Gestion de la MC– Exercices corrigés

Par **R. ZITOUNI**
Centre Universitaire de Bordj Bou Arreridj.

Exercices

Exercice 1 :
-Soit un système ayant 4 cases mémoire, la taille d'une page = 100. - Un Programme P fait successivement référence aux adresses suivantes : 100, 210, 355, 120,420, 110, 200, 550, 139, 201, 395, 404, 505.

1) Donner la chaîne de références aux pages qui correspondent aux adresses.
2) Calculer le nombre de défauts de pages en appliquant la stratégie MFU.
3) Proposer une méthode qui donne un nombre minimum (optimal) de défauts de pages. Et calculer le nombre de défauts de pages optimal.

Exercice 2 :
A) Considérez une mémoire virtuelle ayant une taille de mémoire physique (principale) de 32 Méga-Octets et supportant une taille de blocs de 512 octets. Aussi, supposez un processus occupant un espace d'adressage logique de 856 Kilo-octets.

1) Calculez le nombre de pages dans l'espace d'adressage logique, et le nombre de cases de l'espace d'adressage physique.
2) Montrez les formats des adresses logiques et physiques. Spécifiez le nombre de bits pour les déplacements, les pages, et les cases.
3) Pour l'adresse logique (virtuelle) 11301, spécifiez son emplacement dans la mémoire physique. Supposez que la page contenant l'adresse 11301 se trouve dans la case 15240.
4) Est-ce possible que le nombre de pages requis par un processus soit plus élevé que le nombre de case disponible pour exécuter le processus en mémoire physique (MC) ? Expliquez.

B) Considérez la séquence de référence de page 0,1,2,1,2,1,2,1,2,3,4,5, 6, 5, 6, 7 pour un processus.

1) En utilisant le modèle FIFO, représentez l'allocation en mémoire physique des pages pour une taille de 3 cases (ou cadres).
2) Estimez le taux de fautes de pages.

Exercice 3 :

Considérez un système de mémoire virtuelle ayant les caractéristiques suivantes :
- Taille d'une page et d'un cadre (ou cases) = 1 KO (1 kilo-octet).
- Taille de la mémoire physique (principale) = 32 MO (32 méga-octets).
- Taille de la mémoire virtuelle = 512 MO.
- Utilisation combinée des techniques de pagination et de segmentation : l'espace d'adressage virtuel d'un processus est composé de segments contigus. Chaque segment peut contenir entre 1 et 128 pages. La numérotation des pages d'un segment est relative au segment.

- Utilisation de l'algorithme de remplacement de pages LRU (i.e. la moins récemment utilisée).

1) Calculez le format d'une adresse virtuelle et le format d'une adresse physique (i.e. réelle), en spécifiant le nombre de bits réservés pour chaque champ.
2) Supposez un processus de 9 KO de segment de code et 3 KO de segment de données. Dans l'espace virtuel du processus, le segment de code est suivi du segment de données. Par conséquent, le segment de code débute à l'adresse 0 alors que celui des données débute à l'adresse 9216 relativement au début de l'espace d'adressage virtuel.
    a. Calculez l'adresse qu'occupe en mémoire principale une donnée se trouvant à l'adresse 10728, relative au début de l'espace d'adressage. Le segment de données du processus est chargé au complet en mémoire physique dans les cadres (cases) contigus 4096, 4097 et 4098.
3) Considérez la séquence de références de pages de code R={0, 1, 0, 1, 2, 3, 4, 2, 3, 4, 5, 6, 7, 8} faite par le processus décrit en (2). Les opérandes référés par les instructions dans les pages 0, 1 et 2 se trouvent dans la page 0 du segment de données ; les opérandes des instructions des pages 3, 4 et 5 sont dans la page 1 ; les opérandes des instructions des pages 6, 7 et 8 sont dans la page 2. Supposez que toutes les instructions du processus ont des opérandes qui réfèrent en mémoire. Au départ, 4 cadres contigus sont alloués pour le code du processus à l'adresse X et 2 cadres contigus pour les données du processus à l'adresse Y. Il est à noter que les adresses X et Y ne sont pas nécessairement contiguës, et le chargement des pages dans les cadres alloués est réalisé à la demande (aucun chargement préalable). De plus, aucun cadre supplémentaire n'est alloué au processus durant son exécution.
    a. Représentez l'état d'occupation de la mémoire principale à chaque instant t_i (i.e. t_0, t_1, t_2, …) où une nouvelle page est chargée.
    b. Calculez le nombre de défauts de pages générées par l'algorithme de remplacement de pages LRU. Ce nombre est-il optimal ?

