Les systèmes de fichiers : implantation

Polytech Paris-Sud

Cycle ingénieur de la filière étudiant

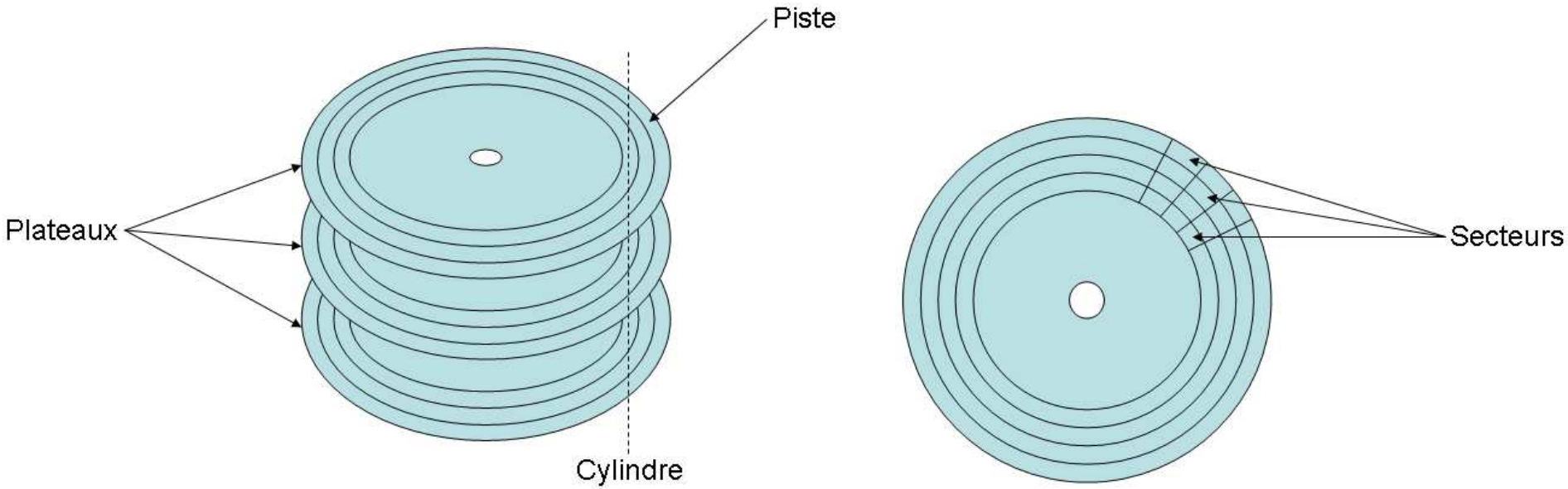
Louis Mandel

Université Paris-Sud 11

Louis.Mandel@lri.fr

année 2011/2012

Structure physique d'un disque



- Secteur : plus petite unité qui peut être lue par un disque (typiquement 512 octets)

Remarques : des figures de ce cours ont été prises sur Wikipedia

Organisation et codage des secteurs

- Adressage
  - CHS (*Cylinder, Head, Sector*)

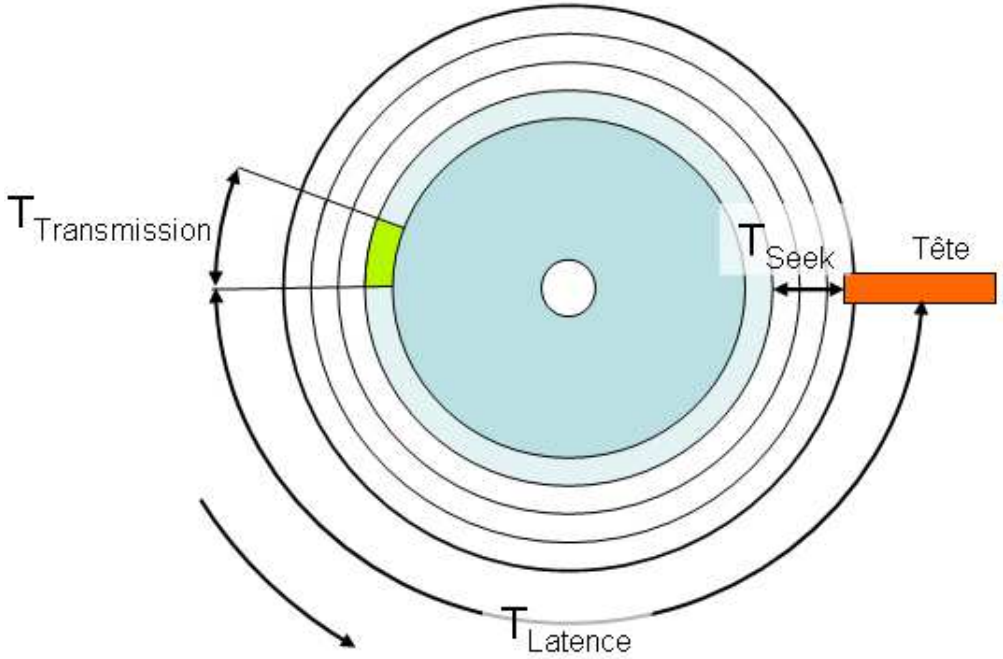
    $\{$num. cylindre, num. face, num secteur dans la piste$\}$
    - num. cylindre, num. face = num. piste
  - LBA (*Logical Block Address*)
    - Adresses logiques par numéros consécutifs
    - Le contrôleur de disque dispose d'une table de correspondance entre les adresses LBA et CHS

