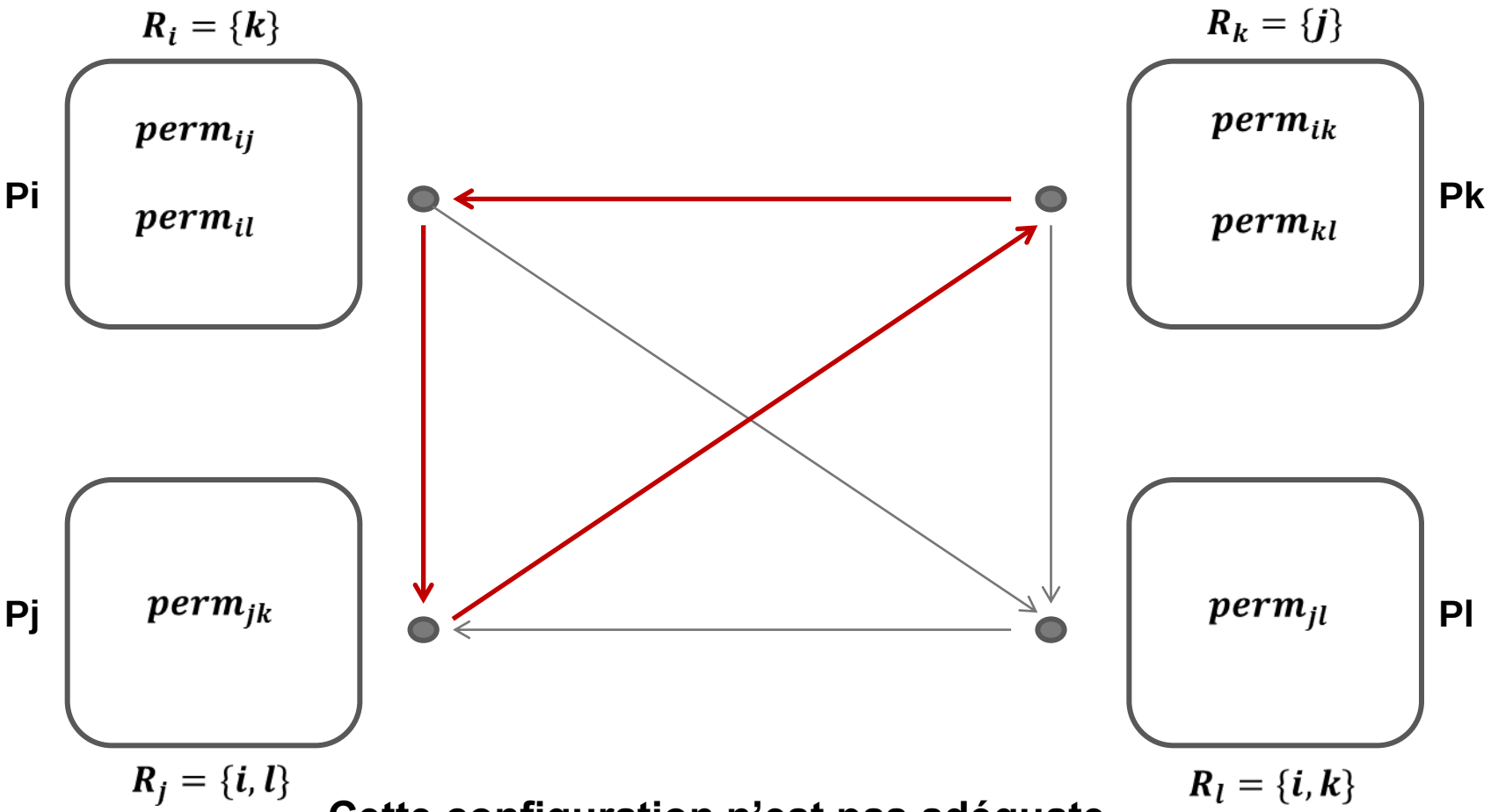
i. Sur le site i et dans l'état $utilisée$

