CHAPITRE II : SYNCHRONISATION DES PROCESSUS

INTRODUCTION

- Les processus dans un système d'exploitation, n'évoluent pas toujours de manière indépendante.
- Les relations entre processus peuvent prendre deux formes :
  - Compétition pour l'usage d'une ressource partagée,
    - Exemple: Allocation du PC, Accès à une variable partagée …
  - Coopération pour l'exécution coordonnée d'une tâche commune.
    - Exemple: Communication par transmission de messages entre deux processus à travers une zone de mémoire commune …
- La mise en œuvre de ces relations nécessite des mécanismes dits de synchronisation qui permettent d'ordonner l'exécution des processus en vue de respecter les contraintes de compétition ou de coopération.

PARTAGE DE RESSOURCES ET DE DONNÉES

- L'exécution d'un processus nécessite un certain nombre de ressources.
- Deux types de ressources :
  - Physiques telles que : la mémoire, le processeur, les périphériques …
  - Logiques telles que une variable, un fichier, un code …
- Ces ressources peuvent être :
  - Partageables ou à accès multiple : utilisées en même temps (ou partagées) par plusieurs processus ⟹ Pas de contrôle d'accès.
  - Non partageables ou à un seul point d'accès ou à accès exclusif : un seul processus à la fois ⟹ nécessité d'ordonner l'accès à ce type de ressources pour éviter des situations incohérentes.

EXEMPLES DE SITUATIONS INCOHÉRENTES

- Problème de ressource matérielle partagée

Exemple:

Deux processus P1 et P2 désirent imprimer sur la même imprimante deux fichiers texte F1 et F2.

- En l'absence de synchronisation explicite, on peut voir imprimer n'importe quelle séquence de caractères :
  - Quelques mots de F1 et d'autres de F2;
  - Quelques lignes de F2 et d'autres de F1;
  - Quelques pages de F1 et d'autres de F2;….
- Mais, on devrait voir imprimée le contenu de F1 ensuite le contenu de F2 ou l'inverse.

EXEMPLES DE SITUATIONS INCOHÉRENTES

- Problème de variables partagées

Exemple 1:

Incrémentation et décrémentation d'un compteur par deux processus P1 et P2.

```
P1 :

(T1) Mov Ax, Cpt
(T2) Inc Ax
(T3) Mov Cpt, Ax
```

```
P2 :

(T4) Mov Ax, Cpt
(T5) Dec Ax
(T6) Mov Cpt, Ax
```

- Si Cpt = 1, l'entrelacement de l'exécution de P1 et P2 donne en résultat une valeur de Cpt égale à 0, 1 ou 2.

EXEMPLES DE SITUATIONS INCOHÉRENTES

- Problème de variables partagées

Exemple 2:

- Soit le programme de réservation de places (avion, théâtre, etc.) suivant :

```
Programme_reservation
   Si PLACE_DISPO > 0
   |     Alors    PLACE_DISPO ← PLACE_DISPO - 1 ;
   |              Répondre ('Place réservée') ;
   |     Sinon    Répondre ('Plus de places') ;
   Fsi;
```

EXEMPLES DE SITUATIONS INCOHÉRENTES

- Problème de variables partagées

Exemple 2: (suite)

```
(B1)    Mov Ax, PLACE_DISPO
(B2)    JZ  Etiq  // Saut à Etiq si Ax = 0
(B3)    Dec Ax
(B4)    Mov PLACE_DISPO, Ax
(B5)    Repondre    ('Place réservée')
(B6)    JMP         Fin
(B7)    Etiq:   Repondre    ('Plus de places')
(B8)    Fin:    Stop
```

- Plusieurs demandes de réservation simultanées émanant d'agences de réservation différentes engendreront l'exécution de plusieurs processus composés de cette même séquence d'instructions avec PLACE_DISPO comme variable commune.

