Cours 5

Chapitre 3
Algorithmes Distribués

1. Détection de Terminaison
2. L'Interblocage distribué
3. … etc.

- A fundamental problem in distributed systems is to determine if a distributed computation has terminated.
- A distributed computation is considered to be globally terminated if every process is locally terminated and there is no message in transit between any processes

1. Détection de Terminaison

- Hypothèses :
  - Communications asynchrones.
  - Processus sont ordonnés.

Termination Detection : token ring algorithm

- This algorithm is based on a token going around the ring.
- The token collects the information from all processes and determines whether the computation has terminated.

Termination Detection : token ring algorithm

- **Each process maintain the following variables :**
  1. State: The state of a process is either *active or passive (*Each process in the system is either *passive or active.* ***It is*** **assumed that a passive process can become active *only* on receiving a message** (an active process can become passive at any time).

Termination Detection : token ring algorithm

2. Color : The color of a process is either ***white*** *or* ***black****. If the process is* ***white****,* then it has not received any message since the last visit of the token. This variable is initialized to ***white****.*
3. C: This is an integer variable maintained by each process. It records the value of the number of messages sent by the process *minus* the number of messages received by that process. This variable is initialized to 0

Termination Detection : token ring algorithm

- Process *Po begins the detection probe by sending token to the next process when* it is passive.
- The token consists of two fields: color and count. The color simply records if the token has seen any *black process.*
- *The count records sum of all* ***c*** variables seen in this round.
- When a process receives the token, it keeps the token until it becomes **passive**.

Termination Detection : token ring algorithm

- It then forwards the token to the next process.
  - Thus, if a black process forwards the token, the token turns black.
- The count variable in the token is increased by c
- The process resets its own color to *white* after sending the token.

Termination Detection : token ring algorithm

Termination Detection : token ring algorithm

- Process *P0 is responsible for detecting termination. On receiving the token, Po* detects termination, if :
  1. its own color is *white*,
  2. the token is *white and* the sum of token *count and c of P0 is 0*.

- If termination is not detected, then *Po can start* a new round of token passing

- Chaque algorithme :
  1. Hypothèses
  2. Description
- Algorithmes:
  - Dijkstra, Feijen et Van Gasteren.
  - …

D’autres Algorithmes de déctection de terminaision ‘Homework -2-’.

2. Distributed deadlock

Distributed deadlock

- Les interblocage dans les SD sont comparable à ceux des SE.
- Difficile à éviter, à prévenir, ou même à détecter et y remédier.
- Information nécessaire est disséminée dans plusieurs machines