Exercice 4 :

Une firme bien connue concurrente de Apple décide de lancer un nouvel appareil qui permet d'écouter de la musique de format mp3 et d'en « stocker » plus de 80 Go, soit 40Go de plus que le fameux « iPod ». Cependant, avant de faire l'envoi sur le marché, elle décide de vous engager pour vérifier si leurs choix technologiques ont été judicieux pour ce type d'application.

Ce fameux appareil possède 64 Mo de mémoire principale et 80 Go de mémoire secondaire. Il peut en tout temps n'exécuter que trois processus (en plus du SE) sur son processeur embarqué.

Le processus le plus important parmi ces trois est l'Afficheur chargé en mémoire dès le départ (désigné par A). De plus, un processus Transfert (nommé T) permet de transférer les données d'un ordinateur à sa mémoire secondaire à partir d'un port externe de type « USB ». Finalement, le dernier processus Son (appelé S) permet d'envoyer de la musique à partir des données de la mémoire secondaire vers un port externe d'écouteur. Il est important de savoir que l'écoute de la musique est la principale fonction de cet appareil. L'affichage et le son doivent toujours être parfaits, tandis que le transfert de données n'est que secondaire.

Cette firme cherche une solution quasi-idéale pour la disposition de la mémoire principale. Elle désire obtenir une disposition qui donne une excellente rapidité d'accès et surtout le moins de fragmentation externe possible.

Elle vous spécifie que

- 16Mo sont réservés pour le système d'exploitation.

- L'afficheur nécessite 500Ko de mémoire de façon stable.
- Le processus de Son a besoin initialement de 500Ko mais sachant qu'une chanson nécessite entre 4Mo et 10Mo, il aura besoin d'un espace contigu supplémentaire (pour éviter des sauts dans la musique).
- le transfert prend initialement 5Mo mais nécessite des tampons dynamiques pour faciliter le transfert.

1- Proposez une organisation pour la mémoire principale de manière à satisfaire tous ces critères (référez-vous à l'énoncé de l'exercice pour d'autres informations). Justifiez votre proposition.
2- Quel type d'adressage proposeriez-vous? (Relatif ou absolu). Donnez les avantages de votre choix et expliquer les dispositifs à utiliser pour assurer le bon fonctionnement du système (protection du système).

Exercice 5 :

Considérez un système de gestion de mémoire qui a les caractéristiques suivantes :

- Un adressage virtuel sur 32 bits
- Une taille de Page de 4Ko
- Une mémoire physique de 1 Mo

a) Supposez que le système utilise la segmentation paginée et que l'adresse virtuelle est de la forme :

Quelles sont les données manquantes à ce problème pour traduire l'adresse virtuelle de 32 bits suivante : (hexadécimal) AE854C9C en adresse physique ?
Si vous aviez ces informations, identifiez brièvement les étapes à suivre pour effectuer cette translation.

b) Supposez que le système utilise une pagination à deux niveaux, où les entrées des tables de pages sont sur 4 octets. La structure de l'adresse virtuelle est illustrée par la figure suivante :

b.1) Si un processus utilise tout l'espace adressable qui lui est fourni, combien de pages seront elles nécessaires pour contenir toutes les tables de pages de ce processus.
b.2) Un second processus nécessite 22Mo pour s'exécuter entièrement (son code, ses données, pile…). La partie contenant son code est disposée dans sa mémoire virtuelle aux adresses suivantes [2Mo à 6Mo-1], les données sont quant à elles dans l'intervalle [12Mo-21Mo-1]. Si nous devons charger les tables de pages associées à ces deux parties, combien de pages de niveaux 2 seront chargées en mémoire centrale.
c) Supposez maintenant que le système utilise une pagination pure et que les bits 12 à 31 correspondent à un numéro de page. Si nous utilisons une table inversée, combien d'entrées la table de pages inversée contiendra-t-elle ?

