Systèmes UNIX. Fichiers

Sergey Kirgizov

Système de fichiers

Codage d'information

Arborescence de fichiers

Système de fichiers

Codage d'information

Fichier et codage

De l'information vers le fichier

- Toute information (image, texte, vidéo) peut se représenter sous une forme binaire
- Information binaire : 0 1 1 0 0 0 0 0 1 1 0 1 0 1 . . .
- Cette transformation est le résultat de l'application de règles de conversions strictes et connues, le format de données
- Pour une information données (exemple image), plusieurs formats de données possibles (gif, jpeg, png, bmp,. . .)

Exemple d'encodage d'une image

Codage d'une image noir et blanc vers suite d'octets

- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

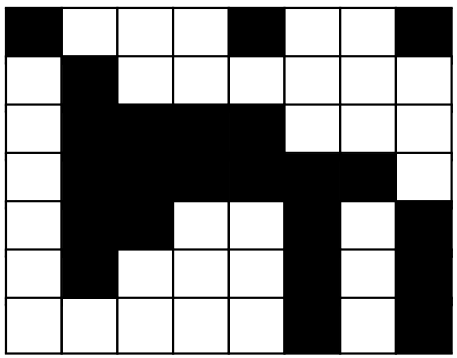


image 8×7 pixels

$\Rightarrow$ 0 0 0 0 1 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0 1 0
0 1 0 1 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 0 1 1 0 0 1 0
1 0 1 0 0 0 1 0 1 0 0
0 0 0 1 0 1

Exemple d'encodage d'une image

Codage d'une image noir et blanc vers suite d'octets

- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

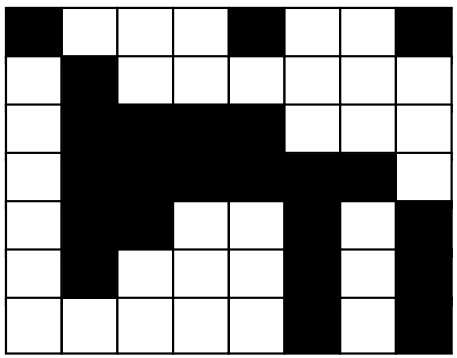


image 8×7 pixels

$\Rightarrow$ 0 0 0 0 1 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0 1 0
0 1 0 1 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 0 1 1 0 0 1 0
1 0 1 0 0 0 1 0 1 0 0
0 0 0 1 0 1

Exemple d'encodage d'une image

Codage d'une image noir et blanc vers suite d'octets

- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

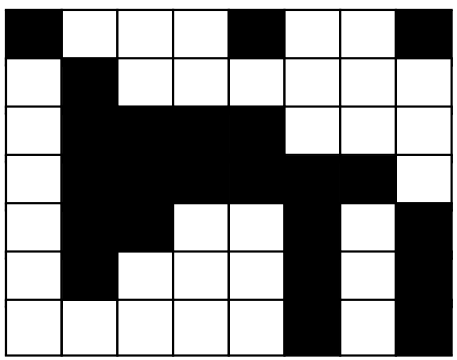


image 8×7 pixels

$\Rightarrow$ 0 0 0 0 1 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0 1 0
0 1 0 1 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 0 1 1 0 0 1 0
1 0 1 0 0 0 1 0 1 0 0
0 0 0 1 0 1

Exemple d'encodage d'une image

Codage d'une image noir et blanc vers suite d'octets

- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

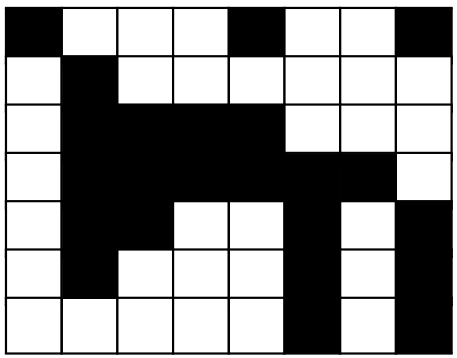


image 8×7 pixels

$\Rightarrow$

0 0 0 0 1 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0 1 0
0 1 0 1 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 0 1 1 0 0 1 0
1 0 1 0 0 0 1 0 1 0 0
0 0 0 1 0 1

Exemple d'encodage d'une image

Codage d'une image noir et blanc vers suite d'octets

- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

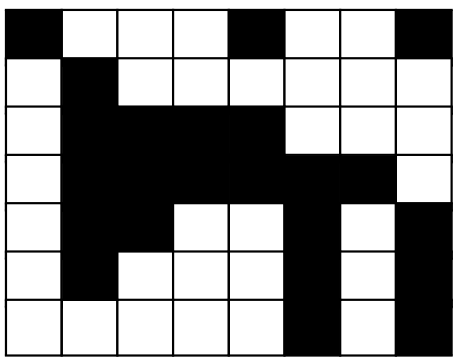


image 8×7 pixels

$\Rightarrow$ 0 0 0 0 1 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0 1 0
0 1 0 1 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 0 1 1 0 0 1 0
1 0 1 0 0 0 1 0 1 0 0
0 0 0 1 0 1

