Détection et prévention de l'interblocage dans les systèmes répartis

Babahenini Med Chaouki

Partage des ressources et interblocage

- Interblocage :

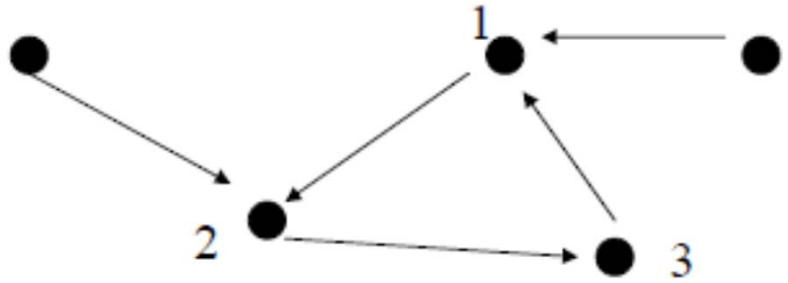
  chaque processus d'un ensemble attend l'occurrence d'un événement qui ne peut être produit que par un autre processus de ce même ensemble.
- Exemple : allocation de trois ressources R1, R2 et R3 à trois processus P1, P2 et P3 dont chacun désire utiliser deux ressources simultanément en accès exclusif.
- Formalisation de l'interblocage :
  - modèle de la théorie des graphes
  - un processus est associé un sommet du graphe,
  - l'attente par un processus Pi d'un événement qui doit être produit par Pj, est matérialisée par un arc allant de Pi vers Pj.
- Ce graphe, appelé graphe des attentes, traduit à un instant donné, les relations de blocage entre les processus.

Caractérisation des interblocages
Modèle basé sur l'allocation de ressources

- Une SR est constituée de N sites S1, S2, … SN.
- Un processus local sur chaque site Sj, le contrôleur Cj , ordonnance les processus, gère les ressources, et met en œuvre les émissions/réceptions de messages.
- M transactions T1,… TM s'exécutent sur le SR.
- Une transaction est mise en œuvre par un ensemble de processus (au plus un par site).
- Pendant le déroulement de la transaction il peut être nécessaire de verrouiller des ressources (par exemple des fichiers).
- Un processus ne peut s'exécuter que lorsqu'il a obtenu toutes les ressources qu'il avait demandées.

Caractérisation des interblocages
Modèle basé sur l'allocation de ressources

- *p1 attend une ressource de p2 qui attend p3 qui attend p1*



Graphe des processus en attente

- *Détecter un interblocage revient à détecter une boucle dans le graphe des processus en attente.*

Caractérisation des interblocages
Modèle basé sur la communication de messages

- Dans ce cas il n'y a pas de contrôleurs ou ressources explicites.
- Les requêtes d'allocation et libération de ressources peuvent être implémentées par des échanges de messages.
- L'interblocage peut se produire dans un ensemble de processus communicants à cause des échanges de messages.

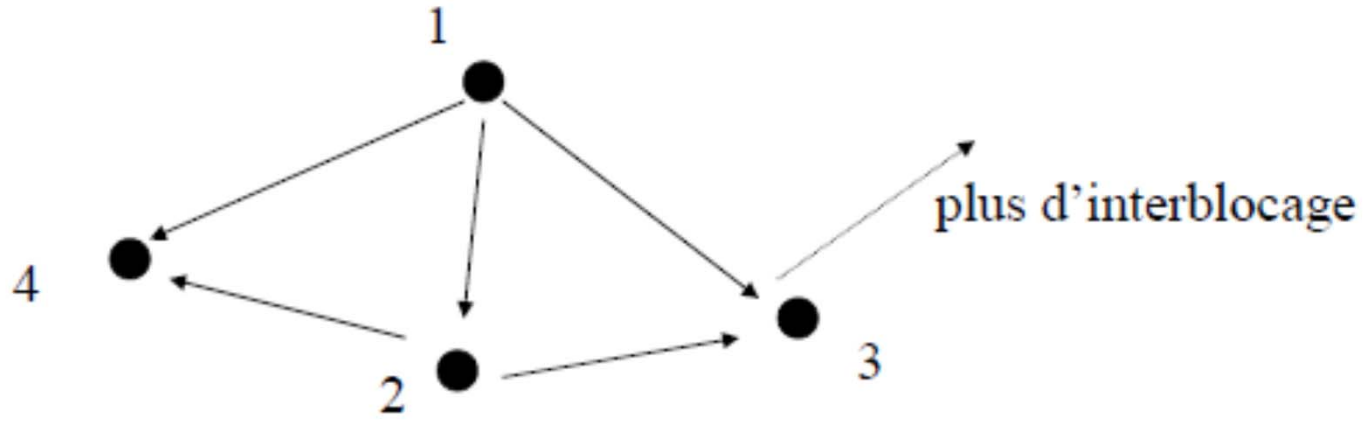
  Exemple : cas où chaque processus attend un message d'un autre processus alors qu'aucun message n'est en transit.

