
TD 4

Gestion de la mémoire - Segmentation et pagination
semaine du 23/11/98

Objectifs : Etudier la gestion d'une mémoire segmentée et le principe de la pagination.

SEGMENTATION

La segmentation divise l'ensemble des informations nécessaires à l'exécution d'une tâche en segments. Un segment, de longueur variable, représente une zone de mémoire pour la tâche. Il est associé à un aspect spécifique de la représentation de la tâche en mémoire (code, données, pile, ...). Un segment est chargé entièrement dans des zones de mémoire contiguës. La mémoire centrale est allouée par granules de **G** kilo-mots.

Dans ce TD, on se propose d'étudier l'implémentation d'une allocation de mémoire segmentée.

1) Considerons le système étudié au TD précédent (cf. TD 6) où le système dispose d'un registre de base B, et d'un registre de limite L...

1.1) Ce système, tel quel, permet-il l'implémentation d'une allocation de mémoire segmentée ? Justifier.

1.2) On veut pouvoir utiliser dans le même programme plusieurs segments de code et plusieurs segments de données. Quelle opération doit faire le système lors des changements de segment ? De quelles données le système doit-il disposer ?

1.3) Pour générer des programmes performants, on veut aussi disposer d'un adressage indirect dans tout l'espace de données du processus (rappel : adressage indirect = on lit en mémoire l'adresse de la donnée à lire). Quelle condition la machine doit-elle satisfaire pour cela ?

2) On considère à nouveau les tâches du TD 6, réalisées avec une allocation de mémoire segmentée. Chaque tâche comporte 4 segments : CODE, DATA (données non initialisées), BSS (données initialisées), PILE. La taille des segments est indiquée dans le tableau suivant :

Tâche	T1	T2	T3	T4
Taille de CODE en Kmots	30	20	60	40
Taille de DATA en Kmots	35	40	25	60
Taille de BSS en Kmots	5	0	5	20
Taille de PILE en Kmots	30	20	30	50
Instant de soumission	0	2	5	8
Durée	7	22	13	13

La mémoire est allouée dans l'ordre CODE, DATA, BSS avec la stratégie du premier choix (First fit).

2.1) Représentez l'état de la mémoire au temps 9.

2.2) Quel gain apporte la segmentation ? Un algorithme de compactage est-il encore nécessaire ?

PAGINATION

La pagination consiste à diviser l'adressage de la mémoire en deux couches :

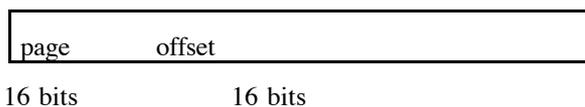
- La couche logicielle utilise un espace d'adressage plus grand que la mémoire physique effectivement disponible (l'espace des adresses linéaires, autrement appelé *mémoire virtuelle* ou *mémoire linéaire*). Par exemple l'allocateur de mémoire attribue aux processus ou aux segments de la mémoire virtuelle.
- La mémoire linéaire est divisée en pages. A un moment donné, certaines pages sont en mémoire physique, d'autres sont sauvegardées sur disque. Une couche matérielle pilotée par le SE (grâce aux tables de pages) est chargée de lire les données qui sont en mémoire, charger en mémoire les pages demandées absentes, décharger les pages inutilisées pour faire de la place.

Prérequis

1. Si le bus adresse fait 64 bits, quelle est la taille maximale de la mémoire centrale ?
2. Ecrire 2^{16} sous forme d'un nombre hexadécimal.

Exercice 1

Un processus a une table des pages de 5 entrées. Chaque entrée a 1 octet de longueur. Une adresse virtuelle est donnée par le schéma suivant:



1.1 Quelle est la capacité maximale de la mémoire centrale?

1.2 Il reste au moment du chargement du processus 5 cases disponibles, les cases (ou cadres) 0Ah, 1Ah, 1Dh, 20h, 21h.

Décrire la table des pages après le chargement en mémoire du processus. Les cases libres sont utilisées dans l'ordre croissant.

1.3 Quelles sont les adresses effectives correspondantes aux adresses virtuelles 00000000h, 000200FEh, 000404FFh, 00050300h?

1.4 On suppose maintenant que le processus a 6 pages. Quelle est l'adresse virtuelle correspondant à l'adresse 00050300h ?

Exercice 2

Sur un ordinateur, une adresse virtuelle est exprimée sur 32 bits. La mémoire réelle fait 64 méga-octets découpée en cases de 4Koctets. (1 méga-octets= 2^{20} octets)

2.1 Donner le nombre d'entrées maximal d'une table des pages.

2.2 Décrire une entrée de la table des pages. Quelle est la taille maximale d'une table des pages.

Exercice 3

Soit un système paginé dans lequel le temps d'accès à un mot de la mémoire centrale est de 500 nanosecondes et le surcoût dû à un défaut de page est de 20 microsecondes.

3.1 Calculer le temps d'accès moyen à un mot en fonction du taux de défauts de page, en supposant que la table des pages est entièrement en mémoire centrale.

3.2 Quel est le taux minimum tel que le temps d'accès soit inférieur à 1,2 microsecondes.

Exercice 4

On considère une mémoire centrale M1 constituée de 2 cases et une mémoire secondaire M2. Soit P= abacabdacd la suite des références aux pages a,b,c,d situées initialement en M2.

4.1 Produire une trace sous forme d'une table qui illustre les défauts de page et le contenu de M1 pour la suite des références P en utilisant la stratégie FIFO.

4.2 On définit la fréquence de succès s comme suit: $s = 1 - (F/L)$ où F est le nombre de défauts de pages et L la longueur de la suite P. Calculer s dans le cas 1.

4.3 Répéter 1 en utilisant la stratégie LRU.

4.4 Quelles caractéristiques de P ont permis à la stratégie LRU d'être plus performante que la stratégie FIFO.

4.5 Répéter 1 et 2 en considérant une mémoire centrale de 3 cases.

4.6 Quel est l'effet de la taille de M1 sur s ?

La mémoire centrale comprend 25 cases, 5 processus s'exécutent simultanément. Chaque processus dispose de 5 cases et produit la suite de références (en numéros de pages):

processus $i + 1$: $0 + 5 * i, 1 + 5 * i, 2 + 5 * i, 3 + 5 * i, 4 + 5 * i, 0 + 5 * i, 1 + 5 * i, 2 + 5 * i, \dots$ pour $i = 0,1,2,3,4$.

La mémoire est initialement vide. Lorsqu'on choisit une page à remplacer, on choisit celle qui a été le moins récemment utilisée.

P1 ($i=0$) : 0,1,2,3,4,0,1,2,3,4,etc.

P2 ($i=1$): 5,6,7,8,9,5,6,7,8,9, etc

idem pour P3, P4, P5

Combien chaque processus provoque de défauts de pages?

Avec 5 cases, il n'y a pas de défaut de pages (si on passe à 4 cases, ce serait différent).