Formatage de bas niveau

- Exemple de structure d'un secteur

- Informations auxiliaires
  - Synchronisation
  - Identification du secteur
  - Codes de détection et correction d'erreurs (CRCC = Cyclic Redundancy Check Code)
  - Séparations des données par des « *gaps* »
- Le formatage de bas niveau :
  - partage le disque en secteurs
  - marquage des informations auxiliaires

Performances des disques



- Temps de recherche (*seek time*) : positionnement de la tête en face du bon cylindre
- Temps de latence : positionnement du secteur en face de la tête de lecture (dépend de la vitesse de rotation)
- Temps de transfert : transfert des données vers l'ordinateur

Stratégies d'ordonnancement de requêtes

- Changer l'ordre des requêtes pour :
  - Réduire les délais de recherche de cylindres
  - Réduire les délais de latence pour un même cylindre
  - Optimisation globale des délais de positionnement et rotation

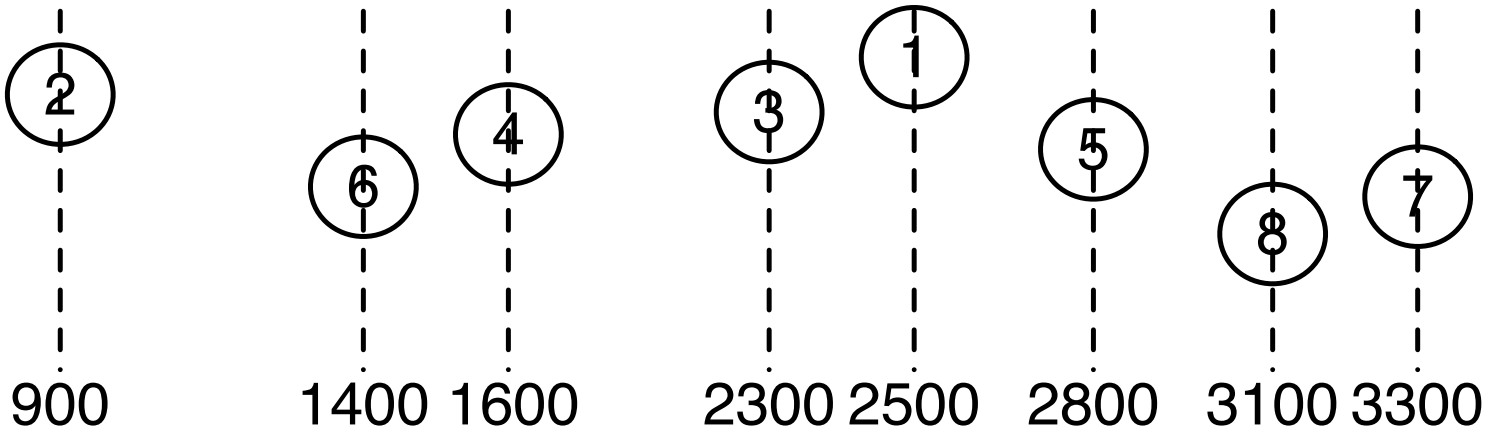
- Attention : Pour garantir que l'on accède toujours à la dernière version d'une donnée, les optimisations doivent conserver l'ordre des requêtes d'écriture et de lecture concernant un même secteur.

Réduction des positionnements de cylindres

- Les stratégies doivent tenir compte des nouvelles requêtes qui sont dynamiquement déposées pendant les échanges
- Stratégie Premier Arrivé Premier Servi (FCFS = First Come First Served)
  - Avantage : équité de service
  - Inconvénient : Nombreux déplacement d'un bout à l'autre du disque
- Stratégie Plus Proche Piste (SSF = Shortest Seek Time)
  - Inconvénient : problème d'équité. L'optimisation doit éviter de ne servir qu'un sous ensemble de cylindres proches
- Stratégie de Balayage (Scan)
  - Parcourir tous les cylindres dans un sens en servant toutes les requêtes puis dans l'autre sens
  - Variantes :
    - Ne servir les requêtes que dans un sens (Circular Scan)
    - Limiter de nombre de requêtes servies pour chaque cylindre

