TD2: GESTION DES MEMOIRES ET MEMOIRES VIRTUELLES

Nom: Nguyen Sang
Code'etudient: 18126029

Exercice 1: Supposez les partitions de mémoires sont 100k, 500k, 200k, 300k et 600k (dans cet ordre), comment chacun des algorithmes Fist-fit, Best-fit and Worst-fit placerait-il des processus 212k, 417k, 112k et 426k (dans cet ordre). Quel algorithme effectue l'utilisation la plus efficace de la mémoire.

P1 (212k), P2(417k), P3(112K), P4(426K)

- First-fit:

Partitions de mémoires	First-fit	Best-fit	Worst-fit
100K			
500K	P1 (212K)	P2 (417K)	P2 (417K)
200K	P3 (112K)	P3 (112K)	
300K		P1 (212K)	P3 (112K)
600K	P2 (417K)	P4 (426K)	P1(212K)

À mon avis, le algorithme **Best-fit** effectue l'utilisation la plus efficace de la mémoire parce que **Worst-fit** et **First-fit** cela créera deux zones de mémoire discontinues**.** Additionellement, L'algorithme "**Best-fit**" fait que les partitions de mémoire contiennent tous les processus.

Exercice 2: Soit un système utilisant la pagination dont la taille de la page est 1KO. Le programme P est chargé au système. Il nous faut huit pages pour contenir toutes les adresses virtuelles générées par le programme P. La mémoire centrales du système peut se diviser en 32 cadres de pages. Donner le nombre de bits d'une adresse réelle et d'une adresse virtuelle.

+) La taille de la page = 1KO = (1KO/1 octet) = 2^10=> 10 bits (registre de base)
+) 32 cadres de pages = 2^5 => 5 bits (registre de base)
=> Le nombre de bits d'une adresse réelle = 10 + 5 = 15 bits

+) La taille de la page 1KO = (1KO/1 octet) = 2^10=> 10 bits (registre de base)
+) 8 page pour contenir toutes les adresses virtuelles: 2^3 => 3 bits (registre de base)
=> Le nombre de bits d'une adresse virtuelle = 10 + 3 = 13 bits

Exercice 3: Considérer un système ayant les caractéristiques suivantes :
• Adresse logique sur 48 bits
• Taille de la mémoire physique 320 MO
• Taille d'une page 8 KO pour contenir les mots de mémoire de 1 octet
a. Donner le nombre de cadres de pages en mémoire physiques
b. Donner le nombre de pages en espaces d'adressage logique
c. Convertir l'adresse logique 20300 en l'adresse sous forme <p, d>.

a) Le nombre de cadres de pages en mémoire physiques:
320 MO / 8KO = (1024*320)/8 = 40960 cadres de pages

b) La **taille** de la addresse logique sur 48 bits = 2^48
=> ((2^48)/1028) KO/8 KO ≈ 34,22604289x10^9
Mais, 2^34 < 34,22604289x10^9 < 2^35 => Le **nombre de pages** en espaces d'adressage logique = 2^35

c) Taile d'une page est 8 KO = 1024*8 = 8192 (mots memoires)
- L'address logique = 20300
=> p = 20300 / 8192 = 2 (pages)
=> d = 20300 – (8192*2) = 3916
=> le forme <p, d> = <2, 3916>

Exercice 5:
a) Donner le nombre de cadre de pages, nombre de pages dans l'espace d'adressage de chaque processus.
La taille de la mémoire physique = 512 MO
La taille d'une page = 4096 octets = 0.004096 MO
=> Nombre de cadre de pages = 512/(0.004096) = 125000 = 2^17 (pages)
Le système utilise 32 bits pour stocker une adresse logique => 2^32 octets => Le nombre de pages dans l'espace d'adressage de chaque procesus = 2^32/4096 = 2^20 (pages)

b. Expliquer comment le système établit les emplacements physiques correspondant aux les adresses virtuelles générées par les trois processus comme suivant:

• P2 – adresse 13000
• P1 – adresse 13000
• P1 – adresse 16383
• P3 – adresse 4096
• P3 – adresse 13000
• P2 – adresse 16383

Supposons que :

• Le système n'utilise que les cadres de pages : #0, #300, #301, #302 et #400.
• Pour assurer l'efficacité du système, il faut que quatre pages ci-après restant dans la mémoire : page #0, page #2 du processus P1 et P2.
• Lors que les fautes de pages se produisent, le système utilise l'algorithme LRU pour remplacement des pages.