Caractérisation des interblocages
Modèle basé sur la communication de messages

- L'interblocage d'un ensemble S de processus communicants peut se caractériser de la façon suivante :
  - tous les processus de S sont passifs (en attente de messages)
  - quel que soit le processus de S, son ensemble de dépendance est inclus dans S
  - il n'y a pas de messages en transit entre les processus de S.
- Un processus Pi appartenant à un ensemble S satisfaisant ces trois conditions est interbloqué
- En termes de graphes :
  - l'attente par Pi d'un message en provenance de Pk ou Pl ou … se traduit par autant d'arcs orientés du sommet Pi vers les sommets Pk et Pl…
  - l'émission d'un message de Pk vers Pi et sa réception aura pour effet de supprimer tous les arcs partant de Pi.

Caractérisation des interblocages
Modèle basé sur la communication de messages

- *p1 attend un message de p2 ou de p3 ou de p4 ….*
- *p2 attend un message de p4 ou de p3*



- *détecter un interblocage revient à détecter :*
  - *1) que le graphe des attente est une composante fortement connexe*
  - *2) qu'il n'y a aucun message en transit entre les processus du graphe*

Comparaison des interblocages relatifs aux ressources et aux messages

- *l'agent de détection de l'interblocage n'est pas le même dans les deux environnements* :
  - dans le modèle basé sur les messages, un processus connaît l'identité des processus dont il attend les messages

    ⇒ les **processus** ont suffisamment d'informations pour entreprendre une détection d'interblocage s'ils agissent collectivement.
  - dans le modèle basé sur les ressources la dépendance d'une transaction vis-à-vis d'une autre transaction n'est pas directement connue :

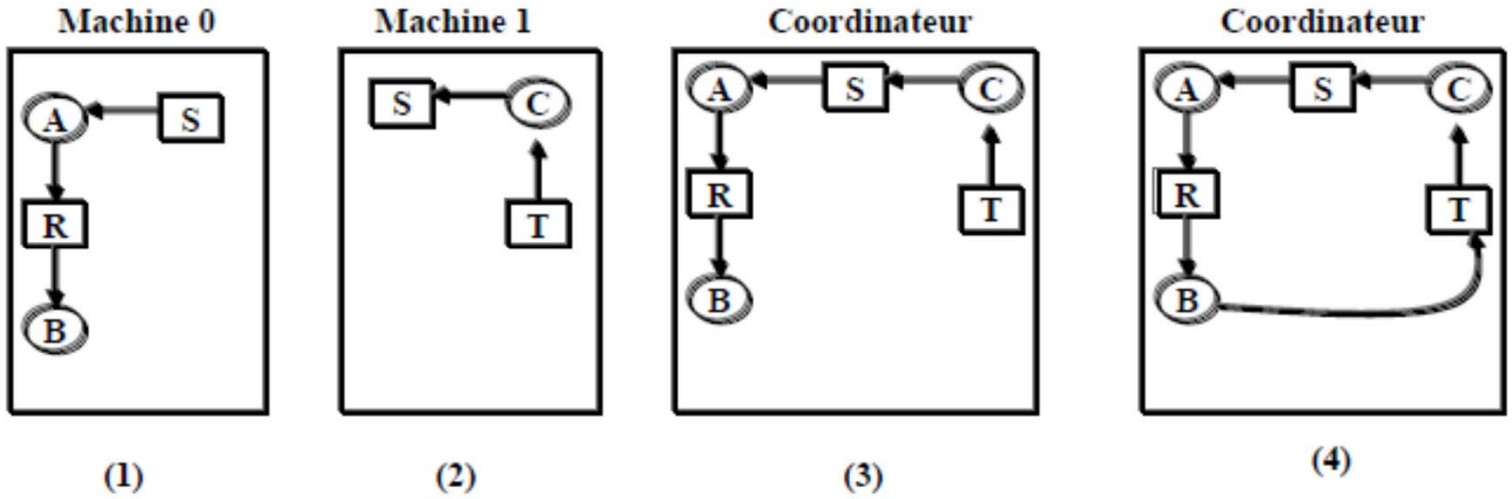
    ⇒ Seul le **contrôleur** sur chaque site peut garder la trace de l'utilisation de ses ressources et déduire qu'une transaction est en attente.