ii. Ou en transit de $i \; vers \; j$

iii. Ou sur le site j et dans l'état $nonutilisée$

ALGORITHME DE CHANDY ET MISRA

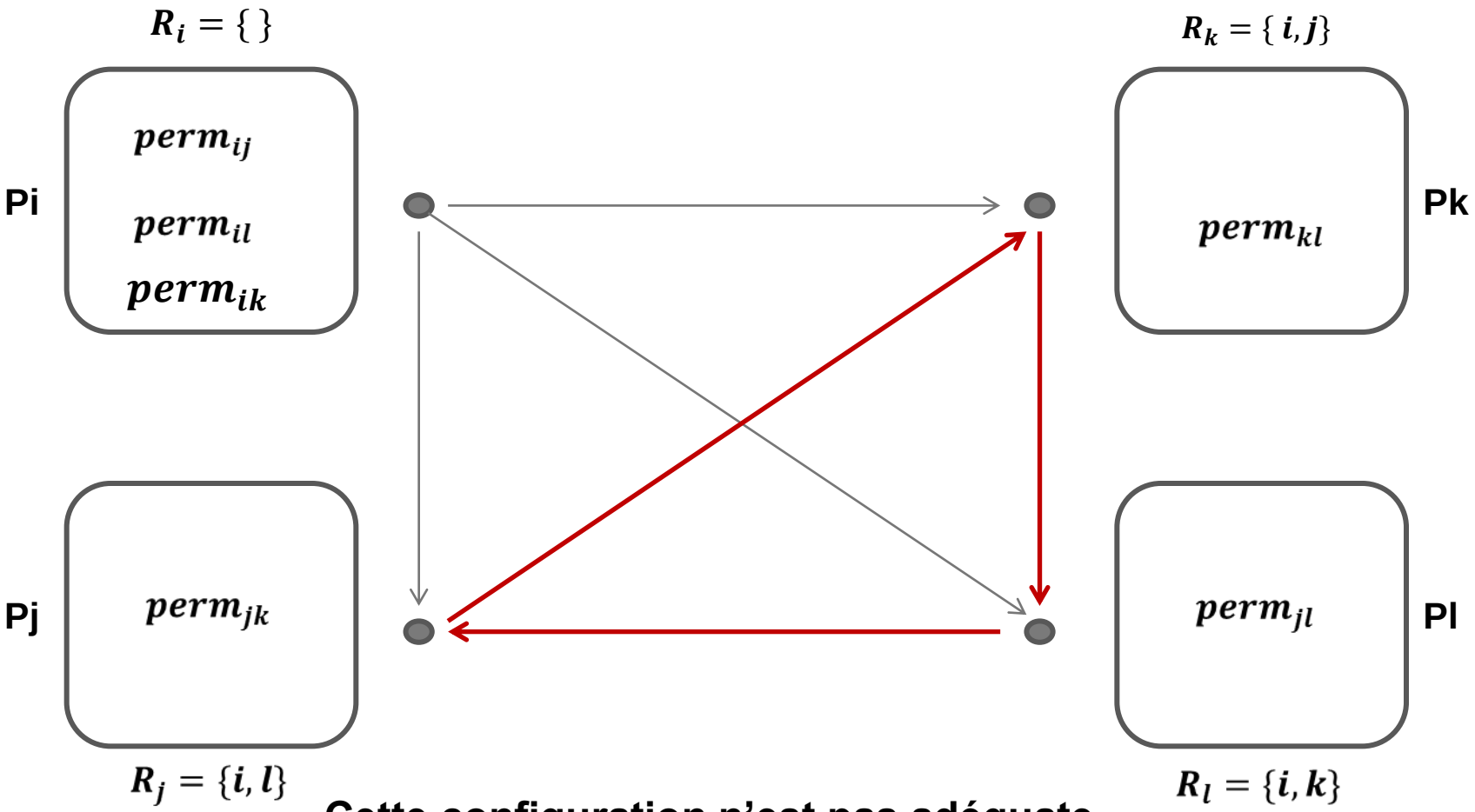
EXEMPLE ILLUSTRATIF



Cette configuration n'est pas adéquate

ALGORITHME DE CHANDY ET MISRA

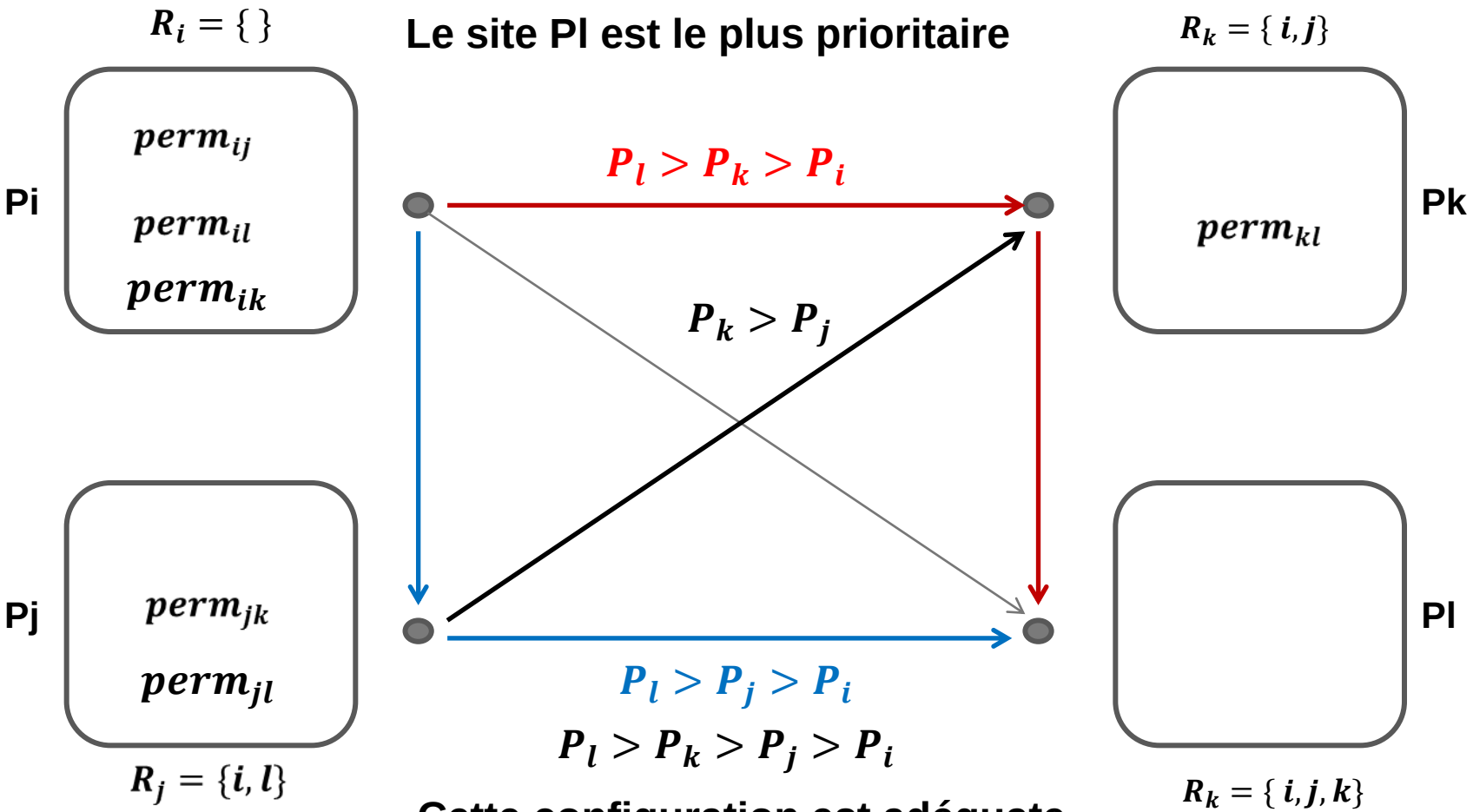
EXEMPLE ILLUSTRATIF



Cette configuration n'est pas adéquate

ALGORITHME DE CHANDY ET MISRA

EXEMPLE ILLUSTRATIF



PREUVE DE L'ALGORITHME

Propriété: Le graphe reste acyclique.

Preuve: Seul le changement d'état d'une permission ou son envoi peuvent à priori changer le sens de l'arc correspondant. Or, lors de l'envoi d'une permission, celle-ci passe de l'état ***utilisée*** à l'état ***nonutilisée***, ce qui ne change pas le sens de l'arc associé.

Un arc ne change donc de sens que lorsque la permission associée passe de l'état ***nonutilisée*** à l'état ***utilisée*.** C'est-à-dire lorsque le site qui effectue ce changement passe de l'état ***demandeur*** à l'état ***dedans***. Ce site modifiera alors tous les arcs entrants. Ceci ne peut créer un cycle qui passerait par ce site. G reste donc acyclique.

PREUVE DE L'ALGORITHME

Propriété: La propriété de vivacité est vérifiée.