Exercise 6:

+) 3 cadres de pages, **FIFO 2e** :

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
C1	1	1	1	4	4	4	4	6	6	6	6	3	3	3	3	3	1	1	1	1
C2		2	2	2	2	2	5	5	5	1	1	1	1	6	6	6	6	6	3	3
C3			3	3	3	1	1	1	2	2	2	2	7	7	7	2	2	2	2	6
FIFO	1	12	123	234	234	421	145	516	652	261	261	123	317	736	736	362	621	621	213	136
BIT	1	12	123	4	42	41	15	56	62	21	21	13	37	76	763	2	21	21	213	6
Défaut de pages	*	*	*	*		*	*	*	*	*		*	*	*		*	*		*	*

=> 16 défauts de pages

+) 3 cadres de pages, **LRU**:

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
C1	1	1	1	4	4	4	5	5	5	1	1	1	7	7	7	2	2	2	2	2
C2		2	2	2	2	2	2	6	6	6	6	3	3	3	3	3	3	3	3	3
C3			3	3	3	1	1	1	2	2	2	2	2	6	6	6	1	1	1	6
Défaut de pages	*	*	*	*		*	*	*	*	*		*	*	*		*	*			*

=> 15 défauts de pages

+) 3 cadres de pages, **OPTIMAL**:

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
C1	1	1	1	1	1	1	1	1	1	1	1	3	3	3	3	3	3	3	3	6
C2		2	2	2	2	2	2	2	2	2	2	2	7	7	7	2	2	2	2	2
C3			3	4	4	4	5	6	6	6	6	6	6	6	6	6	1	1	1	1
Défaut de pages	*	*	*	*		*	*	*	*	*		*	*	*		*	*			*

=> 10 defauts de pages

+) 4 cadres de pages, **FIFO 2e** :

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
C1	1	1	1	1	1	1	5	5	5	5	5	3	3	3	3	3	1	1	1	1
C2		2	2	2	2	2	2	6	6	6	6	6	7	7	7	7	7	7	3	3
C3			3	3	3	3	3	3	2	2	2	2	2	6	6	6	6	6	6	6
C4				4	4	4	4	4	4	1	1	1	1	1	1	2	2	2	2	2
FIFO	1	12	123	1234	1234	1234	2345	3456	4562	5621	5621	6213	2137	1376	1376	3762	7621	7621	6213	6213
BIT	1	12	123	1234	1234	1234	5	56	562	5621	5621	3	37	376	376	3762	1	12	13	136
Défaut de pages	*	*	*	*			*	*	*	*		*	*	*		*	*		*	*

=> 15 défauts de pages

+) 4 cadres de pages – **LRU:**

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
C1	1	1	1	1	1	1	1	1	1	1	1	1	1	6	6	6	6	6	6	6
C2		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
C3			3	3	3	3	5	5	5	5	5	3	3	3	3	3	3	3	3	3
C4				4	4	4	4	6	6	6	6	6	7	7	7	7	1	1	1	1
Défaut de pages	*	*	*	*			*	*				*	*	*			*			

=> 10 défauts de pages

+) 4 cadres de pages – **OPTIMAL:**

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
C1	1	1	1	1	1	1	1	1	1	1	1	1	7	7	7	7	1	1	1	1
C2		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
C3			3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
C4				4	4	4	5	6	6	6	6	6	6	6	6	6	6	6	6	6
Défaut de pages	*	*	*	*			*	*					*				*			

=> 8 défauts de pages

Exercice 7: Soit un programme de la taille 1067 KO chargé dans une mémoire réelle de 16 MO. Ce programme s'exécute dans un système utilisant la pagination dont la taille de page est 32 KO, la taille de chaque mot de mémoire de la page est 1 octet. Le système utilise 25 bits pour contenir une adresse du processus.

a. Donner le nombre de bits du déplacement

b. Donner le nombre de pages possibles et de cadre de pages

c. Lorsque ce programme est chargé en mémoire centrale, est-ce qu'il peut causer le problème de fragmentation ? Quel type de fragmentation se produit?

a) La taille de page = 32 KO = 2^15 => Le nombre de bits du déplacement est 15 bits

b)

+) Le nombre de pages possibles = 2^(25-15) = 2^10

+) Une mémoire réelle de 16 MO => (16 MO / 1 octet) = 2^24 = 24 bits (registre de base)

=> Le nombre de cadre de pages = 2^(24) – 2^(15) = 2^9