Réduction des positionnements de cylindres : Exemple

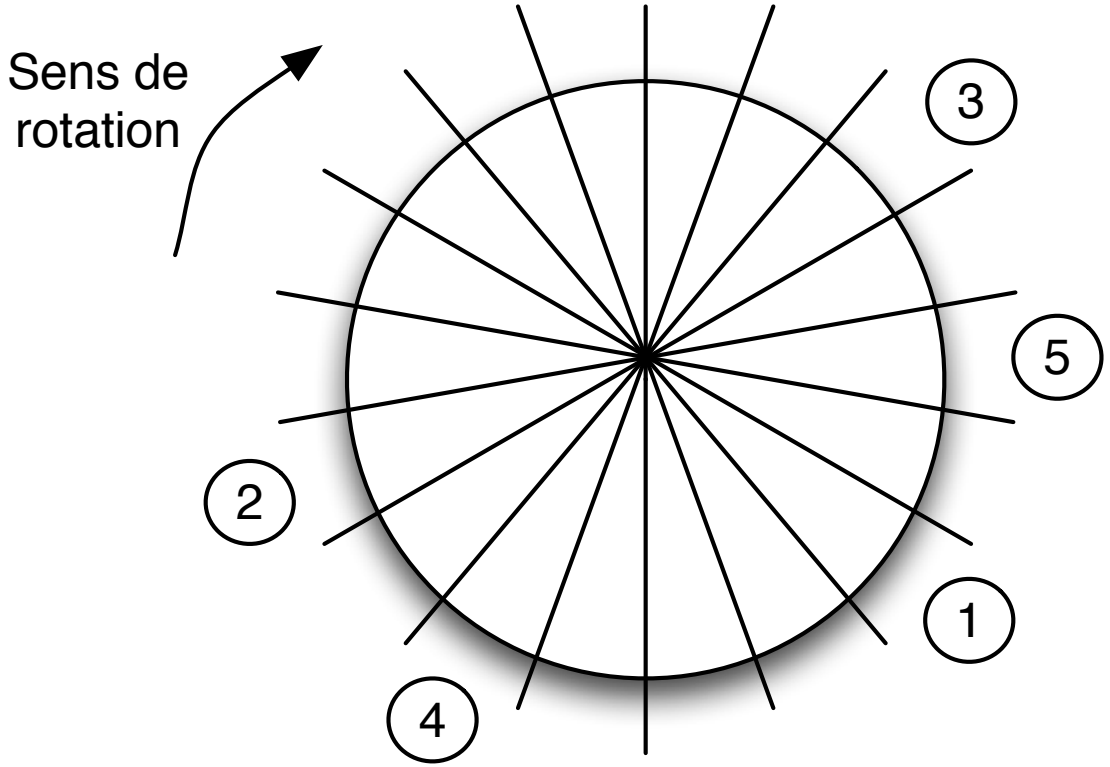
- Supposons que le disque doive traiter les demandes requêtes suivante :



- Représenter les stratégies FCFS, SSF, Scan et Circular Scan

Réduction des attentes de rotation

- Stratégie Premier Arrivé Premier Servi (FCFS)
- Stratégie Plus Proche Secteur (SATF = Shortest Acces Time First)



Optimisations globales

- Pour des cylindres proches, les délais de positionnement et de rotation sont comparables
- Optimisation de la somme des temps
- STF = Shortest Time First
  - ▷ On sert d'abord la requête qui minimise la somme des deux délais
  - ▷ Problème d'équité
- GSTF = Grouped STF
  - ▷ On divise le disque en groupes de cylindres
  - ▷ On sert circulairement les groupes de cylindres
  - ▷ On applique STF dans chaque groupe de cylindres
- GSTF(x) où x est le nombre de cylindres par groupe

Optimisation du temps de transfert

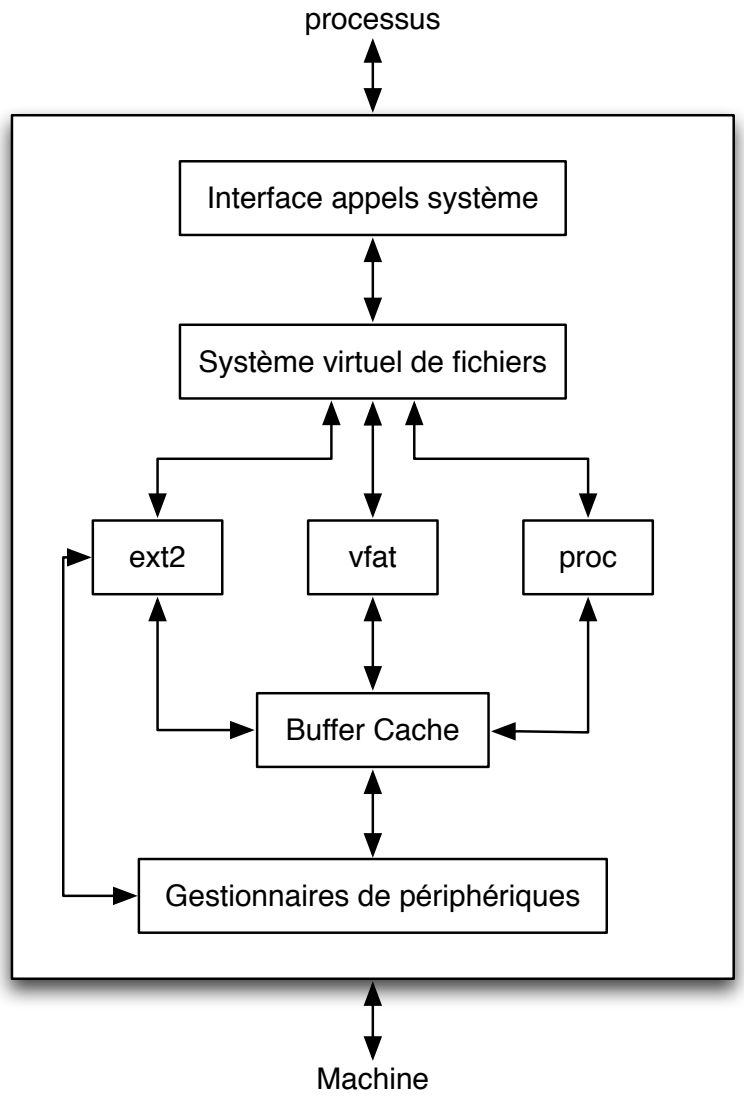
- Utilisation de caches
  - Mémoire cache du processeur d'entrées/sorties
  - Mémoire principale
    - Tampon système (*buffer cache*)