Exemple d'encodage d'une image

Codage d'une image noir et blanc vers suite d'octets

- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

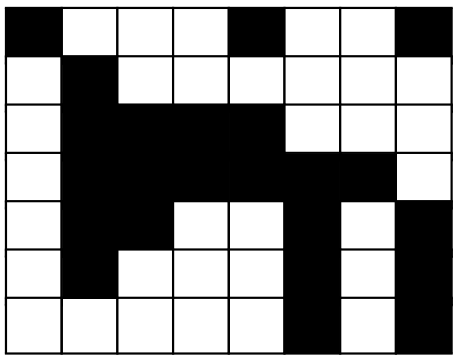


image 8×7 pixels

$\Rightarrow$ 0 0 0 0 1 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0 1 0
0 1 0 1 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 0 1 1 0 0 1 0
1 0 1 0 0 0 1 0 1 0 0
0 0 0 1 0 1

Exemple d'encodage d'une image

Codage d'une image noir et blanc vers suite d'octets

- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

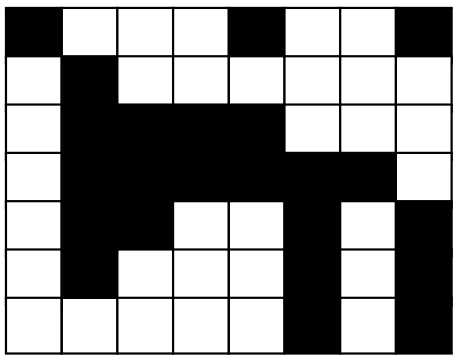


image 8×7 pixels

$\Rightarrow$ 0 0 0 0 1 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0 1 0
0 1 0 1 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 0 1 1 0 0 1 0
1 0 1 0 0 0 1 0 1 0 0
0 0 0 1 0 1

Exemple d'encodage d'une image

Codage d'une image noir et blanc vers suite d'octets

- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

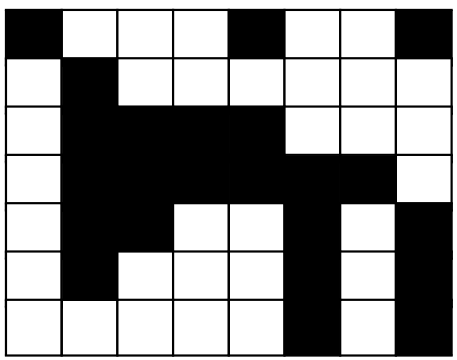


image 8×7 pixels

$$\Rightarrow \begin{array}{cccccccccccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & & & & & \end{array}$$

Exemple d'encodage d'une image

Question

- En utilisant le même format de données, quelle image obtient-on avec la séquence suivante ?

```
0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0
0 0 1 1 1 1 0 0 0 1 0 0 0 0 1 0
1 0 1 0 1 0 0 1 1 0 1 0 1 0 0 1
1 0 0 0 0 1 0 1 1 0 1 1 1 0 0 1
0 1 0 0 0 0 1 0 0 0 1 1 1 1 0 0
```

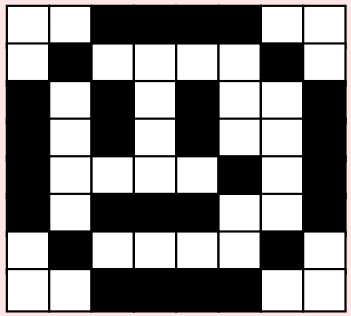
- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

Exemple d'encodage d'une image

Question

- En utilisant le même format de données, quelle image obtient-on avec la séquence suivante ?

```
0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0
0 0 1 1 1 1 0 0 0 1 0 0 0 0 1 0
1 0 1 0 1 0 0 1 1 0 1 0 1 0 0 1
1 0 0 0 0 1 0 1 1 0 1 1 1 0 0 1
0 1 0 0 0 0 1 0 0 0 1 1 1 1 0 0
```



- 8 premiers bits : largeur de l'image en pixels
- 8 bits suivants : hauteur de l'image en pixels
- lecture séquentielle des lignes de l'image, en commençant en haut. Pour chaque pixel noir, ajout d'un bit à 1, pour chaque pixel blanc, ajout d'un bit à 0. On recommence jusqu'à la dernière ligne.

Exemple d'encodage d'un texte

Codage d'un texte comportant des lettres et ponctuation

- 26 caractères + les caractères _ , .
- Chaque caractère est associé à son indice dans l'alphabet : A = 1, B=2, C=3, . . ., Z=26
- le caractère _ = 0, le caractère , = 27 et . = 28
- Total : 29 caractères de 0 à 28. Chaque caractère être codé de manière unique avec 5 bits (car $2^5 = 32 > 29$)

I	N	F	O	T	R	O	Message
⇓							
9	14	6	15	20	18	15	Décimal
⇓							
0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	binaire

Exemple d'encodage d'un texte

Codage d'un texte comportant des lettres et ponctuation

- 26 caractères + les caractères _ , .
- Chaque caractère est associé à son indice dans l'alphabet : A = 1, B=2, C=3, . . ., Z=26
- le caractère _ = 0, le caractère , = 27 et . = 28
- Total : 29 caractères de 0 à 28. Chaque caractère être codé de manière unique avec 5 bits (car $2^5 = 32 > 29$)

