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

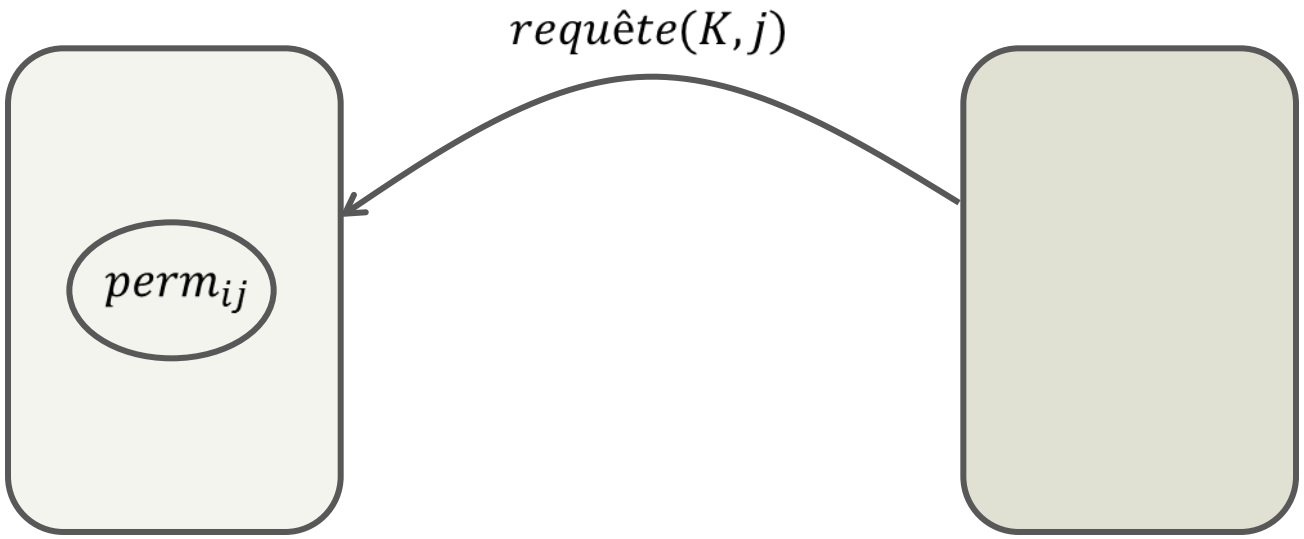
Djamel.saidouni@univ-constantine2.dz
saidounid@hotmail.com
Tel: 0559082425

ALGORITHMES A PERMISSIONS INDIVIDUELLES

ALGORITHME DE CARVALHO ET ROUCAIROL

ALGORITHME DE CARVALHO ET ROUCAIROL

PRINCIPE



Si P_i désire rentrer plusieurs fois avant que P_j ne devient demandeur, il ne demande que les permissions qui lui manquent

P_i

Le message $\boldsymbol{perm_{ij}}$ est similaire à un **jeton** qui lie les processus P_i et P_j.

P_j

A son tour P_j peut rentrer plusieurs fois dans sa SC sans demander la permission de P_i, cela jusqu'à ce que ce dernier devient demandeur.