Allocation du disque

- Problèmes d'allocation du disque
  - Représentation de l'espace libre
    - limiter la taille des tables
  - Recherche d'un secteur libre ou d'un ensemble de secteurs libres
    - Fragmentation des fichiers : dispersion sur des cylindres éloignés
    - Fragmentation de l'espace libre : rend l'allocation difficile
- Les blocs = N secteurs successifs sur la même piste
  - Blocs petits :
    - décomposition peu efficace des accès
    - allocation trop fine
    - tables trop importantes
  - Blocs gros :
    - Fragmentation interne
- Un disque trop rempli a des performances dégradées

Implantation du système de fichiers de Linux

Le système virtuel de fichiers



Opération assurées par le VFS

- Les fonctionnalités assurées par le VFS sont indépendantes des systèmes de fichiers physiques.
- Le VFS implante
  - le cache des noms : association noms de fichiers / identifient (numéro de périphérique, numéro d'i-nœud)
  - le buffer cache : cache pour les entrées/sorties en mode bloc
  - les parties communes des appels systèmes
    - vérification des arguments
    - conversion des noms de fichiers
    - vérification des permissions
    - appel de la fonction du système de fichiers sous-jacent correspondant à l'appel système

Système de fichiers virtuel

- Le VFS utilise une approche « orientée objet »
  - chaque système de fichiers monté doit implanter des opérations de bases
- Types d'opérations que doivent fournir un système de fichiers
  - Opérations sur des systèmes de fichiers
    - lecture/écriture d'un i-nœud, modification du super-bloc, etc.
  - Opérations sur les i-nœuds en cours d'utilisation
    - création/suppression de liens, répertoires, etc.
  - Opérations sur des fichiers ouverts
    - primitives d'entrées/sorties sur les fichiers, etc.
  - Opérations sur les quotas

Buffer cache

- Liste des tampons mémoire en cours d'utilisation
- Lecture d'un bloc :
  - ▷ le contenu du bloc est placé dans le tampon
- Écriture d'un bloc :
  - ▷ le changement est effectué dans le tampon
  - ▷ le tampon est marqué comme modifié
- À intervalles réguliers, le processus `update` appelle la primitive `sync` pour forcer la réécriture de tous les tampons modifiés sur les disques

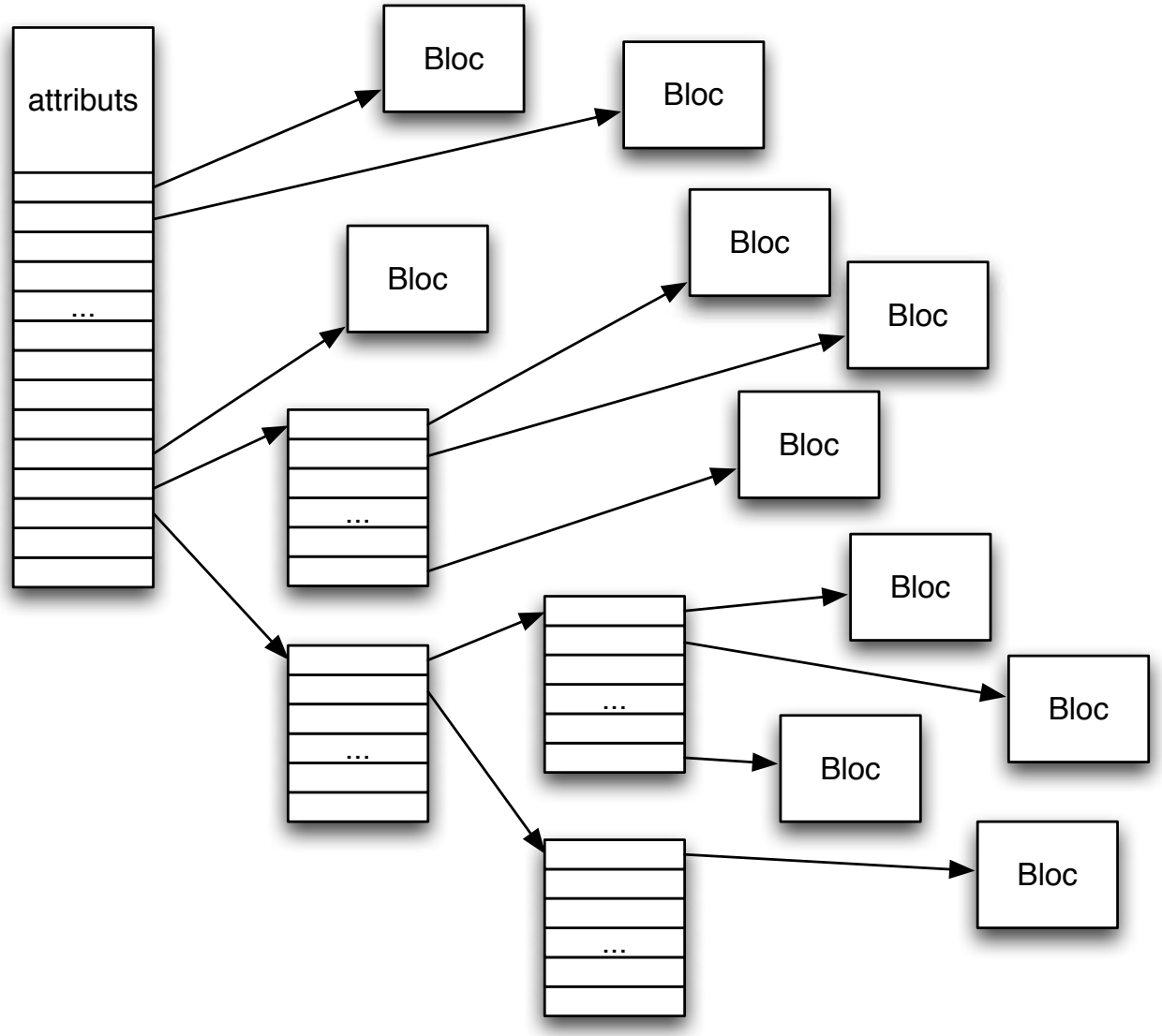
Implantation de systèmes de fichiers à i-nœuds

