

Examen de Remplacement de Systèmes d'Exploitation 1

Date : 01/06/2023

Durée: 1h30 - Documentation non autorisée

Les exercices (2) et (5) sont à comptabiliser pour l'interrogation pour les étudiants ayant ratés l'interrogation.

Exercice 1 : (Questions de Compréhension : 3 pts) (10 minutes)

Q1) Décrire (sur 2 à 3 phrases) la différence entre les algorithmes d'ordonnancement de programmes préemptifs et non préemptifs. Lequel qui convient mieux pour un système à temps partagé ? (1. pt)

- ✓ L'ordonnancement de programme préemptif permet à un processus d'être interrompu au milieu de son exécution, en lui retirant la cpu et l'allouant à un autre processus.
- ✓ L'ordonnancement non préemptif s'assure qu'un processus lâche l'unité centrale de traitement seulement quand il termine avec son exécution en cours.

C'est les algorithmes préemptifs (ou bien round robin) qui conviennent au système à temps partagé.

Q2) Qu'est-ce qu'une interruption? Quand cela se produit-il? (1. pt)

Une interruption est un évènement généralement imprévu qui interrompt la séquence normale d'exécution des instructions par le microprocesseur. Les interruptions se produisent quand un périphérique signale un évènement, quand une erreur matérielle ou logicielle survient ou quand le programmeur appelle une interruption.

Q3) Qu'est-ce qu'un "déroutement" ? Donnez en un exemple. (1. pt)

Le déroutement est un type d'interruption interne où le contrôle passe au système d'exploitation pour traiter l'interruption. Exemple : Un défaut de page (tentative d'accès à une page n'existant pas en mémoire)

Exercice 2 : (Ordonnancement : 5 pts) (20 minutes)

Six étudiants du département Informatique viennent voir l'enseignant X (nous tenons à garder son anonymat). L'enseignant X dispose d'une heure pour discuter avec les étudiants et passer d'un étudiant à un autre lui prend une minute. La table suivante donne l'ordre d'arrivée des étudiants et le temps dont ils ont besoin. Les étudiants arrivent tous au même moment.

Etudiant	Ordre d'arrivée	Temps requis
e1	1	45 min
e2	2	15 min
e3	3	10 min
e4	4	5 min
e5	5	6 min
e6	6	20 min

Q1) Sachant que l'enseignant X ne peut consacrer en tout qu'**une heure** à l'ensemble des étudiants, quel est l'algorithme d'ordonnancement qui lui permettra de traiter **le plus d'étudiants complètement**.

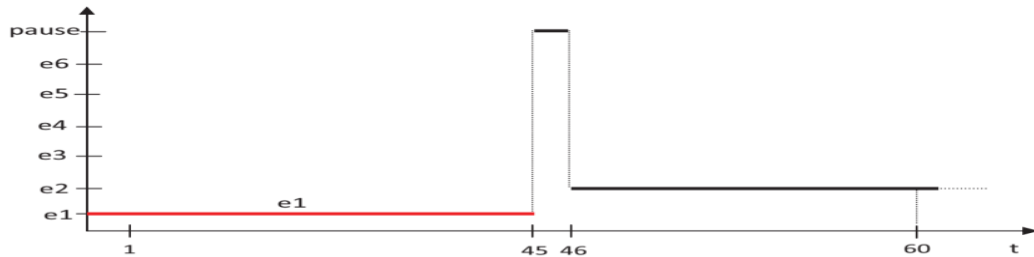
Justifier à l'aide des diagrammes.

