TD N°2 : Système de gestion des fichiers (SGF)

Exercice n°1 :

1) Quelles sont les techniques d'allocation des blocs physique ? Avantages et inconvénients ?
2) Rappelez la structure d'un *inode* UNIX ?
3) Quelle technologie augmente la fiabilité des unités de stockage ?
4) Discuter les avantages et les inconvénients des caches (Buffer) ?

Exercice n°2 :

On veut stocker un fichier F avec 6000 articles de taille de 1KO par article.

1) Combien de page a-t-on besoin au minimum pour stocker tout le fichier si la taille de bloc est 32 KO.

Supposons que les valeurs de l'attribut clé de fichier sont des entiers avec une distribution uniforme.

2) Définissez une fonction d'hachage, expliquer votre choix.

Le FAT16 est utilisé par MS-DOS. En FAT16, les numéros de blocs sont écrits sur 16 bits. Si on suppose que la taille d'un bloc est 32Ko,

3) Calculer la taille maximale adressable ?

Exercice n°3 :

On considère un système de fichiers tel que l'information concernant les blocs de données de chaque fichier est donc accessible à partir du i-noeud de celui-ci (comme dans UNIX). On supposera que :

- Le système de fichiers utilise des blocs de données de taille fixe 1K (1024 octets) ;
- L'i-noeud de chaque fichier (ou répertoire) contient 12 pointeurs directs sur des blocs de données, 1 pointeur indirect simple, 1 pointeur indirect double et 1 pointeur indirect triple.
- Chaque pointeur (numéro de bloc) est représenté sur 4 octets.

a) Quelle est la plus grande taille de fichier que ce système de fichiers peut supporter ?
b) On considère un fichier contenant 100,000 octets. Combien de blocs de données sont-ils nécessaires (au total) pour représenter ce fichier sur disque ?

Exercice n°4 :

On considère un système disposant d'un système de fichiers similaire à celui d'UNIX avec une taille de blocs de données de 4K (4096 octets) et des pointeurs (numéros de blocs) définies sur 4 octets. On supposera que le i-noeud de chaque fichier compte 12 pointeurs directs, 1 pointeur indirect simple, 1 pointeur indirect double et 1 pointeur indirect triple. On désire créer un fichier contenant un total de 20.000.000 (vingt millions) de caractères (caractères de fin de ligne et de fin de fichier compris).

- Quelle est la fragmentation interne totale sur le disque résultant de la création de ce fichier.

Exercice n°5 :

Soit un fichier stocké dans le disque sous forme de 12 blocs nommés : 1, 2, ..., 12.
Les demandes d'E/S sont représentées comme suit :
D1= 1, 2, 5, 8, 3, 7, 9, 4, 10 ; **D2**=1, 2, 5, 8 ; **D3**=1, 2, 5, 8, 3, 7, 9, 4, 10 ; **D4**=2, 5, 3, 6, 11, 4, 12 ; **D5**=2, 5, 3, 6, 11. On suppose que le buffer peut contenir 6 blocs.

a) On utilise une politique de *remplacement* FIFO (First-In, First-Out), calculer le nombre des E/S de ces demandes dans l'ordre : **D1→D2→D3→D4→D5.**

Le tableau suivant représente le nombre de **références** et la **fréquence** de chaque bloc dans la charge de demande.

Blocs	1	2	3	4	5	6	7	8	9	10	11	12
Fréquence	3	5	5	3	5	2	2	3	2	2	2	1

b) Calculer le nombre des E/S de ces demandes dans l'ordre précédent, si on utilise l'algorithme LFU.

L'algorithme LFU (*Least Frequently Used*) se déroule comme suit :

- Remplacer le bloc qui a été référencé le moins souvent,
- pour chaque bloc de la chaîne de référence, nous devons conserver un nombre de références.
- Les références des blocs sont incrémentées à chaque fois qu'un bloc est référencé.