Exercice 6 :

Un système qui implémente la pagination à la demande dispose de 4 cases de mémoire physique qui sont toutes occupées, à un instant donné, avec des pages de mémoire virtuelle. La table 3 donne, pour chaque case de mémoire, le moment du chargement de la page qu'elle contient ($t_{chargement}$), le temps du dernier accès à cette page ($t_{dernier\ accès}$) et l'état des bits référencé (R), modifié (M) et présence (P). Les temps sont donnés en tops d'horloge.

Case	$t_{chargement}$	$T_{dernier\ accès}$	R	M	P
0	126	270	0	0	1
1	230	255	1	0	1
2	110	260	1	1	1
3	180	275	1	1	1

Indiquez quelle est la page qui sera remplacée en cas d'un défaut de page si l'algorithme de remplacement de page est :

a) LRU
b) FIFO

3. On considère un système avec une mémoire virtuelle segmentée paginée où la taille d'une page est de 4Ko et une mémoire physique de 64Ko. L'espace d'adressage d'un processus P est composé de trois segments S1, S2 et S3 de taille, respectivement 16Ko, 8Ko et 4Ko. À un moment donné, pour le processus P, les pages 2 et 3 du segment S1, la page 2 du segment S2 et la page 1 du segment S3 sont chargées en mémoire physique, respectivement dans les cases 2, 0, 9, 12.
Pour une donnée située dans l'espace d'adressage du processus P à l'adresse décimale 8212, **indiquez** :

a) le segment
b) le numéro de page dans le segment.
c) le déplacement dans la page
d) le numéro de case.
e) le déplacement dans la case.
f) l'adresse physique.

Exercice 7 :

I) Soit un programme dont le code occupe 1024 octets en mémoire et qui utilise un vecteur avec 1000 éléments de type caractère (un caractère = un octet en mémoire). Ce programme est exécuté dans un système qui utilise la pagination de la mémoire dont la taille de la mémoire réelle est de 1 Mo, la taille d'une page est de 512 octets et les instructions à référence mémoire ont un champ d'adresse de 24 bits.

a) Donnez :

1) la taille de l'espace logique d'adressage
2) le nombre de bits du déplacement
3) le nombre de bits du numéro de page virtuelle
4) le nombre de bits d'une adresse réelle
5) le nombre de bits du numéro de page réelle (case)
6) le nombre d'entrées de la table des pages

b) Le chargement de ce programme en mémoire engendre-t-il une fragmentation interne? Justifiez votre réponse.

II) Donnez le format d'une adresse virtuelle de 32bits avec des pages de 256 octets et une table des pages à trois niveaux.
Si toutes les tables ont le même nombre d'entrées, donnez le nombre de tables ainsi que leur nombre d'entrées.

III) On appelle "anomalie de Belady" le fait que le nombre de défauts de pages augmente quand on augmente le nombre de cadres de pages disponibles en mémoire physique. On appelle "algorithme à pile" un algorithme présentant la propriété d'inclusion :
M(m,r) inclus dans M(m+1, r),
où m est le nombre de cadres de mémoire et r un index dans le vecteur de références aux pages (ω). M(m,r) représente l'ensemble des pages présentes en mémoire après r références mémoire, lorsque l'on dispose de m cadres de page. Une condition suffisante pour qu'un algorithme ne présente pas l'anomalie de Belady est qu'il soit à pile.

a) Montrez l'anomalie de Belady pour l'algorithme FIFO avec m=3, puis m=4, pour le vecteur de références aux pages suivant : ω = { 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5 }.
b) Montrez sur cet exemple que l'algorithme FIFO n'est pas un algorithme à pile, en indiquant les états pour lesquels la propriété d'inclusion n'est pas vérifiée.