Le plus court d'abord (0.5 pts)

Justification :

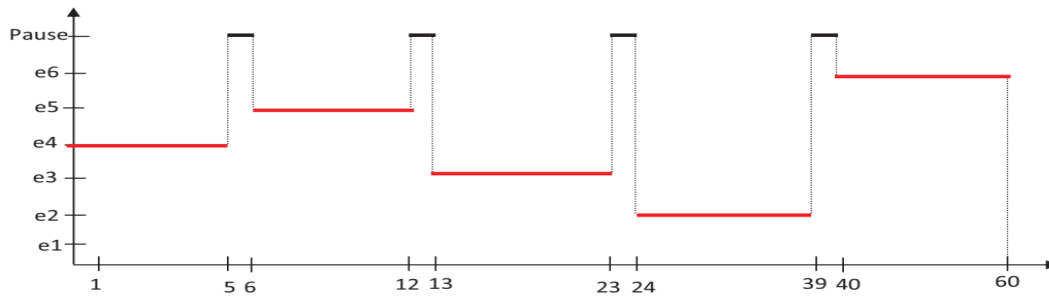
a) premier arrivé, premier servi (FCFS) **(1. pt)**

Premier arrivé, premier servi : 1 étudiant complètement traité



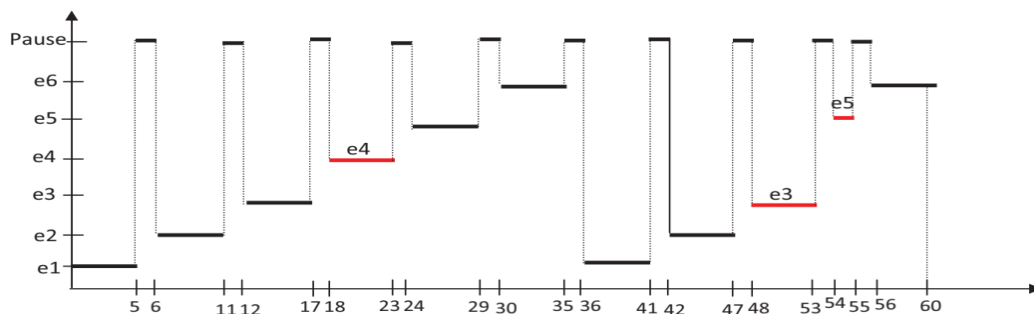
b) le plus court d'abord (SJF) **(1. pt)**

Le plus court d'abord : 5 étudiants complètement traités



c) Tourniquet ou Round Robin avec un quantum = 5min. (l'intervalle de **1min** reste valable même si l'enseignant X reste avec le même étudiant pendant 2 quantum de suite) **(1. pt)**

Tourniquet (quantum = 5mn) : 3 étudiants complètement traité.



Q2) Un ordonnancement de type tourniquet peut utiliser différents quantum. Donner une raison d'utiliser un petit quantum ainsi qu'une raison d'utiliser un grand quantum. **(0.5 pts)**

Un petit quantum assure un temps de réponse rapide (plus d'interactivité). Un grand quantum minimise le temps de changement de contexte (augmente le rendement du pc).

Exercice 3 : (Allocation de mémoire : 4 pts) (20 minutes)

Soit un système à partitions variables de mémoire avec allocation contigüe. A un instant donné les partitions libres sont 100K, 500k, 200K, 300k et 600K (par ordre croissant des adresses). On considère une liste d'arrivée des processus qui demandent 212K, 417K, 112K et 426K.

Si aucun espace mémoire n'est suffisant pour contenir le bloc à allouer, la mémoire est compactée. Si, après compactage, l'allocation n'est toujours pas possible, alors l'allocation est refusée.

Q1) Donner le comportement des algorithmes selon les stratégies First Fit, Best Fit et Worst Fit.

Etape initial (la mémoire vide) : 100K, 500k, 200K, 300k et 600K

100	500	200	300	600
-----	-----	-----	-----	-----

On a les processus ; 212K (A) , 417K (B) , 112K (C) et 426K (D).