Rappels

- Structure logique du disque

- Le bloc d'initalisation (bootstap) contient un exécutable utilisé au démarrage
- Le superbloc contient en particulier
  - nom et taille du système de fichier
  - date et mode de montage
  - la table d'allocation des blocs
    - Utilisation d'un bit par bloc
  - la table d'allocation des i-nœuds
- Le nombre d'i-nœud fixé au moment du formatage (de haut niveau)

Rappels : Structure d'un i-nœud



Avantages et inconvénients

- Avantages :
  - Structure arborescente des i-nœuds permet un accès quasi-immédiat à chaque bloc du fichier
  - Structure des répertoires est simple : toutes les attributs des fichiers sont directement dans les i-nœuds
  - Possibilité de créer des liens physiques
  - Table d'allocation petite
    - Permet de faire des petits blocs et donc de limiter la fragmentation interne
- Inconvénients :
  - Beaucoup de méta-données
  - Pointeurs inutiles pour les petits fichiers (compensé par la faible fragmentation interne)

Exemple : EXT2

- Ext était le premier système de fichier à utilise le VFS
- Ext est un système de fichiers à base d'i-nœuds
- Ext2 est une extension de Ext
- Ext2 supporte tous les types de fichiers Unix
- Limitations :

	Minix FS	Ext	Ext2
Taille maximale	64 Mo	2 Go	2 To
Taille maximale d'un fichier	64 Mo	2 Go	2 Go
Taille maximale d'un nom de fichier	14 c	255 c	255 c

Ext2 : Structure physique

- Structure à deux niveaux
  - Les groupes de blocs

init	Groupe de blocs 1	Groupe de blocs 2	...	Groupe de blocs N

  - Un groupe de blocs

super bloc	descripteurs	bitmap blocs	bitmap i-nœuds	table des i-nœuds	blocs de données

Ext2 : Structure physique

- Chaque groupe de bloc contient
  - Une copie du superbloc
    - augment la fiabilité du système de fichier
    - réduit les temps d'accès au superbloc
  - Table des descripteurs
    - Contient les adresses des blocs de bitmap et de la table des i-nœuds
  - Bitmap pour les blocs
    - Table de bits
    - Indique pour chaque bloc du groupe s'il est libre ou occupé
  - Bitmap pour les i-nœuds
    - Indique pour chaque i-nœuds du groupe s'il est alloué ou disponible
  - Table des i-nœuds
    - une partie de la table des i-nœuds du système de fichiers
  - Blocs de données

Ext2 : Caractéristiques

- Caractéristiques de performances
  - Prélecture : lecture de plusieurs blocs contiguës à la requête de lecture
  - Allocation optimisées : allocation le plus possible dans le même groupe de blocs
  - Préallocation : allocation jusqu'à huit blocs adjacents lors de l'allocation d'un bloc
  - Lutte contre la fragmentation

Ext2 : Caractéristiques

- Autres caractéristiques
  - Choix de la taille du bloc
  - Liens symboliques rapides
  - Mémorisation de l'état
    - Propre : il n'y a eu que des lecture
    - Sale : il y a eu au moins une écriture
  - Mémorisation du nombre de fois que le système a été monté et date de la dernière vérification d'intégrité
  - Blocs réservés au super-utilisateur
  - Effacement sécurisé : le contenu des i-nœuds sont physiquement effacés
  - Fichiers auxquels on peut seulement ajouter des données

Persistance des Fichiers

Incohérences du système de fichiers

Détection et Réparation des incohérences

- But : savoir remettre le système de fichier dans un état cohérent même en cas de problème (exemple : après une coupure de courant)
- Exemple de problème : suppression d'un fichier se fait en au moins deux étapes :
  - ▷ supprimer son entrée dans le répertoire courant
  - ▷ marquer l'i-nœud du fichier comme libre

  Que se passe-t-il si la machine s'arrête entre ces deux opérations ?