EXEMPLES DE SITUATIONS INCOHÉRENTES

- Problème de variables partagées

Exemple 2: (suite)

- Un scénario possible de l'exécution de deux processus P1 et P2 est le suivant :
  - Supposons que PLACE_DISPO := 1 ;
  - P1 s'exécute jusqu'à B3, les valeurs correspondantes de Ax seront : 1 ensuite 0.
  - Puis, P2 s'exécute jusqu'à la fin (PLACE_DISPO est toujours égale à 1), il réserve une place et se termine.
  - Ensuite, P1 continue son exécution à partir de B4, il réserve une place et se termine.
- Problème d'incohérence car une même place a été réservée deux fois.

EXCLUSION MUTUELLE

- Le partage d'une ressource à un seul point d'accès par plusieurs processus nécessite une utilisation séquentielle selon un ordre défini par le mécanisme de synchronisation utilisé.
- La portion de code utilisant la ressource est appelée section critique.
- La ressource partagée est appelée ressource critique.
- Le mécanisme utilisé pour la synchronisation est appelé mécanisme d'exclusion mutuelle.

PROBLÈME DE LA SECTION CRITIQUE

- Considérons un système comprenant *n* processus $\{P_0, P_1, \ldots, P_{n-1}\}$

  P_0 P_1 … P_{n-1}

  … … …

  $\langle SC_0 \rangle$ $\langle SC_1 \rangle$ $\langle SC_{n-1} \rangle$ ⟶ un seul processus qui exécute $\langle SC_i \rangle$

  … … …

- Quand un processus est en exécution dans sa section critique, aucun autre n'est autorisé à exécuter sa section critique.

- Nécessité de concevoir un protocole permettant aux processus de s'exclure mutuellement quant à l'utilisation de la ressource critique.

PROBLÈME DE LA SECTION CRITIQUE

- La forme générale d'un processus devient:

Pi

…

<Protocole d'entrée>

<Section critique>

<Protocole de sortie>

…

- Synchronisation
  - Matériel
  - Logiciel
    - SE
    - Pgmes d'application

CONDITIONS DE LA SECTION CRITIQUE

- Une solution adaptée au problème de la section critique doit satisfaire les conditions suivantes :
  - Exclusion mutuelle : Si un processus Pi est dans sa section critique, aucun autre ne doit être dans sa section critique (au plus un processus doit être autorisé à y accéder : 0 ou 1).
  - Absence de blocage mutuel : Les processus ne doivent pas s'empêcher mutuellement d'entrer en sc. Si deux ou plusieurs processus essayent d'entrer dans leurs sections critiques, au moins un réussira.
  - Progression : Si un processus opère en dehors de sa section critique, il ne doit pas empêcher un autre d'entrer dans sa section critique, s'il le désire.
  - Attente bornée : un processus ayant demandé d'entrer en section critique doit y accéder au bout d'un temps fini (Eviter le problème de famine).

MÉCANISMES D'EXCLUSION MUTUELLE

- Différents mécanismes de synchronisation existent:
  - Solutions sans support matériel (les variables d'état).
  - Solutions matérielles.
  - Les sémaphores.
  - Les moniteurs.

MÉCANISMES D'EXCLUSION MUTUELLE

Hypothèses

- Le système est composé de *n* processus P_0, P_1, …, P_n qui s'exécutent de manière parallèle.
- Les vitesses d'exécution des processus sont quelconques et inconnues.
- Les instructions de base du langage machine sont exécutées de manière atomique.
- Tout processus qui entre dans sa section critique doit forcement sortir au bout d'un temps fini.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

- Ces solutions consistent à utiliser des variables communes partagées entre les processus.
- Pour simplifier la présentation de ces solutions, on considérera uniquement deux processus Pi et Pj.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 1

```
Tour := i ou j ;
Pi :
Répéter
    Tant que (Tour ≠ i)
     faire
       <rien>
     fait ;
    <SCi>
    Tour := j;
Jusqu'à faux
```

Analyse

1) Condition d'entrée en SC

Pi : Tour = i / Pj : Tour = j

2) Invariant pendant la SC

Pi : Tour = i / Pj : Tour = j

3) Vérification des 4 conditions

EM assurée ?

BM évité ?

Progression assurée ?

AB assurée ?