Le comportement des algorithmes :

First Fit (1. pt)

Arrivée du A (212k)

100	A	288	200	300	600
-----	---	-----	-----	-----	-----

Arrivée du B (417k)

100	A	288	200	300	B	183
-----	---	-----	-----	-----	---	-----

Arrivée du C (112k)

100	A	C	176	200	300	B	183
-----	---	---	-----	-----	-----	---	-----

Arrivée du D (426k)

On a recours au compactage : Donc, après compactage on a :

A	C	B	959
---	---	---	-----

A	C	B	D	533
---	---	---	---	-----

Best Fit (1. pt)

Arrivée du A (212k)

100	500	200	A	88	600
-----	-----	-----	---	----	-----

Arrivée du B (417k)

100	B	83	200	A	88	600
-----	---	----	-----	---	----	-----

Arrivée du C (112k)

100	B	83	C	88	A	88	600
-----	---	----	---	----	---	----	-----

Arrivée du D (426k)

100	B	83	C	88	A	88	D	174
-----	---	----	---	----	---	----	---	-----

Worst Fit (1. pt)

Arrivée du A (212k)

100	500	200	300	A	388
-----	-----	-----	-----	---	-----

Arrivée du B (417k)

100	B	83	200	300	A	388
-----	---	----	-----	-----	---	-----

Arrivée du C (112k)

100	B	83	200	300	A	C	276
-----	---	----	-----	-----	---	---	-----

Arrivée du D (426k)

On a recours au compactage : Donc, après compactage on a :

B	A	C	959
---	---	---	-----

B	A	C	D	533
---	---	---	---	-----

Q2) Quel est le meilleur algorithme dans ce cas ? (1. pt)

Le meilleur algorithme est le best fit parce que tous les processus sont logés en mémoire centrale sans recours au compactage.

Exercice 4 : (Gestion de la mémoire : 4 pts) (20 minutes)

On considère un système utilisant la technique de pagination et ayant les caractéristiques suivantes :

- Une mémoire virtuelle a une capacité de 2^{46} octets organisés en pages de 512 octets.
 - La mémoire physique qui supporte cette mémoire virtuelle a une capacité de 1 méga-Octets.
 - Une entrée de la table des pages est de la forme : $|n|4|1|$ où
 - ✓ n est nombre de bits pour coder un cadre de page (une case)
 - ✓ 4 est le nombre de bits pour coder la date de chargement de la page
 - ✓ 1 est le bit d'absence/présence
- ➔ Chaque entrée de la table contient donc $n+5$ bits

Répondez aux questions suivantes en justifiant toujours votre réponse :

Q1) Déterminer la valeur de n ? (1.5 pts)

Taille d'un cadre = Taille d'une page = 512 Octets = 2^9 Octets

Taille de la Mémoire Physique (MP) = 1 MO = 2^{20} Octets

Taille de la Mémoire Physique (MP) = nbr cadres * taille d'un cadre

$$\Rightarrow \text{nbr cadres} = \text{nb cadres} = \frac{\text{Taille de la Mémoire Physique (MP)}}{\text{taille d'un cadre}} = \frac{2^{20}}{2^9} = 2^{11} \text{ cadres}$$

On a n est nombre de bits pour coder un cadre de page (une case)

$$\Rightarrow \text{nbr cadres} = 2^n = 2^{11} \rightarrow n = 11$$

Q2) Quelle est la taille de la table de pages ? (1.5 pts)

$$\text{Taille MV} = \text{taille d'une page} * \text{nbr de pages} \rightarrow \text{nbr de pages} = \frac{\text{Taille MV}}{\text{taille d'une page}} = \frac{2^{46}}{2^9} = 2^{37} \text{ pages}$$

$$\text{Taille de la table des pages (TP)} = \text{taille d'une entrée de la TP} * \text{nb entrées dans la TP}$$

$$\text{Taille de la table des pages (TP)} = \text{taille d'une entrée de la TP} * \text{nbr de pages}$$

$$\text{Taille de la table des pages (TP)} = (n+5) * 2^{37} = 16 * 2^{37} = 2^4 * 2^{37}$$

$$\text{Taille de la table des pages (TP)} = 2^{41} \text{ Octets} = 2 \text{ TO}$$

