UNIVERSITE DE ZIGUINCHOR···

Informatique Appliquée

Licence 2

♦◊♦♦◊♦♦◊♦◊♦

UFR Sciences et Techniques

♦◊♦♦◊♦

Département d’informatique

Examen Session de Rattrapage Semestre1 : Système d’Exploitation

Documents non autorisés Durée 1h 30mn

**Question de cours**

Question 1

Quels sont les différents services offerts par un système d’exploitation et illustrez chaque fonction à l’aide d’un exemple ? (citez de manière non exhaustive)

Question 2

Décrire en quelques lignes (et éventuellement des schémas) 4 techniques

d’ordonnancement sur une machine mono-processeur. Donner les inconvénients et les avantages de chacune.

Exercice 1

On considère 4 processus, A, B, C, D. On suppose que l’exécution des processus nécessite :

Pour A : 7 unités de temps CPU, 3 unités de temps d'E/S et 5 unités de temps CPU.

Pour B : 6 unités de temps CPU, 4 unités de temps d’E/S, 4 unités de temps CPU.

Pour C : 5 unités de temps CPU.

Pour D : 1 unité de temps CPU, 4 unités de temps d’E/S et 2 unités de temps CPU.

On suppose que

A se présente en premier, à l’instant 0,

B se présente à l’instant 1,

C se présente à l’instant 9,

D se présente à l’instant 12.

Montrez comment les 4 processus vont utiliser le processeur dans chacun des cas suivants :

1) Chaque processus a son propre périphérique d’E/S et l’ordonnanceur fonctionne selon Shortest Job First.

2) Les trois processus utilisent le même périphérique d'E/S dont la file d'attente est gérée premier arrivée premier servi. L’ordonnanceur du processeur utilise l'algorithme du tourniquet, avec un quantum de 2. Le temps de commutation est supposé égal à 0.

Exercice 2 :

1) Soient trois processus concurrents P1, P2 et P3 qui partagent les variables n et out. Pour contrôler les accès aux variables partagées, un programmeur propose les codes suivants :

Semaphore mutex1 = 1 ;

Semaphore mutex2 = 1 ;

Code du processus p1 :

P(mutex1) ;

P(mutex2) ;

out=out+1 ;

n=n-1 ;

V(mutex2) ;

V(mutex1) ;

Code du processus p2 :

P(mutex2) ;

out=out-1 ;

V(mutex2) ;

Code du processus p3 :

P(mutex1) ;

n=n+1 ;

V(mutex1) ;

Cette proposition est-elle correcte ? Sinon, dites pourquoi et proposer une solution correcte ?

Bonne Chance !!!!!!!

Réponses aux questions

**Question de cours**

*Question 1*

Services: Gestion de périphérique : périphériques exemple l’Imprimante

Gestion des processus exemple la multiprogrammation

Gestion de la mémoire exemple la RAM

Gestion des fichiers exemple opération lecture, écriture

*Question 2*

Quatre Techniques d’ordonnancement:

FIFO (first In First Out) : Le traitement du processus est séquentiel. Premier arrivé en file d'attente, premier traité

Avantageux pour les processus longs.

Inconvénient : processus court sont pénalisés.

SJF (Shortest JOB First) : La file d'attente est ordonnée non plus de façon chronologique mais en fonction du temps d'exécution nécessaire (on fait passer en tête les travaux courts).

Avantage  pour les travaux courts.

Inconvénient : Les travaux longs sont pénalisés au pire ils risquent de ne jamais être exécutés si beaucoup de petits travaux sont soumis au système.

SRTF (shortest remaining-time first) même définition que SJF mais avec requisition, si un processus qui dure moins que le *restant* du processus courant se présente plus tard, l’UCT est enlevée au processus courant et donnée à ce nouveau processus

Avantageux pour les plus petits travaux.

Inconvénient : travaux longs pénalisé

Le tourniquet : consiste à vider les processus qui s'attardent trop dans le processeur. Le système passe d'un processus à un autre. Un processus est remis en file d'attente dès que sa durée d'occupation du processeur dépasse une durée prédéfinie : le quantum de temps.

Avantage : Les travaux assez courts sont vite servis. Le tourniquet garanti que les travaux longs sortiront du système au bout d'un temps fini. Cependant l'efficacité du système dépend de la valeur du quantum

Inconvénient : Trop petit quantum, la machine perd du temps à changer de contexte (swapping)

ce qui la rend lente. Trop long quantum : les petits travaux ne sont pas exécutés

Exercice 1

) Chaque processus a son propre périphérique d’E/S et l’ordonnanceur fonctionne selon Shortest Job First.

Le schéma d’occupation du CPU en fonction du temps sera présenté sous forme de tableau :

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 |  | 6 |  | 1 |  | 5 |  | 2 |  | 4 |  | 5  -->unité temps/passage CPU |
| A |  | B |  | D |  | C |  | D |  | B |  | A  -->Occupation CPU |

0 7 13 14 19 21 25 30

Temps

2) Les trois processus utilisent le même périphérique d'E/S dont la file d'attente est gérée premier arrivée premier servi. L’ordonnanceur du processeur utilise l'algorithme du tourniquet, avec un quantum de 2. Le temps de commutation est supposé égal à 0.

NB : Ici deux schémas sont possibles car on a pas précisé si un processus quitte la file d’attente des E/S et au même moment un autre quitte le processeur qui sera en tête de file si on applique FIFO. A priori c’est le processus qui vient du processeur qui sera en tête de file car il n’a pas été bloqué alors que l’autre était à l’état bloqué, raison pour la quelle les deux schéma seront acceptés. L’étudiant doit au moins présenter l’un des schémas suivant

2 2 2 2 2 2 2 1 1 2 2 2 1 2 2 3

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | A | B | A | B | C | A | D | C | B | A | C | D | B | A |

0 2 4 6 8 10 12 14 15 16 18 20 22 23 25 27 30

temps

ou

2 2 2 2 2 2 2 1 1 2 2 1 2 2 2 3

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | A | B | A | B | C | A | D | C | B | C | A | D | B | A |

0 2 4 6 8 10 12 14 15 16 18 20 21 23 25 27 30

temps

Exercice 2

Non, car si P2 est en section critique et P1 a exécuté P(mutex1) alors P1 est bloqué et empêche P3 d’entrer en section critique. Conditions non vérifiées : Un processus en dehors de sa section critique bloque un autre processus.

Proposition :

*Semaphore mutex1 = 1 ;*

*Semaphore mutex2 = 1 ;*

*Processus P1*

*P(mutex1) ;*

*n=n-1 ;*

*V(mutex1) ;*

*P(mutex2) ;*

*Out = out +1 ;*

*V(mutex2) ;*

*Code du processus p2 :*

*P(mutex2) ;*

*out=out-1 ;*

*V(mutex2) ;*

*Code du processus p3 :*

*P(mutex1) ;*

*n=n+1 ;*

*V(mutex1) ;*