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 1

Exclusion mutuelle

- Deux scenarios à considérer :
  - On suppose que Pj est en SC et Pi désire entrer. Risque t-il de le joindre?
    - Pj en SC ⇒ Tour = j ≠ i ⇒ Pi ne peut pas entrer en SC.
  - SC est libre, Pi et Pj désirent enter. Risque t-ils d'enter en même temps?
    - 1er cas : Tour = j ⇒ Pj rentre en SC et Pi boucle sur la condition (Tour ≠ i).
    - 2e cas : Tour = i ⇒ Pi rentre en SC et Pj boucle sur la condition (Tour ≠ j).
    - D'après les deux cas, un seul processus en SC.
- Conclusion : l'EM est assurée.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 1

Blocage mutuel

- SC est libre, Pi et Pj désirent enter. Risque t-ils de s'empêcher mutuellement?
  - Un seul processus entre en SC selon la valeur de Tour.
- Conclusion : le BM est évité.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 1

Progression

- Pj loin de la SC (Pas venu/ Sorti), et Pi désire enter. Risque t-il d'attendre indéfiniment?
  - 1er cas : Pj quitte la SC ⇒ Tour = i ⇒ Pi rentre en SC.
  - 2e cas : Supposons qu'initialement Tour = j et Pj ne désire pas accéder à la SC (pas venu) ⇒ Pi boucle indéfiniment sur la condition (Tour ≠ i).

- Conclusion : la progression n'est pas assurée.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 1

Attente Bornée

- Pj est en SC et Pi en attente. Risque t-il d'attendre indéfiniment si Pj sort de la SC et la redemande une autre fois ?
  - Pj quitte la SC $\Rightarrow$ Tour = i $\Rightarrow$ Pi rentre en SC. Quelque soit la vitesse de Pj, il ne pourra jamais accéder une autre fois à la SC jusqu'à ce que Pi accédera.

- Conclusion : l'Attente Bornée est assurée.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 1

Discussion

- La solution engendre l'alternance quant à l'accès à la section critique.
- Elle indique seulement lequel des processus est en section critique.
- Elle ne garde pas des informations sur l'état d'un processus par rapport à la section critique.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 2

```
drapeau : Tableau[0,1] de
booléen := faux ;
Pi :
Répéter
    drapeau[i] := vrai ;
    Tant que (drapeau[j])
     faire
       <rien>
     fait ;
    <SCi>
    drapeau[i] := faux ;
Jusqu'à faux
```

Analyse

1)Condition d'entrée en SC

Pi : drapeau[j] = faux

Pj : drapeau[i] = faux

2) Invariant pendant la SC

Pi : drapeau[i] = vrai

Pj : drapeau[j] = vrai

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 2 : 3) Vérification des 4 conditions

Exclusion mutuelle

- Deux scenarios à considérer :
  - On suppose que Pj est en SC et Pi désire entrer. Risque t-il de le joindre?
    - Pj en SC ⇒ drapeau[j] = vrai ⇒ Pi met son drapeau à vrai et reste en attente tant que drapeau[j] = vrai.
  - SC est libre, Pi et Pj désirent enter. Risque t-ils d'enter en même temps?
    - 1er cas : Pi et Pj n'ont pas la même vitesse d'exécution ⇒ le premier qui franchit la boucle rentre en SC et l'autre reste en attente ⇒ un seul processus en SC.
    - 2e cas : Pi et Pj ont la même vitesse d'exécution ⇒ Pi et Pj mettent leurs drapeaux à vrai et franchisent la boucle en même temps ⇒ aucun processus ne rentre en SC.
- Conclusion : l'EM est assurée.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 2

Blocage mutuel

- SC est libre, Pi et Pj désirent enter. Risque t-ils de s'empêcher mutuellement?
  - Si Pi et Pj ont la même vitesse d'exécution et arrivent en même temps ⇒ aucun processus ne rentre en SC.

