TELECOM Nancy

RSA : Réseaux et Systèmes Avancés
Deuxième année

UNIVERSITÉ DE LORRAINE

TD3 : la mémoire virtuelle (2/2)

★ Exercice 1. Segmentation paginée

Soit une mémoire segmentée paginée pour laquelle les cases en mémoire centrale sont de 4 Ko. La mémoire centrale compte au total 16 cases numérotées de 0 à 15. On considère 2 processus A et B.

Le Processus A a un espace d'adressage composé de trois segments S1A, S2A et S3A qui sont respectivement de 8 Ko, 12 Ko et 4 Ko. Seules les 2 premières pages du segment S1A, seule la 2ème page du segment S2A et la 1ère page du segment S3A sont chargées en mémoire centrale respectivement dans les cases 3, 12, 1 et 6.

Le processus B a un espace d'adressage composé de deux segments S1B et S2B qui sont respectivement de 16 Ko et 8 Ko. Seules les 2 dernières pages du segment S1B et la 1ère page du segment S2B sont chargées en mémoire centrale respectivement dans les cases 10, 0 et 15.

Remarque. la numérotation (désignation) des pages (comme pour les cases) commence à 0.

▷ **Question 1.** Représenter sur un dessin la table des segments (taille,segment) et les tables des pages (case,v/i) pour chaque processus ainsi que la mémoire centrale correspondant à l'allocation décrite.

▷ **Question 2.** Proposer un schéma de MMU capable de faire la conversion des adresses logiques en adresses physiques de ce système.

▷ **Question 3.** Soit 12292, une adresse linéaire pour A, déterminez l'adresses virtuelle correspondantes sous le format (segment,page,déplacement) et (segment,déplacement). Que produit la MMU au final ?

▷ **Question 4.** Soit 2098, une adresse linéaire pour B, déterminez l'adresse virtuelle correspondante sous le format (segment,page,déplacement) et (segment,déplacement). Dans ce cas, que génère la MMU ? Quel est le traitement associé ?

★ Exercice 2. Algorithmes de remplacement de pages

On considère une mémoire contenant 3 cadres de page et une mémoire virtuelle constituée de 5 pages (numérotées de 0 à 4). Les pages sont appelées comme suit : 0-L, 1-E, 2-L, 3-L, 4-E, 1-E, 2-L, 4-L, 0-E, 1-L (L pour lecture et E pour écriture). Donnez l'état des pages et des cadres pour chaque algorithme.

▷ **Question 1.** Algorithme FIFO

Page appelée	0-L	1-E	2-L	3-L	4-E	1-E	2-L	4-L	0-E	1-L
Cadre 1	0	0	0	3	3	3	2	2	2	2
Cadre 2		1	1	1	4	4	4	4	0	0
Cadre 3			2	2	2	1	1	1	1	1
Défauts de page	1	2	3	4	5	6	7	7	8	8

▷ **Question 2.** Algorithme optimal.

Page appelée	0-L	1-E	2-L	3-L	4-E	1-E	2-L	4-L	0-E	1-L
Cadre 1	0	0	0	3	4	4	4	4	0	0
Cadre 2		1	1	1	1	1	1	1	1	1
Cadre 3			2	2	2	2	2	2	2	2
Défauts de page						5	5	5	6	6

▷ **Question 3.** Algorithme LRU (Least Recently Used).

Page appelée	0-L	1-E	2-L	3-L	4-E	1-E	2-L	4-L	0-E	1-L
Cadre 1	0	0	0	3	3	3	2	2	2	1
Cadre 2		1	1	1	4	4	4	4	4	4
Cadre 3		2	2	2	1	1	1	1	0	0
Défauts de page	1	2	3	4	5	6	7	7	8	9

▷ **Question 4.** Algorithme de l'horloge (ie, de la seconde chance).

Page appelée	0-L	1-E	2-L	3-L	4-E	1-E	2-L	4-L	0-E	1-L
Cadre 1										
Cadre 2										
Cadre 3										
Défauts de page										
R de page 0										
R de page 1										
R de page 2										
R de page 3										
R de page 4										

▷ **Question 5.** Algorithme NRU (Not Recently Used).

Page appelée	0-L	1-E	2-L	3-L	4-E	1-E	2-L	4-L	0-E	1-L
Cadre 1										
Cadre 2										
Cadre 3										
Défauts de page										
M et R de cadre 1										
M et R de cadre 2										
M et R de cadre 3										

★ Exercice 3. Impact de la structure des programmes

On considère un système à mémoire paginée où la taille d'une page est de `200*sizeof(int)`. Un petit processus occupe la page 0 de la mémoire. Il reste 3 cadres de pages libres à l'instant où un programme arrive. Ce dernier initialise le contenu du tableau A défini par :

```
int A[][] = new int[100][100];
```

▷ **Question 1.** Sachant que ce programme est chargé dans la cadre de page numéro 1 et que les 2 autres pages restantes sont initialement libres, quel est le nombre de défauts de pages expérimenté avec un remplacement LRU et ce pour les deux programmes d'initialisation suivants :

```
for (int j = 0; j < 100; j++)
   for (int i = 0; i < 100; i++)
      A[i][j] = 0;
```

50000 défauts de pages

(50 × 100)

```
for (int i = 0; i < 100; i++)
   for (int j = 0; j < 100; j++)
      A[i][j] = 0;
```

Début de page toute les 2 lignes : 500 défauts de page au total.