

# Révisions 2

## 1 Gestion de la mémoire<sup>1</sup>

### 1.1 Pagination

Considérons un espace d'adresses logiques de 8 pages contenant chacune 1024 octets. Ces adresses sont traduites pour une mémoire vive de 32 cadres de pages. Sur combien de bits se font l'adressage logique et l'adressage physique ?

### 1.2 Mémoire Virtuelle

On dispose d'une mémoire paginée à la demande. Il faut compter 8ms pour traiter un défaut de page lorsqu'il existe une page vide ou lorsque la page remplacée n'est pas modifiée. Le temps nécessaire pour remplacer une page modifiée se monte à 20 ms. Le temps d'accès normal à la mémoire est de 100 ns. En supposant que les pages remplacées soient modifiées dans 70% des cas, quel est le taux de défaut de page acceptable pour une dégradation des performances de 100% ?

### 1.3 Système de fichier, gestion des espaces libres

Dans un système de gestion des disques par allocation contiguë, on veut éviter la fragmentation en faisant du compactage à chaque fois qu'un fichier est supprimé. Avec la contiguïté, le temps de lecture (ou d'écriture) d'un fichier comprend le temps de positionnement des têtes, le temps de latence, puis un transfert à pleine vitesse.

1. En supposant un temps de positionnement de 5 ms, un temps de latence de 4 ms, un taux de transfert de 8 Mo/s et une taille moyenne de fichier de 8 Ko, combien de temps prendra le déplacement d'un fichier ?
2. Partant de là, combien de temps prendrait le compactage de la moitié d'un disque de 16 Go ?
3. Pensez vous que le compactage soit une stratégie réaliste ?

## 2 Pages mémoire et blocs disque (Annales 2010)

Rappelons qu'un disque lit ou écrit des données par blocs complets, un à la fois.

On suppose que la mémoire virtuelle (paginée à la demande) d'une machine utilise des pages (et des cadres de pages) de 4Ko, avec un disque de *swap* qui a des temps d'accès de 10 ms.

Dans le cours, on a supposé que la taille des pages mémoire correspondait à la taille des blocs du disque. On remet ici en cause cette hypothèse, et on s'intéresse aux conséquences en termes de performances du *swapping*.

- Que se passe-t-il si on utilise un disque de swap qui a
  1. des blocs de 2 Ko (une page mémoire utilisera 2 blocs)
  2. des blocs 8 Ko (en mettant deux blocs par page)
- Dans les deux cas, on peut retrouver des performances "optimales" en consentant un léger sacrifice de matériel. Proposez une solution, évaluez le coût financier (on suppose une mémoire virtuelle de 64 Go, 500 Euros pour un disque de 1024 Go octet de qualité pro).

---

1. D'après des sujets d'annales de l'EPITA (années 2006 et 2007)

### 3 Système de fichier<sup>2</sup>

On considère un système fonctionnant sous Unix disposant d'un système de fichiers doté des caractéristiques suivantes :

- la taille de bloc est de 1Ko,
- l'inode contient 10 pointeurs directs vers des blocs de données, et trois pointeurs de blocs d'indirection (un simple, un double et un triple),
- chaque adresse est codée sur 4 octets,
- le temps d'accès moyen au disque est de 40 ms.

La taille des pages est de 1Ko.

1. Un processus lit séquentiellement un fichier de 8 Mo, à raison de 256 octets à la fois (il fait donc 32768 demandes de lecture successives). On suppose qu'il n'y a qu'un seul processus dans le système, et que le système n'utilise pas de tampons de bloc disque, ce qui implique que chaque fois qu'une information située dans un bloc disque est nécessaire, ce bloc doit être lu depuis le disque. L'inode d'un fichier ouvert reste en mémoire centrale.
  - (a) A l'aide d'un schéma, donnez une idée de la façon dont le fichier est stocké sur le disque dur.
  - (b) Décrivez ce qui se passe lors des 41ème, 42ème, 45ème demandes de lecture de 256 octets.
  - (c) Évaluez le nombre et le temps d'accès disque nécessaire pour lire ce fichier.
2. On suppose maintenant que le système dispose d'un tableau de 100 tampons en mémoire centrale, dans lesquels il peut conserver 100 blocs de disque. Lorsque le système a besoin d'un bloc disque, pour lui ou pour le compte d'un processus, il recherche dans ces tampons si ce bloc est déjà en mémoire. Si ce n'est pas le cas, si aucun tampon n'est libre, il commence par en libérer un suivant un algorithme de remplacement de pages LRU (le moins récemment utilisé), puis lit le bloc dans un tampon libre et effectue le traitement sur ce bloc.
  - (a) Décrivez ce qui se passe lors de la 41ème, 42ème, 45ème demandes de lecture de 256 octets.
  - (b) Évaluez le nombre et le temps d'accès disque nécessaires pour lire ce fichier.