Ext2 : File System Check (fsck)

- Comment retourner dans un système de fichiers « propre » après une erreur ?
- File System Check : Opération très couteuse !
- Vérification au niveau du fichier
  - Parcourir toute l'arborescence du système de fichier pour corriger
    - bitmap des i-nœuds
    - compte de liens dans les i-nœuds
- Vérification au niveau du bloc

fsck : Vérification au niveau du bloc

- Pour chaque bloc `b` on construit :
  `TF[b]` = nombre de référence de `b` dans les fichiers
- On appelle `TA` la table d'allocation des blocs
- le système est cohérent si :
  (`TF[b] = 0` et `TA[b] = 0`) ou (`TF[b] = 1` et `TA[b] = 1`)
  - ▷ si `TF[b] = 0` et `TA[b] = 1` alors `b` est orphelin
  - ▷ si `TF[b]` $\geq$ `1` et `TA[b] = 0` alors il faut modifier `TA`
  - ▷ si `TF[b]` $\geq$ `2` et `TA[b] = 1` alors il faut allouer et recopier le bloc pour avoir dans chaque fichier des blocs distincts mais de contenu identiques
- Remarque : l'information contenu dans les fichiers est considérée comme plus pertinente

Persistance des Fichiers

Les systèmes à journal

Les systèmes à journal

- Idée : marquer les changements à faire dans un journal avant les effectuer. Ainsi, les changements sont
  - soit réalisés normalement
  - soit réalisés au redémarrage
  - soit pas réalisés si l'arrête arrive avant l'écriture dans le journal
- Formes physiques possibles d'un journal :
  - Fichier standard
  - Zone réservée du disque
- Types de journaux :
  - Journal complet : contient les modifications faites aux données et aux méta-données
  - Méta-journal : contient seulement les modifications faites aux méta-données

Les systèmes à journal : Exemple

- Augmenter la taille d'un fichier peut se faire en trois étapes
  - changer l'information de taille dans l'i-nœud
  - allouer l'espace nécessaire dans la bitmap des blocs
  - ajouter les données à la fin du fichier
- Supposons un arrêt entre les étapes 2 et 3
  - Un journal complet sera détecter et corriger le problème
  - Un méta-journal laissera un fichier avec un contenu incorrect

Ext3 ∼ Ext2 + journal

- Ext3 propose trois niveaux de journal :
  - Journal complet
  - Méta-journal
  - Journal ordonné
    - identique au méta-journal
    - mais les données sont écrites avant les méta-données
    - On peut perdre des données mais un fichier ne peut pas contenir des informations fausses
- Avantages
  - Compatibilité avec Ext2
  - Simplicité
- Inconvénients
  - Basé sur une architecture ancienne
  - Exemple : le nombre d'i-nœuds est fixé au moment du formatage

NTFS

- Système inspiré des systèmes à base d'i-nœuds
- Système de fichiers journalisé
- Structure

  - Master Boot Record
  - Master File Table $\sim$ table des i-nœuds
- Structure d'une entrée de la MFT

- Traitement spécial des petits fichiers

Erreurs physiques

Erreurs physiques

- Fautes possibles
  - mauvais état du disque $\Rightarrow$ secteurs endommagés
  - faute de disque
  - destruction accidentelle de fichiers par des utilisateurs

- Gestion des blocs endommagés
  - Détection lors du formatage
    - Marquage des blocs défectueux
  - Détection en écriture
    - réécriture dans un nouveau bloc (perte de performance)
  - Détection en lecture
    - récupération éventuelle s'il existe une copie de sauvegarde assez récente

Sauvegardes et Redondance

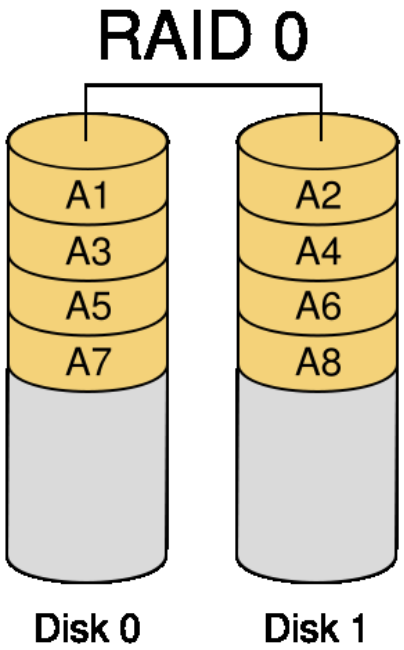
- Copies de sauvegarde (*Backup*)
  - Tous les fichiers sont copiés régulièrement sur un autre support

- Copies incrémentales
  - Le système gère des dates de sauvegarde
  - Sauvegarde des modifications

- Tableau de disques à contrôle de parité
  - RAID = Redundant Array of Inexpensive Disks
  - RAID = Redundant Array of Independent Disks

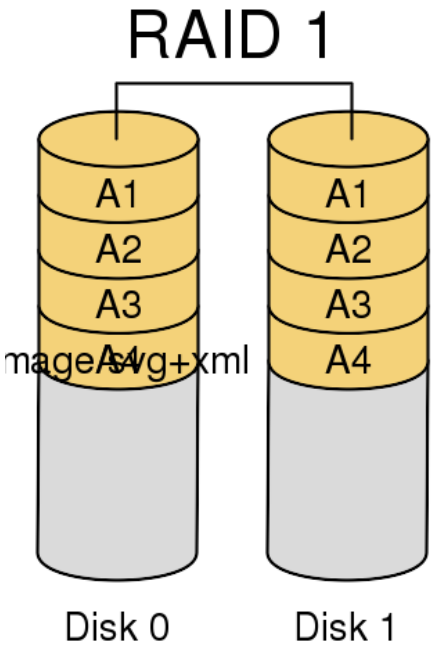
RAID

- RAID 0 : Entrelacement (stripping)
  - Les blocs d'un fichier sont répartis sur plusieurs disques
  - Améliore les performances
  - N'augmente pas la fiabilité