Trois stratégies contre l'interblocage

- Prévention : contraintes sur la manière de demander l'accès aux ressources :
  - Schéma 1 : réservations de toutes les ressources avant de commencer
  - Schéma 2 : relâchement de toutes les ressources avant d'accéder à de nouvelles
  - Schéma 3 : ordonnancement des ressources et accès dans leur ordre
- Évitement :
  - Contrôle pas à pas de la progression des processus
  - Mise en œuvre d'une possibilité d'échapper à un interblocage
  - Dépendant des algorithmes des utilisateurs ⇒ non étudié ici
- Détection puis résolution :
  - Interblocage est une propriété stable ⇒ algorithme de détection
    - Non-interférence : ne pas influencer l'exécution de l'algorithme de l'utilisateur
    - Vivacité (en anglais, liveness) : si interblocage, le détecter en un temps fini
    - Correction (en anglais, safety) : détecter un interblocage seulement lorsqu'il en existe un
  - Résolution : dépendant des algorithmes des utilisateurs

Détection des interblocages : Algorithme centralisé

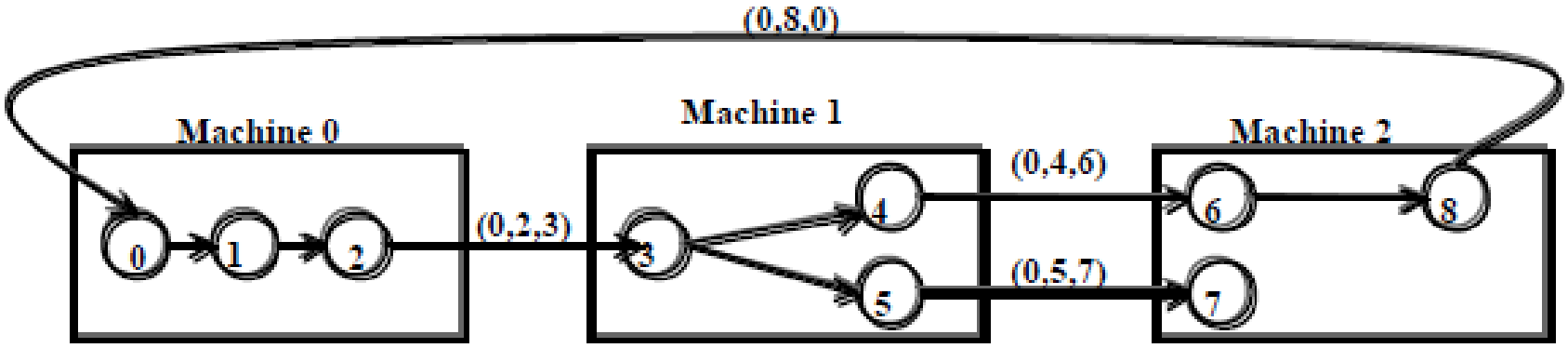
- Algorithmes fondamentalement centralisés insuffisants



- (5) Si en 3 ’, processus B avait relaché R, machine 0 envoie un message au coordinateur, qui l ’obtient en 5.
- Si en 3 ’ ’ processus B avait alors demandé T, machine 1 envoie un message au coordinateur qui l ’obtient en 4.
- => détection de faux interblocages pouvant entrainer « mort injuste » pour sa guérison

Détection des interblocages : algorithmes distribués

- Ex: P0 se bloque sur P1 (P0 demande une ressource que P1 a verrouillé)
- => une sonde (0, 0, 1) est émise , elle sera transmise de site en site.
- Champ1: num du processus qui se bloque; Champ2: emetteur de la sonde; Champ3: recepteur de la sonde.