Q3) Quelle est la taille (en bit) du bus d'adresse de ce système ? (1. pt)

$$\text{Taille d'un bus d'adresses} = \text{nb de bits pour coder la mémoire virtuelle} = 46 \text{ bits}$$

Exercice 5 : (Algorithmes de remplacement de pages : 4 pts) (20 minutes)

On considère un système qui utilise un mécanisme de pagination à la demande pour la gestion de la mémoire. Soit un programme possédant un espace virtuel de **512 Octets** et qui fait référence, durant son exécution, aux adresses virtuelles suivantes :

34, 205, 123, 510, 145, 456, 345, 412, 10, 258, 12, 234, 336, 412

Sachant que la taille de la mémoire physique est de **192 Octets** et la taille de page est de **64 Octets**,

Q1) Donnez la suite des numéros de pages référencés ? (1. pt)

$$\text{Nombre de pages MV} = 512/64 = 8 \text{ pages}$$

$$\text{Nombre de cadres MP} = 192/64 = 3 \text{ cadres}$$

Donc, il suffit de diviser l'adresse par 64, ce qui donne la suite des numéros de pages référencés suivante :

0, 3, 1, 7, 2, 7, 5, 6, 0, 4, 0, 3, 5, 6

Q2) Déterminez le nombre de défauts de page générés en applique les algorithmes de remplacement **FIFO, OPTIMAL, LRU, et FIFO de la seconde chance.**

(0.75 pts) FIFO : le nombre de défauts de page = 12														
La chaîne de références	0	3	1	7	2	7	5	6	0	4	0	3	5	6
Cadre 1	0	0	0	7	7	7	7	6	6	6	6	3	3	3
Cadre 2		3	3	3	2	2	2	2	0	0	0	0	5	5
Cadre 3			1	1	1	1	5	5	5	4	4	4	4	6
Défaut de page	D	D	D	D	D		D	D	D	D		D	D	D

(0.75 pts) OPTIMAL : le nombre de défauts de page = 10

La chaîne de références	0	3	1	7	2	7	5	6	0	4	0	3	5	6
Cadre 1	0	0	0	0	0	0	0	0	0	0	0	3	3	3
Cadre 2		3	3	3	2	2	2	6	6	4	4	4	4	4
Cadre 3			1	7	7	7	5	5	5	5	5	5	5	6
Défaut de page	D	D	D	D	D		D	D		D		D		D

(0.75 pts) LRU: le nombre de défauts de page = 12

La chaîne de références	0	3	1	7	2	7	5	6	0	4	0	3	5	6
Cadre 1	0	0	0	7	7	7	7	7	0	0	0	0	0	6
Cadre 2		3	3	3	2	2	2	6	6	6	6	3	3	3
Cadre 3			1	1	1	1	5	5	5	4	4	4	5	5
Défaut de page	D	D	D	D	D		D	D	D	D		D	D	D

(0.75 pts) FIFO de la seconde chance : le nombre de défauts de page = 12

La chaîne de références	0	3	1	7	2	7	5	6	0	4	0	3	5	6
Cadre 1	0 ¹	0 ¹	0 ¹	7 ¹	7 ¹	7 ¹	7 ¹	6 ¹	6 ¹	6 ¹	6 ¹	3 ¹	3 ¹	3 ¹
Cadre 2		3 ¹	3 ¹	3 ⁰	2 ¹	2 ¹	2 ¹	2 ⁰	0 ¹	0 ¹	0 ¹	0 ⁰	5 ¹	5 ¹
Cadre 3			1 ¹	1 ⁰	1 ⁰	1 ⁰	5 ¹	5 ⁰	5 ⁰	4 ¹	4 ¹	4 ⁰	4 ⁰	6 ¹
Défaut de page	D	D	D	D	D		D	D	D	D		D	D	D

Bon courage