Preuve: Appelons hauteur d'un site i dans G la longueur maximale d'un chemin allant de i au sommet sans successeur. On montre par induction sur k, la hauteur d'un site dans l'état $demandeur$, que ce site passera à l'état $dedans$ si tous les sites dans l'état demandeur est de hauteur inférieure à k y passent. Soit i le site de hauteur k.

- **Hauteur $k = 0$** : Dans ce cas le site i n'a que des arcs entrants. D'après la sémantique des arcs, les requêtes du site i finiront par être satisfaites selon les procédures de réception d'une requête et de l'opération libérer. Il passera donc à l'état dedans.

PREUVE DE L'ALGORITHME
VIVACITÉ (SUITE)

- **Hauteur k (induction)**: Pour les arcs entrants le site i a ou obtiendra les permissions correspondantes. Pour chacun des arcs sortants (i, j) le site j est de hauteur $\leq (k-1)$ et la permission associée est dans l'une des configurations (i), (ii) ou (iii).
  - **Cas (i):** Si la permission est toujours sur le site i lorsqu'il a obtenu toutes les autres, le problème est réglé (il accède à sa SC).
  - **Cas(ii) et (iii):** le site j est dans l'état demandeur, étant donné qu'il reçoit une permission dans l'état $nonutilisée$; comme sa hauteur est $\leq (k-1)$, l'hypothèse d'induction implique qu'il pénétrera dans sa SC. Quant il en sortira il changera l'orientation des arcs entrants. Le site i finira par se trouver à une hauteur $\leq (k-1)$, d'où le résultat.