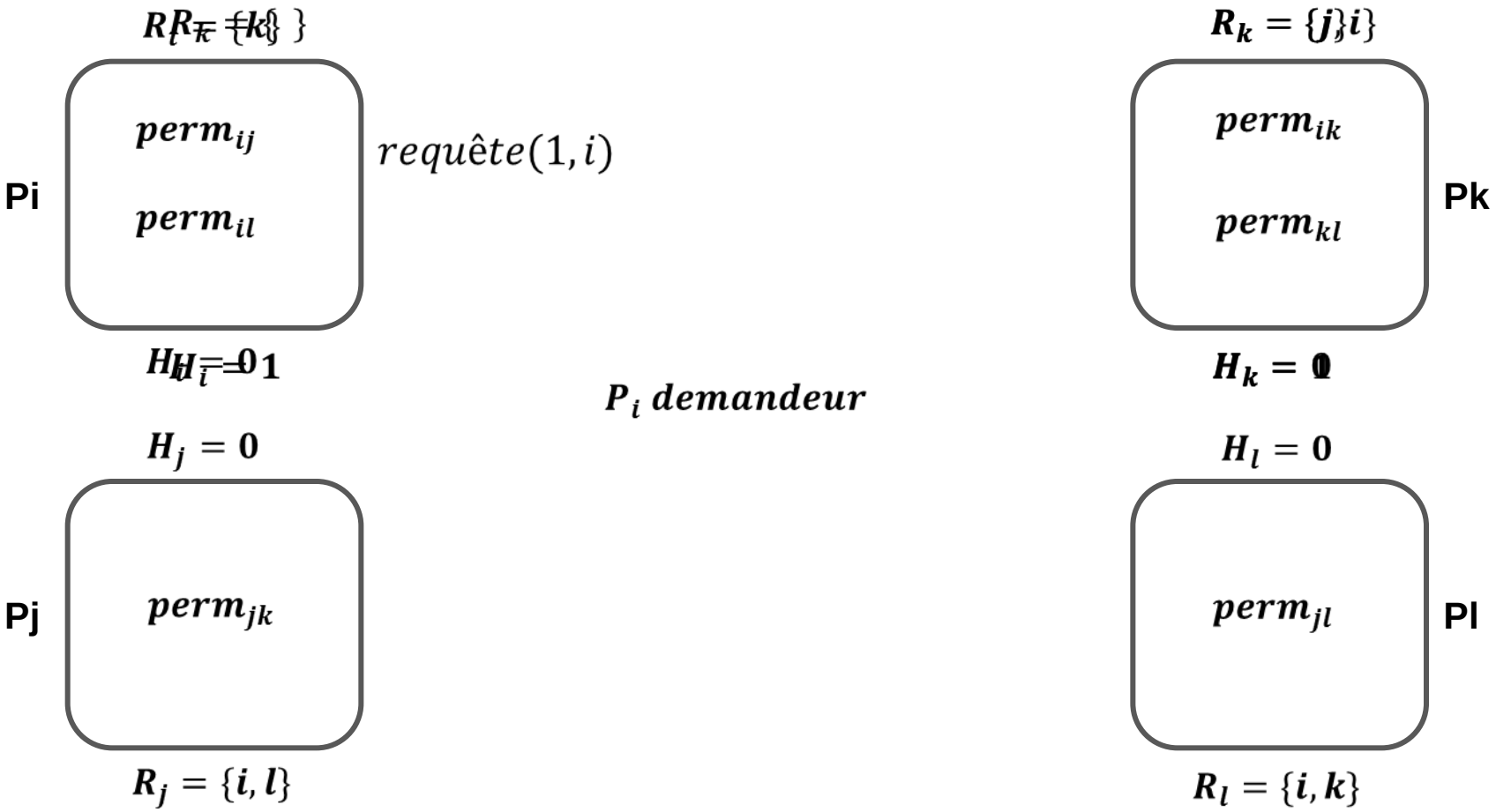
ALGORITHME DE CARVALHO ET ROUCAIROL

MATÉRIALISATION

- A tout couple de sites distincts i et j on associe un message $\boldsymbol{perm_{ij}}$.
- Le message $perm_{ij}$ est soit sur le site i soit sur le site j soit en transit entre les deux sites.
- Initialement les permissions sont placées sur les sites en respectant les conditions précédentes.
- Un site demandeur d'accès à la SC ne demande que les permissions qui lui manquent.
- De ce fait $\boldsymbol{R_i = \{j \mid le\ site\ i\ ne\ possède\ pas\ perm_{ij}\}}$
- Pour cet algorithme $\forall \boldsymbol{i \neq j, i \in R_j\ ou\ exclusif\ j \in R_i}$
- Dans ce qui suit $\boldsymbol{perm_{ij} \Leftrightarrow perm_{ji}}$

ALGORITHME DE CARVALHO ET ROUCAIROL

EXEMPLE ILLUSTARTIF



L'ALGORITHME

Variables locales pour un processus P_i

- $\boldsymbol{R_i} = \{j \mid le\ site\ i\ ne\ possède\ pas\ perm_{ij}\}$
- $\boldsymbol{Etat_i}: \{dehors, demandeur, dedans\}$
- $\boldsymbol{H_i}: entier\ croissant\ init$ à 0
- $\boldsymbol{Last_i}: entier\ croissant\ init$ à 0
- $\boldsymbol{Priorité_i}: Booléen$
- $\boldsymbol{Différés_i}: ensemble\ de\ sites\ init$ à $\emptyset$

L'ALGORITHME

PROCÉDURES DU PROCESSUS P_i

Lors d'un appel à acquérir

$Etat_i = demandeur$;

H_i++ ;

$Last_i = H_i$;

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

attendre$(R_i = \emptyset)$;

$Etat_i = dedans$;

L'ALGORITHME

PROCÉDURES DU PROCESSUS P_i (SUITE)

Lors d'un appel à libérer

$Etat_i = dehors$;

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

$R_i = Différés_i$;

$Différés_i = \emptyset$;

Lors de la réception de $permission(j)$

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

$Attendus_i = Attendus_i - \{j\}$;

L'ALGORITHME

PROCÉDURES DU PROCESSUS P_i (SUITE)

Lors de la réception de $requête(K, j)$

$H_i = Max(H_i, K)_i$;

$Priorité_i = (Etat_i = dedans)\ ou\ \big((Etat_i = demandeur)\ et\ (Last_i, i) < (K, j)\big)$;

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*** $requête(Last_i, i)$ à j **Finsi**

Finsi

PROPRIÉTÉS DE L'ALGORITHME

Propriétés fonctionnelles de l'algorithme:

- **Sûreté :** Assurée par l'unicité du jeton liant chaque deux sites.
- **Vivacité :** Comme dans l'algorithme de Ricart et Agrawala, l'estampillage des requête crée un ordre total entre les requêtes. Ce qui permet d'assurer la vivacité.

Propriétés de performance de l'algorithme:

- $0 \leq nombre\ de\ messages\ de\ contrôle \leq 2 * (n-1)$
- $0 \leq Temps\ SC\ libre\ et\ \exists\ demandeurs \leq 2 * T$
- L'algorithme est adaptatif.
- Les variables ne sont pas bornées.