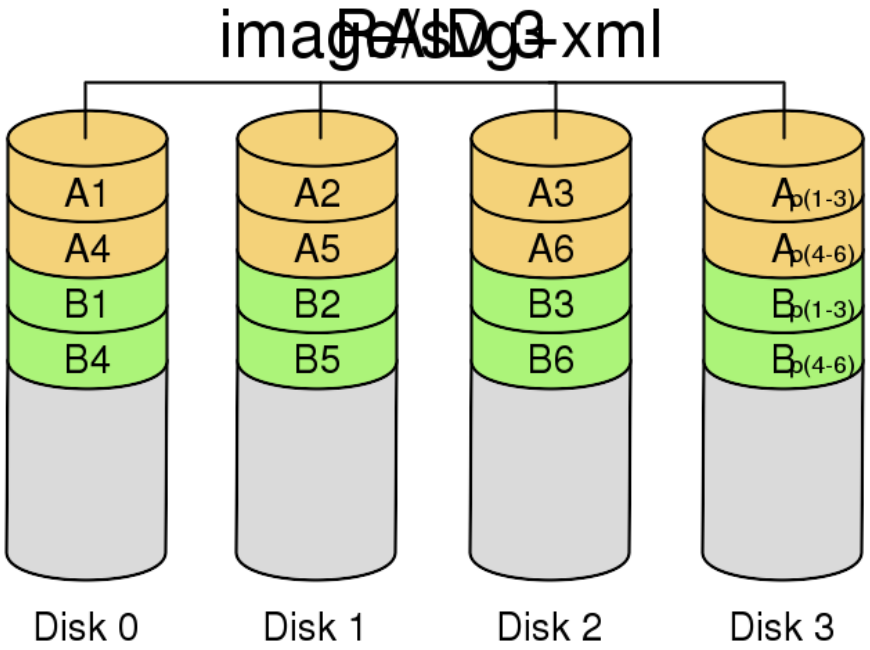
RAID

- RAID 1 : miroir
  - Augmente la fiabilité
  - 2 fois plus de disques



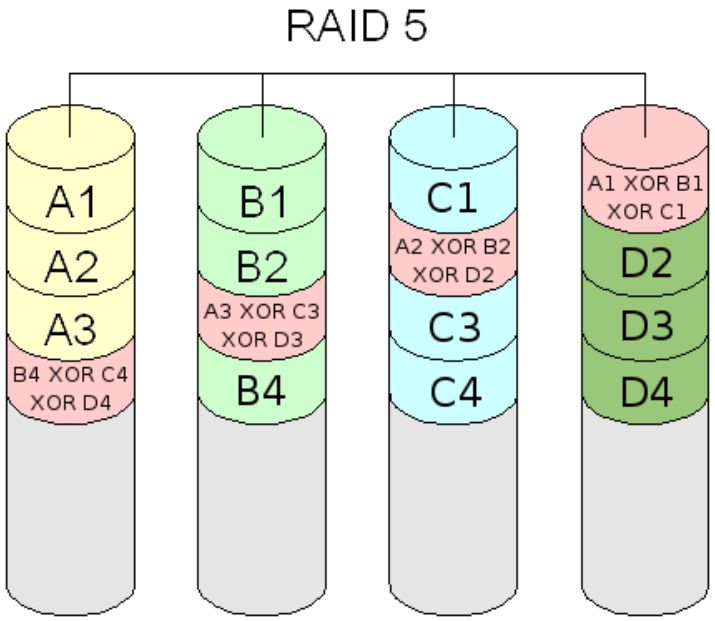
RAID

- RAID 3 : RAID 0 + un disque de code correcteur d'erreur
  - Augmente la fiabilité à un coût inférieur (tolère une panne)
  - Surcharge du disque de parité
- RAID 4 : RAID 3 par blocs



RAID

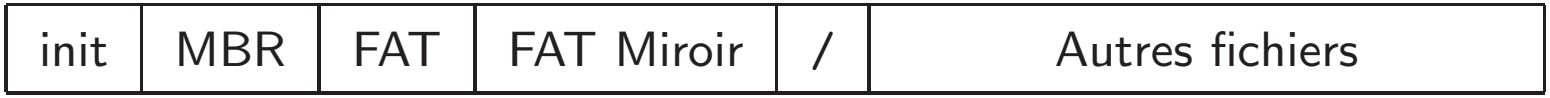
- RAID 5 : RAID 4 mais avec code correcteur réparti sur tous les disques
  - améliore les performances
  - meilleure disponibilité en cas de panne
- RAID 6 : RAID 5 avec plus de redondance (tolère 2 pannes)