- Algorithme de Chandy-Misra-Haas (1983):
- Principe d ’une sonde qui va parcourir le graphe des attentes. Si la sonde revient, alors, la nouvelle attente vient d ’engendrer un interblocage.

Détection des interblocages : algorithmes distribués

- Guérison ? Abandonner la transaction «fautive», celle ayant le moins de verrous, la plus jeune ?
- Détection/Guérison plus simple par Temporisation:
  - Fixer durée max d ’exécution pour une transaction
  - Si au bout du délai une transaction n ’a pas fini, suspicion qu’elle est interbloquée et donc abondon de la transaction

Prévention des interblocages: Principe

- La prévention des interblocages consiste à construire prudemment le système de façon de rendre les interblocages structurellement impossibles. Autrement dit, elle consiste à supprimer l ’une des conditions qui rend possible l’interblocage
  - Contraintes d ’allocation:
    - un processus ne détient qu’une ressource à la fois
    - un processus doit déclarer toutes les ressources dont il a besoin
    - un processus doit relâcher les ressources qu’il détient pour pouvoir en obtenir une nouvelle

      => mal adapté pour verrouillage en 2 phases !
  - Ordonnancement des ressources:
    - préordonner toutes les ressources du système
    - un processus doit acquérir les ressources dont il a besoin dans l’ordre strictement croissant.
  - Ordonnancement des transactions
    - préordonner toutes les transactions par une estampille
    - déterminer une politique d’allocation des ressources adéquate permettant d’éviter les interblocages

Prévention des interblocages: Principe
Algorithmes de Rosenkrantz, Stearns et Lewis

- L'idée de l'algorithme de Rosenkrantz, Stearns et Lewis est de prévenir l'interblocage en basant uniquement sur l'estampillage.
- Principe : lorsqu'une ressource r, utilisée par une transaction T1, est demandée par une transaction T2, les conflits sont résolus par comparaison de leurs estampilles.
- Il n'y a ici aucune annonce préalable : l'algorithme proposé peut donc s'appliquer lorsque les *accès aux ressources sont définis dynamiquement*.
- Il s'agit toujours d'un algorithme de prévention : la comparaison des estampilles a pour effet d'empêcher la formation de circuits dans le graphe des conflits.
- Deux techniques sont proposées : avec ou sans réquisition.

Algorithmes de prévention

Technique sans réquisition : « Wait-Die System»