- Conclusion : le BM n'est pas évité.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 2

Progression

- Pj loin de la SC (Pas venu/ Sorti), et Pi désire enter. Risque t-il d'attendre indéfiniment?
  - 1er cas : Pj quitte la SC ⇒ drapeau[j] = faux ⇒ Pi rentre en SC.
  - 2e cas : Pj ne désire pas accéder à la SC (pas venu) ⇒ drapeau[j] = faux ⇒ Pi rentre en SC.

- Conclusion : la progression est assurée.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 2

Attente Bornée

- Pj est en SC et Pi en attente. Risque t-il d'attendre indéfiniment si Pj sort de la SC et la redemande une autre fois ?
  - Pj quitte la SC ⇒ drapeau [j] = faux et désire entrer une autre fois :
    - 1er cas : Pi et Pj ont la même vitesse d'exécution ⇒ Pi quitte la boucle et rentre en SC. Par contre Pj met son drapeau à vrai et reste en attente car drapeau [i] = vrai.
    - 2e cas : Pj est plus rapide que Pi ⇒ Pj remet son drapeau à vrai avant que Pi ne constate que drapeau[j] est passé à faux ⇒ Pi et Pj se bloque sur la condition drapeau[i ou j] = vrai ⇒ aucun processus ne rentre en SC.

- Conclusion : l'Attente Bornée n'est pas assurée.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 3 (Algorithme de Peterson, 1981)

- Cette solution combine les deux solutions précédentes.

```
drapeau : Tableau[0,1] de booléen :=
faux ;
tour := i ou j ;
Pi :
Répéter
    drapeau[i] := vrai ; tour := j;
    Tant que (drapeau[j] et tour = j )
     faire
       <rien>
     fait ;
   <SCi>
    drapeau[i] := faux ;
Jusqu'à faux
```

```
drapeau : Tableau[0,1] de booléen :=
faux ;
tour := i ou j ;
Pj :
Répéter
    drapeau[j] := vrai ; tour := i;
    Tant que (drapeau[i] et tour = i )
     faire
       <rien>
     fait ;
    <SCi>
    drapeau[j] := faux ;
Jusqu'à faux
```

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 3 (Algorithme de Peterson, 1981)

Analyse

1)Condition d'entrée en SC

Pi : drapeau[j] = faux ou tour = i

Pj : drapeau[i] = faux ou tour = j

2) Invariant pendant la SC

Pi : drapeau[i] = vrai

Pj : drapeau[j] = vrai

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 3 : 3) Vérification des 4 conditions

Exclusion mutuelle

- Deux scenarios à considérer :
  - Pj est en SC et Pi désire entrer. Risque t-il de le joindre?
    - Pj en SC ⇒ drapeau[j] = vrai ⇒ Pi met son drapeau à vrai et tour à j ⇒ Pi reste en attente car drapeau[j] = vrai et tour = j.
  - SC est libre, Pi et Pj désirent enter. Risque t-ils d'enter en même temps?
    - Pi et pj mettent leurs drapeaux à vrai:
    - 1er cas : Tour = j ⇒ Pj rentre en SC et Pi reste en attente.
    - 2e cas : Tour = i ⇒ Pi rentre en SC et Pj reste en attente.
    - D'après les deux cas, un seul processus en SC.
- Conclusion : l'EM est assurée.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 3

Blocage mutuel

- SC est libre, Pi et Pj désirent enter. Risque t-ils de s'empêcher mutuellement?
  - Un seul processus entre en SC selon la valeur de Tour.

- Conclusion : le BM est évité.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 3

Progression

- Pj loin de la SC (Pas venu/ Sorti), et Pi désire enter. Risque t-il d'attendre indéfiniment?
  - 1er cas : Pj quitte la SC ⇒ drapeau[j] = faux ⇒ Pi rentre en SC.
  - 2e cas : Pj ne désire pas accéder à la SC (pas venu) ⇒ drapeau[j] = faux ⇒ Pi rentre en SC.