Autres types de systèmes de fichiers

Le système FAT (File Allocation Table) : MS-DOS

- Système de fichier qui n'est pas à base d'i-nœud
- Structure logique du disque

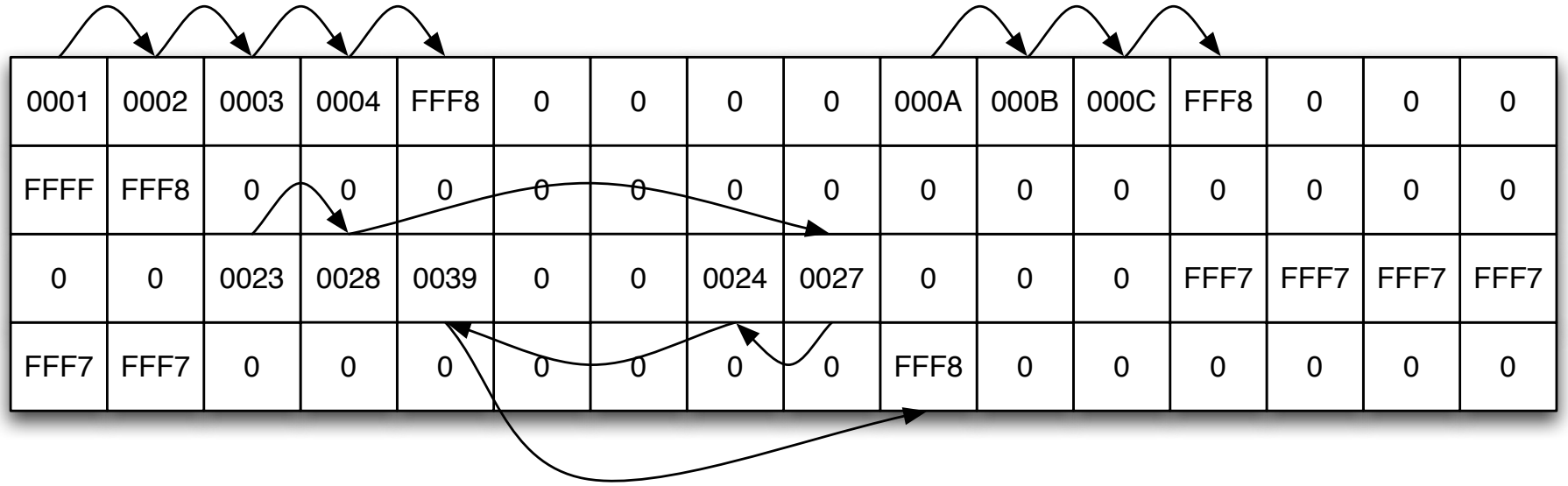


  - MBR (*Master Boot Record*)
    - nombre d'octets par secteurs, nombre de secteurs par cluster, nombre de secteurs sur le disque, etc.
  - Table d'allocation des clusters (FAT)
    - chaque case représente un cluster (clusters = bloc mais généralement plus gros : jusqu'à 128 Ko)
  - Le répertoire racine
  - Les clusters de données

Table d'allocation des clusters

- Chaque case peut contenir une valeur codée sur 16 ou 32 bits
- La signification de ces valeurs est :
  - `0` : cluster vide
  - `1` : cluster réservé au système
  - `0xFFF7` : cluster défectueux
  - Une autre valeur `n` :
    - le cluster est occupé par un fichier
    - la suite du fichier est contenu dans le cluster `n`
    - si `n` est compris entre `0xFFF8` et `0xFFFF`, c'est le dernier cluster du fichier
- Un fichier est une liste chaînée de clusters

Table d'allocation des clusters : exemple



Structure d'un répertoire

- Un répertoire est une liste d'entrées contenant :
  - Le nom du fichier (8 + 3 caractères)
  - Les attributs (lecture seule, fichier caché, sous-répertoire, etc.)
  - Les dates (création, accès, modification)
  - Numéro du premier cluster
  - Taille du fichier

- Contrairement aux systèmes de fichiers avec i-nœuds
  - Les meta-données sur les fichiers sont stockées au niveau des répertoires
  - Il ne peut pas y avoir de liens physiques

FAT : Avantages et inconvénients

- Avantages :
  - Simple et léger
  - Bien adapté à des petits disques ($\leq$ 1 Go)
- Inconvénients :
  - Limitations fonctionnelles : nom de fichier, propriétaires, etc.
  - Mauvaise adaptations aux opérations courantes sur les fichiers : accès aléatoire, écriture en fin de fichier (problème accentué sur les gros fichiers)
  - Limitation sur la taille des disques (table d'allocation)
    - Remarque : dans les systèmes à i-nœuds il suffit d'un bit par bloc
  - Fragmentation interne (dernier cluster sous-utilisé)

procfs

- Informations gérées par le noyau
- Fichiers virtuels (pas des données stockées sur disque mais en mémoire centrale)
- Exemples de fichiers de `/proc`
  - `cpuinfo` : description du ou des processeurs
  - `loadavg` : charge du système
  - `meminfo` : état d'occupation de la mémoire
  - `uptime` : temps écoulé depuis le démarrage du système
  - etc.
- En plus de ces fichiers, `/proc` contient des répertoires comme :
  - `net` : informations sur le réseau
  - `scsi` : informations sur les gestionnaires de périphériques SCSI
  - un répertoire par processus existant dans le système
    - `cmdline`, `cwd`, `environ`, `fd`, etc.