### 4 Allocation mémoire<sup>2</sup>

L'allocation de la mémoire principale peut être gérée de la même manière que celle du processeur. Soit une mémoire principale de 100 Ko, et des processus en ordre d'arrivée 1, 2,3 et 4. L'allocation du processeur se fait selon la méthode du tourniquet, avec un quantum suffisamment faible pour que l'on puisse supposer que les processus peuvent tous s'exécuter en même temps (ainsi, à  $t=0$ , on supposera que P1 et P2 s'exécutent en même temps)

Processus	Arrivée	Taille	Temps d'exécution
1	0	10	1
2	0	60	8
3	1	20	5
4	1	20	2

1. On considère qu'un processus occupe un ensemble contigu de la mémoire, d'un seul tenant.
  - (a) Avec cette méthode, comment calcule t-on l'adresse physique du processus par rapport à son adresse virtuelle?
  - (b) Montrer comment la mémoire est utilisée au cours du temps (en utilisant une représentation de la mémoire en liste chaînée avec une unité de bloc de 1 Ko).
  - (c) Calculer le temps moyen d'exécution (moyenne des durées d'exécution des processus) et tracer le taux d'occupation mémoire au cours du temps. En quoi cette méthode d'allocation n'est pas optimale?
2. Pour éviter le problème précédent, nous allons utiliser la méthode de pagination ou chaque processus sera découpé en blocs (pages) de taille fixe de 1Ko.

---

2. source originale non spécifiée

- (a) Avec cette méthode, comment calculer l'adresse physique du processus par rapport à son adresse virtuelle? Quelle table faudra-t-il stocker en mémoire?
- (b) Illustrer l'évolution de la mémoire avec l'exemple précédent.
- (c) Calculer le temps moyen d'exécution et tracer le taux d'occupation mémoire. Conclusion?

## 5 Ordonnancement de processus, quantum de temps<sup>3</sup>

On considère un système préemptif dans lequel l'ordonnancement des processus est fait par tourniquet. Indiquez quels sont les effets des choix suivants pour le quantum  $q$ , sachant que  $s$  est le temps de changement de contexte et  $t$  le temps moyen d'utilisation du processeur entre deux événements bloquants ( $t \gg s$  et  $[\epsilonpsilon] \ll s$ ) :

1.  $q = [\textit{infini}]$
2.  $q = [\epsilonpsilon]$
3.  $q = s$
4.  $q = t$
5.  $s < q < t$
6.  $q > t$

Pour chaque question, étudiez les cas où  $s$  est compris dans le quantum ou non.

## 6 Pagination<sup>3</sup>

### Exercice 1

On considère la table des pages suivante :

Page	Cadre
0	4
1	6
2	8
3	9
4	12
5	1

Sachant que les pages virtuelles et physiques font 1 Ko, quelle est l'adresse mémoire correspondant à chacune des adresses virtuelles suivantes codées en hexadécimal : 0x142A et 0x0AF1

### Exercice 2

Un programme a besoin de 5 pages virtuelles numérotées de 0 à 4. Au cours de son déroulement, il utilise les pages dans l'ordre suivant :

0 1 2 3 0 1 4 0 1 2 3 4

- S'il reste 3 pages libre en mémoire, indiquez la séquence des défauts de page, sachant que l'algorithme de remplacement est FIFO.
- Même question avec 4 pages disponibles en mémoire. Observation?

---

3. D'après les exercices de I. Demeure, B. Dupouy et L. Pautet (<http://www.infres.enst.fr/domas/BCI/TD/ExercicesBCI.htmlRTFToC9>)