- Conclusion : la progression est assurée.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Solution 3

Attente Bornée

- Pj est en SC et Pi en attente. Risque t-il d'attendre indéfiniment si Pj sort de la SC et la redemande une autre fois ?
  - Pj quitte la SC ⇒ drapeau [j] = faux et désire entrer une autre fois :
    - 1er cas : Pi et Pj ont la même vitesse d'exécution ⇒ Pi quitte la boucle et rentre en SC. Par contre Pj met son drapeau à vrai et tour à i et reste en attente car drapeau [i] = vrai et tour = i.
    - 2e cas : Pj est plus rapide que Pi ⇒ Pj remet son drapeau à vrai avant que Pi ne constate que drapeau[j] est passé à faux ⇒ Pi reste en attente car drapeau [j] = vrai et tour = j mais, Pj remet la valeur de tour à i, ce qui permet de débloquer Pi. ⇒ Pi rentre en SC.
- Conclusion : l'Attente Bornée est assurée.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS SANS SUPPORT MATÉRIEL (VARIABLES D'ÉTAT)

Discussion

- La généralisation à N processus est soit complexe ou impossible.
- La généralisation à k-Exclusion mutuelle (K ressources) n'est pas réalisée.
- Les solutions présentées sont valables seulement pour le problème de compétition.
- Ces solutions souffrent du problème d'attente active.
- L'attente bornée n'est pas toujours assurée ce qui conduit au problème de famine.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS MATÉRIELLES

- Masquage d'interruptions
  - Le processus masque les interruptions avant d'entrer en SC et les restaure à la sortie.
  - Solutions dangereuses : certaines tâches du système fonctionnant à l'aide des interruptions seront donc inhibées (Ex. horloge).
  - Elles n'assurent pas l'exclusion mutuelle en multiprocesseur : le masquage des interruptions concerne uniquement le processeur qui a demandé l'interdiction. Les autres processus exécutés par un autre processeur pourront donc accéder aux objets partagés.

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS MATÉRIELLES

- Instructions spéciales
  - Instructions élémentaires exécutées par le matériel, qui permettent de lire et d'écrire le contenu d'un mot mémoire de manière indivisible.
  - Lorsqu'un processeur exécute ces instructions, il verrouille le bus de données pour empêcher les autres processeurs d'accéder à la mémoire pendant la durée de l'exécution.
  - Instruction Test-and-Set (TAS ou TS)
  - Instruction SWAP
  - Instruction Compare-and-Swap (CAS)

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS MATÉRIELLES

Test-and-Set (TAS)

```
Fonction Test-and-Set(x : booleen) : booleen ;
Début
   Test-and-Set := x ;
   x := true ;
Fin
```

Programmation de la SC à l'aide de TAS

```
Var Lock : booleen := false ;
Pi
Répéter
   Tant que Test-and-Set(Lock) faire <rien> fait ;
   <SCi> ;
   Lock := false ;
   <Reste du processus>
Jusqu'à faux
```

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS MATÉRIELLES

SWAP

```
Procédure Swap (var X, Y : Booleen)
Var  Temp : Booleen ;
Début
   Temp := X ;
   X := Y ;
   Y := Temp ;
Fin.
```

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS MATÉRIELLES

Compare-and-Swap (CAS)

```
Fonction Compare -and-Swap(V : entier, oldval :
entier, newval : entier) : booleen ;
Début
   Si (V = oldval) Alors  V := newval;
                          Compare-and-Swap := true ;
   Fsi;
 Compare-and-Swap := false ;
Fin
```

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS MATÉRIELLES

Exemple d'utilisation de l'instruction CAS : Incrémentation de la même variable val par plusieurs processus

```
Var val : entier;
Pi
   ...
   val := val + 1;
   ...
```

```
Var val : entier;
Pi
  ...

  oldval := val;
  Tant que (Compare-and-Swap(val, oldval, oldval + 1)) = false
      faire
         oldval := val;
      fait ;

  ...
```

MÉCANISMES D'EXCLUSION MUTUELLE
SOLUTIONS MATÉRIELLES

Discussion

- Solutions fonctionnent pour un nombre indéterminé de processus.
- Ces solutions souffrent du problème d'attente active.
- L'attente bornée n'est pas toujours assurée ce qui conduit au problème de famine.