1. La politique de l’Autriche.
2. La détection.
3. La prévention
4. L’évitement

Gestion de l’interblocage

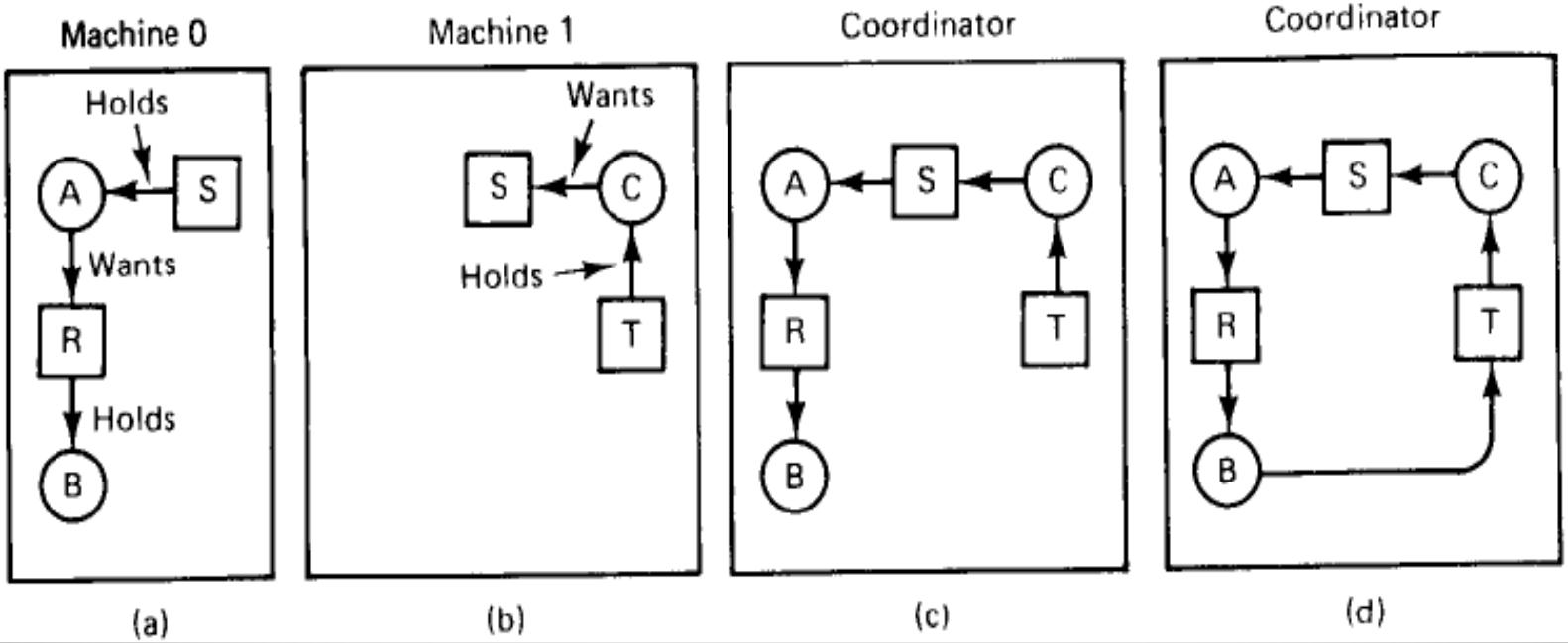
Gestion de l’interblocage

1. Ignorer le problème
2. Permettre aux inter blocage de se produire, les décréter, puis tenter de les éliminer.
3. Rendre de façon statique les inerblocages structurellement impossible.
4. Éviter les interblocage.

1 Détection

- Solution centralisée
  - Graphe des ressources et des processus
  - Coordinateur gère le graphe de tous le système
  - Détection cycle ➔ kill one process
  - **Collection d'information,** plusieurs politiques :
    - Ajout ou suppression d'une arc
    - Liste des arcs ajoutés ou supprimés périodiquement lors de la dernière mise à jour.
    - Coordinateur demande les informations dont il a besoin.

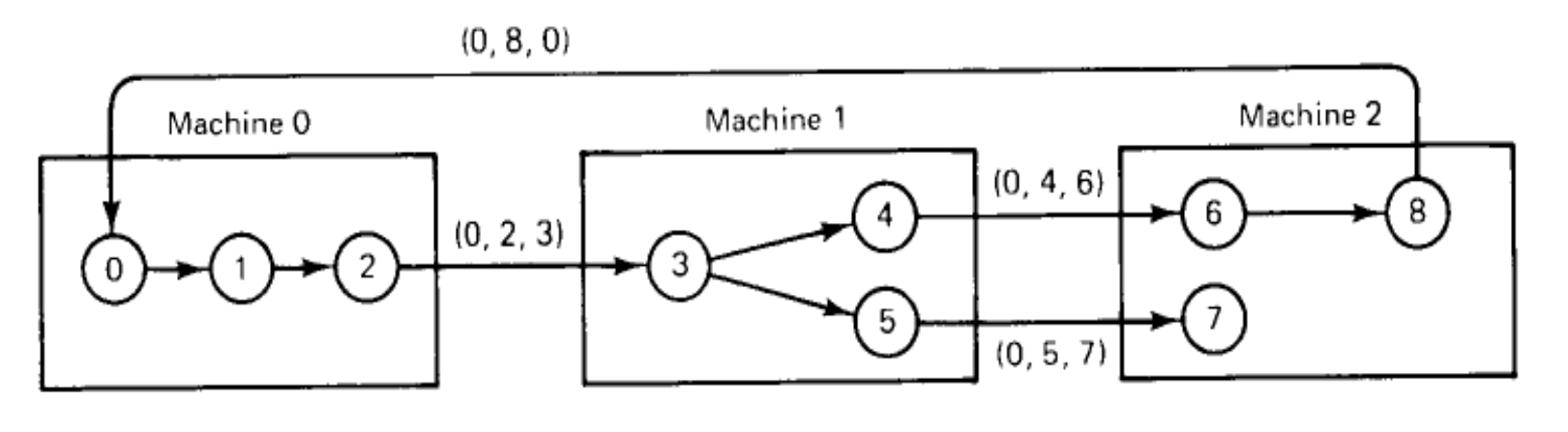
- Exemple : problème et solution ?



After a while, *B* releases *R* and asks for *T*, a perfectly legal and safe swap. Machine 0 sends a message to the coordinator announcing the release of *R*, and machine 1 sends a message to the coordinator announcing the fact that *B* is now waiting for its resource, *T*. Unfortunately, the message from machine 1 arrives first, leading the coordinator to construct the graph of Fig. 3-23(d). The coordinator incorrectly concludes that a deadlock exists and kills some process. Such a situation is called a **false deadlock**. Many deadlock algorithms in distributed systems produce false deadlocks like this due to incomplete or delayed information.

Détection

- Solution Distribuée
- Algorithme de Chandy-Misra-Hass 1983.