Corrections des exercices

Exercice 1 :

1) la chaîne de Références associée est : 1,2,3,1,4,1,2,5,1,2,3,4,5.
2) Politique MFU

Demandes	***1***	***2***	***3***	***1***	***4***	***1***	***2***	***5***	***1***	***2***	***3***	***4***	***5***
Case 1	***1***	*1*	*1*	*1*	*1*	*1*	*1*	***5***	*5*	*5*	*5*	*5*	*5*
Case 2		***2***	*2*	*2*	*2*	*2*	*2*	*2*	***1***	***2***	*2*	*2*	*2*
Case 3			***3***	*3*	*3*	*3*	*3*	*3*	*3*	*3*	*3*	*3*	*3*
Case 4					***4***	*4*	*4*	*4*	*4*	*4*	*4*	*4*	*4*
	D	*D*	*D*		*D*			*D*	*D*	*D*			

7 défauts de pages

3) La méthode qui donne le nombre minimum de défaut de pages est une méthode optimale, elle consiste à choisir une page victime qui ne sera pas référencée dans le futur immédiat.

Demandes	***1***	***2***	***3***	***1***	***4***	***1***	***2***	***5***	***1***	***2***	***3***	***4***	***5***
Case 1	***1***	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	***4***	*4*
Case 2		***2***	*2*	*2*	*2*	*2*	*2*	*2*	*2*	*2*	*2*	*2*	*2*
Case 3			***3***	*3*	*3*	*3*	*3*	*3*	*3*	*3*	*3*	*3*	*3*
Case 4					***4***	*4*	*4*	***5***	*5*	*5*	*5*	*5*	*5*
	D	*D*	*D*		*D*			*D*				*D*	

Pour la méthode optimale 6 défauts de pages.

Exercice 2 :

A)

1) Nombre de pages 856 Ko/512 = 1712 pages
   Nombre de cases 32 Mo/512 = 65536 cases.
2) Adresse logique : 11 bits pour les numéros de pages et 9 bits pour le déplacement.
   Adresse physique 16 bits pour les numéros de cases et 9 bits pour le déplacement.
3) - Adresse de base de la case 15240 alors 15240 * 512 = 7802880
   - Déplacement à partir de l'adresse 11301 alors 11301 Mod 512 = 37.
   - Adresse physique de l'adresse logique 11301 est : 7802880 + 37 = 7802917

4) Avec la mémoire virtuelle, on ne charge jamais tout l'espace d'adressage d'un processus en mémoire physique. Seules les pages dans la localité de référence du processus sont conservées en mémoire physique. Bien souvent, le système d'exploitation n'alloue qu'un nombre de cases souvent, le système d'exploitation n'alloue qu'un nombre de cases inférieur au nombre de pages du processus et même s'il y a de la mémoire physique disponible pour charger tout l'espace d'adressage du processus.

B)

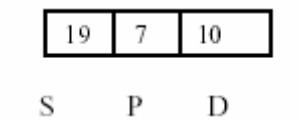
Demandes	***0***	***1***	***2***	***1***	***2***	***1***	***2***	***1***	***2***	***3***	***4***	***5***	***6***	***5***	***6***	***7***
Case 1	***0***	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	***3***	*3*	*3*	***6***	*6*	*6*	*6*
Case 2		***1***	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	***4***	*4*	*4*	*4*	*4*	***7***
Case 3			***2***	*2*	*2*	*2*	*2*	*2*	*2*	*2*	*2*	***5***	*5*	*5*	*5*	*5*
	D	*D*	*D*							*D*	*D*	*D*	*D*			*D*

Le nombre de défauts de page est de 8. le taux par rapport au 16 pages demandées est 8/16 = 50%.

