TD 9 – Systèmes de fichiers et disques

Exercice 1. Questions de cours
Les questions de cours sont à destinées à vous permettre de vérifier votre compréhension du cours. Elles sont à travailler à l'avance et ne seront pas traitées en TD ou TP.

1. Qu'est-ce qu'un code correcteur ? Expliquez le principe général.
2. Qu'est-ce qui définit l'emplacement d'un bloc sur un disque dur ?
3. Quel est l'avantage de l'algorithme SSTF ? Et son inconvénient ?
4. Quel est l'avantage des algorithmes de type look et scan ?

Exercice 2. Tailles de fichiers
On considère un système de fichiers classique où les informations concernant les blocs de données de chaque fichier est accessible à partir du FCB de celui-ci. On supposera que :

- le système de fichier utilise des blocs de données de taille fixe de 1ko (1024 octets) ;
- chaque pointeur (numéro de bloc) est représenté sur 4 octets ;
- le FCB de chaque fichiers (ou répertoire) occupe 1 bloc et contient 12 pointeurs directs sur des blocs de données, 1 pointeur indirect simple, 1 pointeur indirect double et 1 pointeur indirect triple.

1. Quelle est la plus grande taille de fichier que ce système de fichiers peut supporter ?
2. On considère un fichier de 100,000 octets. Combien de blocs de données occupera-t-il au total sur le disque ?

Exercice 3. Système de fichier
On considère un système de fichier utilisant différentes méthodes afin d'indexer les blocs physiques composant les fichiers. Le volume physique est organisé en blocs de 512 octets et les numéros de blocs sont représentés sur 32bits.

Le FCB d'un fichier contient 7 pointeurs directs et 1 pointeur vers un bloc d'index. Les blocs d'index sont chainés, dans chacun d'entre eux le premier pointeur indique quel est le prochain bloc d'index ou contient la valeur 0 indiquant la fin de la chaine.

Afin d'optimiser le stockage des très petits fichiers, ceux-ci peuvent-être stockés directement dans le bloc du FCB. En effet, pour un ficher n'occupant que quelques octets, il est possible de le stocker dans l'espace normalement utilisé pour les pointeurs vers des blocs physiques.

1. Quelle est la taille maximale d'un volume physique supportée par ce système de fichiers ?
2. Quel avantage y a-t-il à utiliser un chainage plutôt que des niveaux d'indirection supplémentaire.
3. Combien de blocs sur le volume physique sont occupés par un fichier de 30 octets ?
4. Combien de blocs sur le volume physique sont occupés par un fichier de 1Mo ?

Exercice 4. Ordonnancement d'accès aux blocs
On considère un disque contenant 256 cylindres (numérotés de 0 à 255) et les demandes d'accès suivantes aux cylindres, avec leur date d'arrivée (les requêtes arrivées à la même date sont donnés dans l'ordre d'arrivée : on suppose que le contrôleur ne met à jour sa liste de requêtes que tous les 100 unités de temps). On suppose que la tête est en position 24 au début.

- temps 0 : 54, 13, 22, 188, 245, 98, 24, 167
- temps 100 : 67, 93, 12, 25, 250, 220, 200
- temps 200 : 94, 230, 97, 30, 20

On suppose qu'il faut une unité de temps pour parcourir un cylindre et, le cas échéant, lire les données.

1. Appliquez l'algorithme FCFS et calculez le temps de traitement total.
2. Appliquez l'algorithme SSTF et calculez le temps de traitement total.

3. Appliquez l'algorithme C-SCAN (en supposant qu'on est ascendant au début) et calculez le temps de traitement total.

Exercice 5. Le système ext4fs

Nous allons considérer une version un peu simplifiée du système de fichier *ext4fs*, un système de fichier très répandu sous Linux. Dans ce système, le disque est découpé en blocs de 4ko.

L'allocation des blocs dans un système de fichier *ext4fs* se fait de manière indexée en utilisant ce que l'on appelle un *extent*. Un extent indique une suite de blocs du fichier qui sont placés de manière contiguë sur le disque. Les extents font 12 octets organisés de la manière suivante :

- les 4 premiers octets contiennent le numéro du premier bloc logique couvert par cet extent ;
- les 2 octets suivants contiennent la taille, en nomber de blocs, de l'extent ;
- les 6 derniers octets contiennent le numéro du bloc physique sur le disque où commence cet extent.

Exemple : Prenons un fichier de 15ko, et qui a donc 4 blocs de fichier (le dernier bloc ayant 1ko non utilisé) et supposons que ses trois premiers blocs sont placés dans les secteurs 1000, 1001 et 1002 du volume et que son quatrième bloc est sur le secteur 42 du volume. Il suffit de deux *extents* pour indiquer où se trouve la totalité du fichiers sur le volume :

(0, 3, 1000) : trois blocs de fichier à partir du bloc 0 sont stockés à partir du bloc de volume 1000.
(3, 1, 42) : l'unique bloc de fichier 3 est stocké sur le bloc de volume 42.

Liste d'extents : Pour indexer plusieurs parties non-contiguës d'un même fichier, on utilise une *liste d' extents*. Celle-ci est composée d'un *extent_header* sur 12 octets, qui indique notamment le nombre d'*extents* dans la liste, puis des *extents* eux même.

1. Quelle est la taille maximale d'un volume, en blocs et en octets ?

2. Quelle est la taille maximale d'un fichier, en blocs et en octets ?

3. Le *File Control Block* d'un fichier en ext4 contient une *liste d'extents* sur 60 octets (la liste contient donc au plus 4 *extents* en plus du *header*).

Quelle est la taille maximale (en blocs ou en octets) d'un fichier dont tous les *extents* sont stockés de cette manière, uniquement du FCB ?

Indexation indirecte : En pratique, on procède à une indexation à plusieurs niveaux : la liste d'extents du FCB peut pointer soit directement sur des blocs de données (comme nous l'avons vu à la question précédente), soit sur des blocs contenant à leur tour une liste d'*extents* :

- Si la liste d'*extents* contenue dans le FCB pointe directement vers des blocs de fichier, on parle d'accès direct (c'est le cas que nous avons vu à la question précédente) ;
- si elle pointe vers des listes d'*extents* qui, eux, pointent vers des blocs de fichiers, on parle d'accès indirect de niveau un ;
- Si elle pointe vers des listes d'*extents* qui pointent à leur tour vers des listes d'*extents* qui pointent vers des blocs de fichiers, on parle d'accès indirect de niveau 2 ;
- Et ainsi de suite...

Le niveau d'accès est indiqué dans le header de la liste d'extents sur l'inode. **Notez bien que lorsqu'un bloc contient une liste d'*extents*, ceux-ci occupent tout le bloc**.

4. Combien d'*extents* peut-on mettre dans une liste qui occupe un bloc complet ?

5. En vous aidant des réponses aux questions précédentes, quelle est la taille maximale (en blocs ou en octets) d'un fichier indexé en accès indirect de niveau un ?

6. De combien de niveaux d'indirection a-t'on besoin dans le pire des cas pour stocker les plus gros fichiers possibles (dont la taille a été calculée à la deuxième question) ?