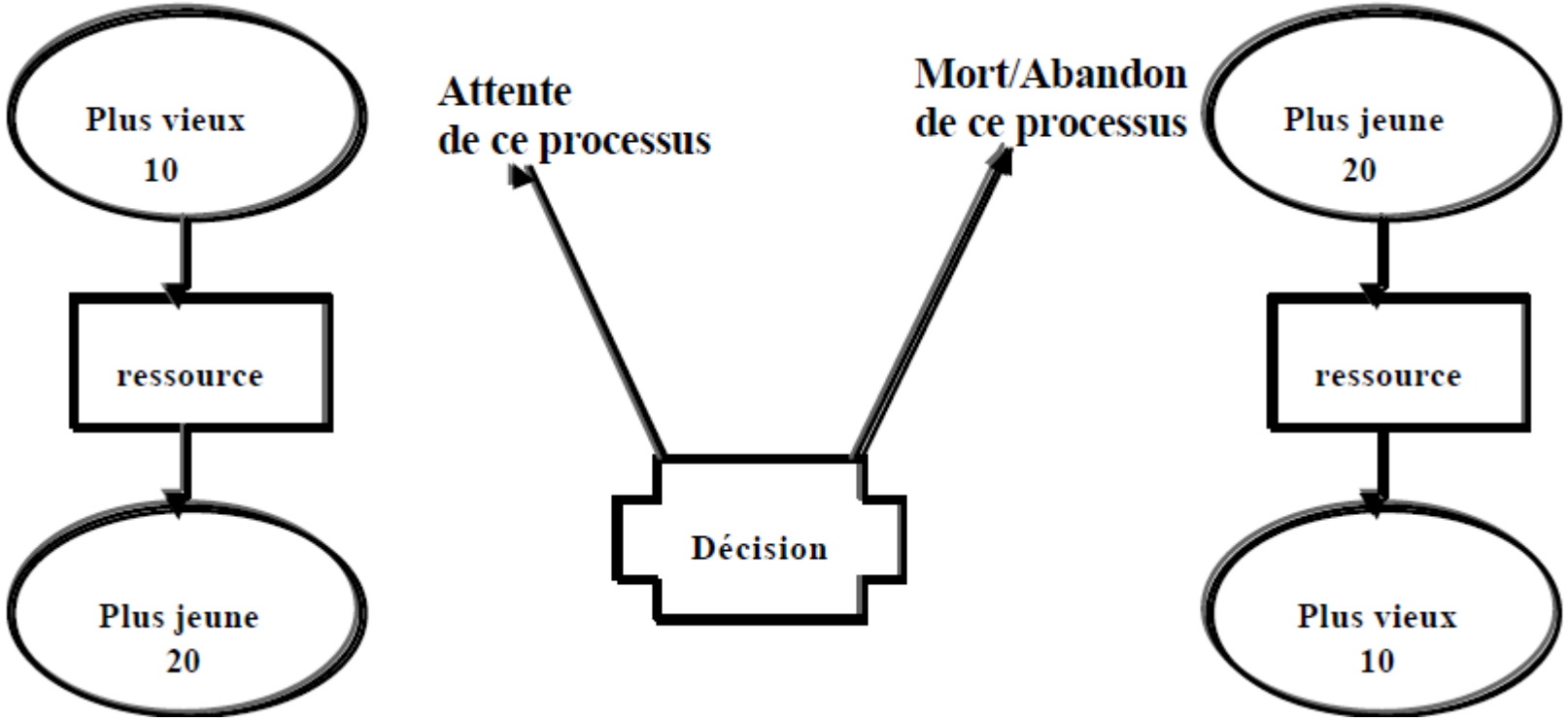
- On considère deux transactions t1 et t2 et une ressource r utilisée par t1 et demandée par t2.
- A la création de chaque transaction une estampille lui est associée; soient e(t1) et e(t2) les estampilles des deux transactions
- L'algorithme auquel obéit l'allocateur situé sur le site de la ressource reste le suivant :

si e(t2) < e(t1) alors bloquer t2 /* wait */
sinon annuler t2 /* die */
fsi

  - si la transaction t2 est la plus ancienne elle reste bloquée jusqu'à ce que t1 libère la ressource.
  - sinon t2 est annulée et sera redémarrée ultérieurement avec le même numéro d'estampille
  - cet algorithme évite la famine des transactions annulées

Algorithmes de prévention

Technique sans réquisition : « Wait-Die System»



- Plus vieux est un processus, plus il risque d'attendre; Un processus tué peut mourir plusieurs fois si la situation n 'a pas changé. Garder même estampille pour éviter famine. Le graphe des processus en attente est sans circuit (ordre des estampilles croissant).

Algorithmes de prévention

Technique sans réquisition : «Wound-Wait System»

- Une solution consiste à satisfaire systématiquement les demandes de transactions plus anciennes et donc d'annuler les transactions plus jeunes qui utilisent les ressources : il y a donc réquisition.
- L'algorithme de l'allocateur situé sur le site de la ressource r est le suivant (r est utilisée par t1 et demandée par t2) :

  si e(t2) < e(t1) alors annuler t1 /* Wound */
  sinon bloquer t2 /* Wait */
  fsi
- Une transaction n'attend jamais une ressource utilisée par une transaction plus jeune.
- Conclusion :
  - Ces algorithmes ont l'avantage d'être simples
  - Peuvent entraîner l'annulation de transactions qui n'auraient pas conduit à un interblocage
  - Intéressant si peu de conflits sur une même ressource