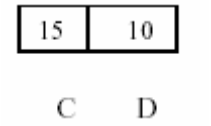
Exercice 3:

1)

. Adresse virtuelle



. Adresse physique



2)

- La page de l'adresse logique 10728 est [(10728-9216)/1024] soit 1.
- Le déplacement à l'intérieur de la page 10 est 10728 déplacement 1024 soit 488.
- La page 1 du segment de données se trouve dans le cadre (case) 4097.
- L'adresse du cadre (case) 4097 est 4097*1024 soit 4195328.
- L'adresse réelle de l'adresse logique 10728 est 4195328+488 soit 4195816.

3)

	t0	t1	t2	t3	t4	t5	t6	t7	t8
		…	…	…	…	…	…	…	…
X	Page 0	Page 0	Page 0	Page 0	Page 4	Page 4	Page 4	Page 4	Page 8
		Page 1	Page 1	Page 1	Page 1	Page 5	Page 5	Page 5	Page 5
			Page 2	Page 2	Page 2	Page 2	Page 6	Page 6	Page 6
				Page 3	Page 3	Page 3	Page 3	Page 7	Page 7
		…	…	…	…	…	…	…	…
Y	Page 0	Page 0	Page 0	Page 0	Page 0	Page 0	Page 2	Page 2	Page 2
				Page 1	Page 1	Page 1	Page 1	Page 1	Page 1
		…	…	…	…	…	…	…	…
		…	…	…	…	…	…	…	…

b. Il y a eu 9 fautes de pages de code et 3 fautes de page de données pour un total de 12 fautes sur 17 références soit un taux de 70.5%.

Vu que la taille du processus = 12 KO = 12 pages différentes, le nombre de fautes optimal est alors de 12 ce qui a été atteint par l'algorithme LRU.

Exercice 4:

1- Les principaux inconvénients de la mémoire à partition fixe est que les partitions sont constantes alors qu'un ordinateur standard peut avoir plusieurs processus de tailles différentes, dans ce cas les partitions variables sont plus adéquates. Cependant, dans le cas présent, il ne peut y avoir que 3 processus de tailles plus ou moins définie et stable. En plus, nous avons assez de mémoire pour satisfaire la majorité des demandes en mémoire des processus. Une solution efficace serait donc **une mémoire a partition fixe sans fil d'attente avec du va-et-vient**, car les processus ont toujours leur place de disponible (on ne peut pas faire deux transferts en même temps ou écouter deux musiques en même temps). On donne 16Mo au SE, 500 Ko à A, 10.5 Mo à S et 37 Mo à T. **T et A pourront** donc utiliser une partie de **leurs mémoires comme tampon de va-et-vient** si c'est nécessaire.

2- Un adressage relatif est choisi, il faut donc utiliser un registre de base, un registre limite et un dispositif de conversion d'adresse qui se charge aussi d'assurer la protection entre les trois processus et aussi avec l'OS.

Exercice 5:

a) Il manque la table des segments qui va nous indiquer la table de pages associée au segment désiré. Cette table des pages nous permettra d'obtenir le cadre associé à notre adresse. L'adresse physique est obtenue en remplaçant les numéros de segment et de page par le numéro de case.

b.1) Il y a 2^{10} tables de pages de second niveau et une table de pages de premier niveau. Chaque table de pages a 2^{10} entrées. Elle occupe donc une page 2^{10} x 4octects = 4096 octets.
REP : 2^{10}+1 pages seront nécessaires pour toutes les tables de pages.
b.2) Chaque entrée de la table des pages de niveau 2 est associée à un cadre de 4Ko (2^{12}). Sachant qu'une table de pages de niveau 2 contient 2^{10} entrées, elle référence 2^{10}* 2^{12} octets =4Mo de la mémoire virtuelle. La première table de niveau 2 référence la mémoire virtuelle comprise entre [0, 4Mo-1], la seconde [4Mo, 8Mo-1] et ainsi de suite…..
Donc :

- pour le code 2 tables de pages de niveaux 2 seront nécessaires (c-à-d 2 pages)
- pour les données 3 tables de pages de niveaux 2 seront nécessaires (c-à-d 3 pages).

c) Le nombre de cadres est : $2^{20} / 2^{12} = 2^8$
1 entrée par cadre (case) => 2^8 entrées.