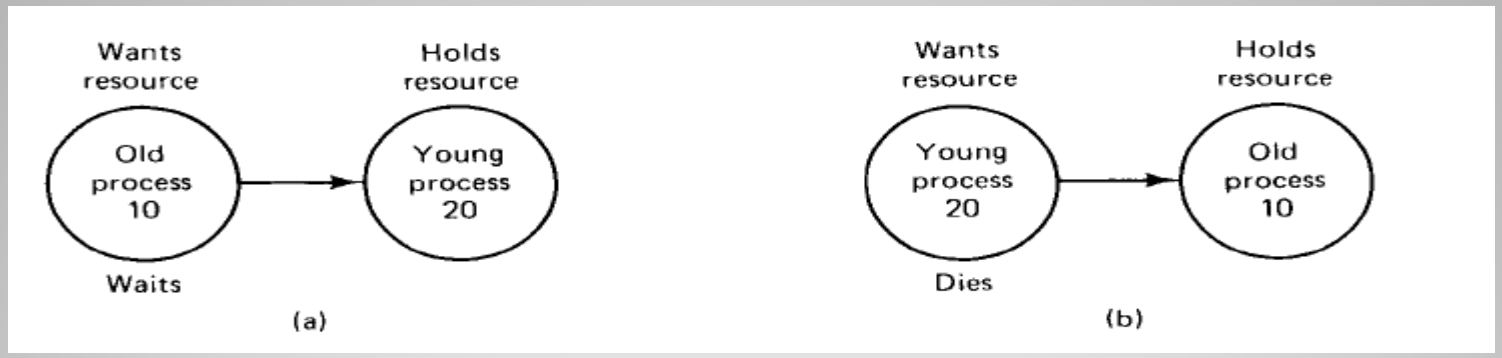
- Lorsque 0 se bloque sur 1 (il demande une ressource détenu par le processus 1) : (0,0,1)
- Envoi ce message à 1, 1:(0,1,2)→2,(0,2,3)→3,(0,3,4)3 ,(0,3,5)5,(0,5,7)→7. 4,→(0,4,6)6,→(0,6,8)8,→(0,8,0)0

- Rendre de façon statique les interblocages structurellement impossible ?

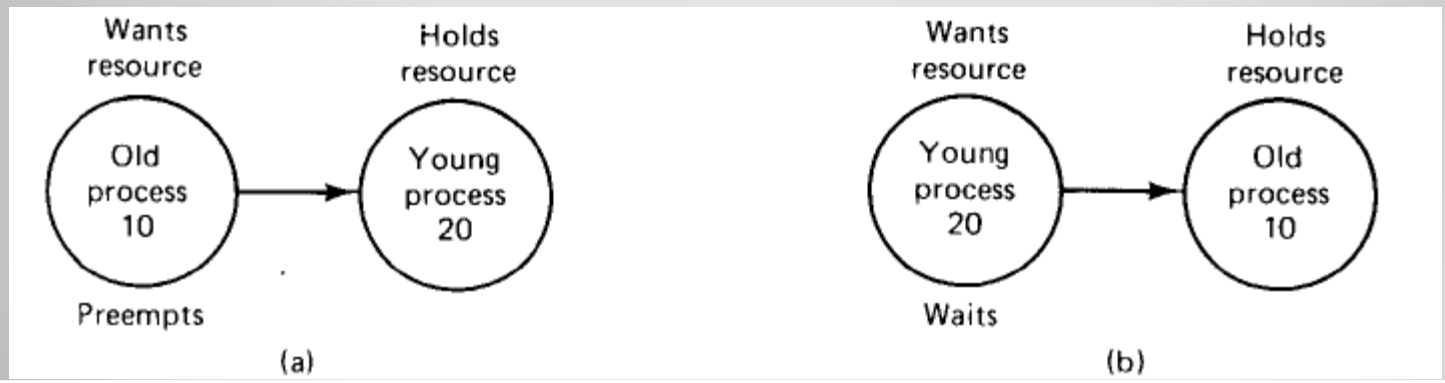
1. Permettre aux processus de détenir qu'une seul ressource à la fois.
2. Exiger aux processus de spécifier au départ toutes les ressources dont ils besoins.
3. Obliger les processus de relâcher toutes celles dont ils disposent lorsqu'ils désirent une nouvelle.
4. Ordonner toutes les ressources et de demander aux processus de les acquérir dans un ordre strictement croissant. Ceci signifier qu'un processus ne peut pas demander une ressource de numéro faible s'il détient une de numéro élevé, ce qui rend tout cycle impossible.
5. Vos propositions : ?

2. Prévention

- Utiliser l'algorithme de Lamport pour l'estampillage → unicité de l'estampille.
- Pas grave de tuer une transaction; par définition elle peut être redémarrer plus tard
- **Algorithme de type attente-mort (*wait-die*)**



- **Algorithme de type bluseur-attente (*wound-wait*)**



Deux algorithmes à comparer ?

Homework -3-

- Pour la prévention de l'interblocage, nous considérions le scénario suivant : un processus (transaction) récent veut une ressource que détient un vieux processus. Quel algorithme mieux à utiliser attente-mort ou blessure-attente. Justifier votre réponse.
- Utiliser l'algorithme de calcule de l'état global pour détecter la terminaison d'un algorithme distribué.