Exercice 6:

1.
a) Dans l'algorithme LRU, on retire la page la moins récemment utilisée. Il s'agit donc de choisir une page selon le critère de la colonne $t_{\text{dernier accès}}$. La page à retirer est celle chargée dans la case 1, qui a été accédée au temps 255.
b) Dans l'algorithme FIFO, on retire la page qui est en mémoire depuis le plus longtemps. Il s'agit donc de suivre le critère de la colonne $t_{\text{chargement}}$. La page à retirer est celle chargée dans la case 2 qui est en mémoire depuis le temps 110.

2. Le taux d'accès réalisés en 100ns est de 65%. Parmi les 35% accès menant aux défauts de page, 70% ont besoin de 20ms et le reste de 30% ont besoin de 10ms.
$t_{\text{accès moyen}} = 0.65*0.0001 + 0.35*(0.7*20 + 0.3*10) = 5.950065$ mS

3.
a) segment S1
b) page 3
c) déplacement 20
d) case 0
e) déplacement 20
f) L'adresse physique est exprimée sur 16 bits (64Ko = 2^{16}), dont 4 bits pour le numéro de case et 12 bits pour le déplacement dans la case (4Ko = 2^{12}) . Alors, l'adresse physique sera : 0000 0000 0001 0100.

Exercice 7:

I a)
1) 2^{24} = 16Mo
2) 9 bits
3) 24 – 9 = 15 bits
4) 20 bits
5) 20 – 9 = 11 bits
6) 2^{15} = 32Ko

I b) Oui. Le programme a besoin de 2000 octets de données et de 1024 octets de code, donc un total de 3024 octets en mémoire. Le nombre de pages occupées : [3024 / 512] = 6 pages. Dans la dernière page, il reste 512 – 464 = 48 octets libres, ce qui cause la fragmentation interne.

II) Le format PT1(8bits) PT2(8bits) PT3(8bits) Offset(8bits)

Le nombre de tables : 1 + 256 + (256*256). Le nombre d'entrées : 256

III)

a) L'algorithme FIFO pour m=3

Demandes	***1***	***2***	***3***	***4***	***1***	***2***	***5***	***1***	***2***	***3***	***4***	***5***
M0	***1***	*1*	*1*	***4***	*4*	*4*	***5***	*5*	*5*	*5*	*5*	*5*
M1		***2***	*2*	*2*	***1***	*1*	*1*	*1*	*1*	***3***	*3*	*3*
M3			***3***	*3*	*3*	***2***	*2*	*2*	*2*	*2*	***4***	*4*
	D	*D*	*D*	*D*	*D*	*D*	*D*			*D*	*D*	

Il y a 9 défauts de pages.

L'algorithme FIFO pour m=4.

Demandes	***1***	***2***	***3***	***4***	***1***	***2***	***5***	***1***	***2***	***3***	***4***	***5***
M0	***1***	*1*	*1*	*1*	*1*	*1*	***5***	*5*	*5*	*5*	***4***	*4*
M1		***2***	*2*	*2*	*2*	*2*	*2*	***1***	*1*	*1*	*1*	***5***
M3			***3***	*3*	*3*	*3*	*3*	*3*	***2***	*2*	*2*	*2*
M4				***4***	*4*	*4*	*4*	*4*	*4*	***3***	*3*	*3*
	D	*D*	*D*	*D*			*D*	*D*	*D*	*D*	*D*	*D*

Il y a 10 défauts de pages. Anomalie de belady.

b) L'algorithme exemplifié au point a) n'est pas à pile. Les états où l'inclusion n'est pas vérifiée sont les états d'index 7, 8 et 11 dans le vecteur de références aux pages (ω).