I	N	F	O	T	R	O	Message
⇓							
9	14	6	15	20	18	15	Décimal
⇓							
0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	binaire

Exemple d'encodage d'un texte

Codage d'un texte comportant des lettres et ponctuation

- 26 caractères + les caractères _ , .
- Chaque caractère est associé à son indice dans l'alphabet : A = 1, B=2, C=3, . . ., Z=26
- le caractère _ = 0, le caractère , = 27 et . = 28
- Total : 29 caractères de 0 à 28. Chaque caractère être codé de manière unique avec 5 bits (car $2^5 = 32 > 29$)

I	N	F	O	T	R	O	Message
⇓							
9	14	6	15	20	18	15	Décimal
⇓							
0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	binaire

Exemple d'encodage d'un texte

Codage d'un texte comportant des lettres et ponctuation

- 26 caractères + les caractères _ , .
- Chaque caractère est associé à son indice dans l'alphabet : A = 1, B=2, C=3, . . ., Z=26
- le caractère _ = 0, le caractère , = 27 et . = 28
- Total : 29 caractères de 0 à 28. Chaque caractère être codé de manière unique avec 5 bits (car $2^5 = 32 > 29$)

I	N	F	O	T	R	O	Message
⇓							
9	14	6	15	20	18	15	Décimal
⇓							
0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	binaire

Exemple de décodage d'un texte

Question

- En utilisant le même format de données, quel texte obtient-on avec la séquence suivante ?

```
0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 1
0 0 0 1 0 0 0 0 1 0 1 0 1 0 1 0 0 1 1 0 1 0
1 0 0 1 1 0 0 0 0 1 0 1 1 0 1 1 1 0 0 1 0 1
0 0 0 0 1 0 0 0 1 1 1 1 0 0
```

- 26 caractères + les caractères _ , et .
- Chaque caractère est associé à son indice dans l'alphabet : A = 1, B=2, C=3, ..., Z=26
- le caractère _ = 0, le caractère , = 27 et . = 28
- Total : 29 caractères. Chaque caractère codé avec 5 bits

Exemple de décodage d'un texte

Question

- En utilisant le même format de données, quel texte obtient-on avec la séquence suivante ?

```
0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 1
0 0 0 1 0 0 0 0 1 0 1 0 1 0 1 0 0 1 1 0 1 0
1 0 0 1 1 0 0 0 0 1 0 1 1 0 1 1 1 0 0 1 0 1
0 0 0 0 1 0 0 0 1 1 1 1 0 0
```

1	0	4	3	24	16	21	9	21	6	2	27	18	16	17	28
A	_	D	C	X	P	U	I	U	F	B	,	R	P	Q	.

- 26 caractères + les caractères _ , et .
- Chaque caractère est associé à son indice dans l'alphabet : A = 1, B=2, C=3, ..., Z=26
- le caractère _ = 0, le caractère , = 27 et . = 28
- Total : 29 caractères. Chaque caractère codé avec 5 bits

Fichier et information

- Un fichier est une suite d'octets, qui décrit une information
- Le codage cette information en octets, et son décodage (interprétation) serait grace au format de fichier
- L'extension d'un fichier (ex .jpg, .doc, .mp3) est simplement une indication sur le format supposé d'un fichier
- Mais cette indication peut-être fausse : si on change l'extension
- Changer l'extension d'un fichier ne change pas son codage

Philosophie

Unix selon l'ordre des raisons : la philosophie de la pratique informatique

Il est parfois fécond, en philosophie des sciences, de chercher si les concepts techniques relèvent d'une nécessité de structure plutôt que des seuls hasards de l'invention.

...

En essayant de fonder de la sorte les concepts fondamentaux des systèmes d'exploitation que sont les notions de processus et de fichier, on s'aperçoit qu'ils sont, depuis Unix, les pendants des notions ontologiques abstraites d'acte et d'objet...

...

la pratique informatique est pour la philosophie un objet aussi pur que les mathématiques.

Baptiste Mélès

`https://journals.openedition.org/philosophiascientiae/897`

`http://baptiste.meles.free.fr/`

Les fichiers sous UNIX

Concept de fichiers

Sous UNIX, tout est vu comme des fichiers :

- Fichiers *réguliers* : documents vidéos, images, texte
- Répertoires
- Périphériques
- Mécanismes de communication entre processus

Le concept de système de fichier

Un système de fichier (*FS* pour *Filesystem*) a pour objectif :

- d'organiser les fichiers sur un support (disque, clé USB, ...)
- de définir comment stocker et accéder à leur contenu
- de stocker des propriétés et informations diverses sur ces fichiers pour permettre leur traitement (accès, sécurité, ...)

Les fichiers sous UNIX

Concept de fichiers

Sous UNIX, tout est vu comme des fichiers :

- Fichiers *réguliers* : documents vidéos, images, texte
- Répertoires
- Périphériques
- Mécanismes de communication entre processus

Le concept de système de fichier

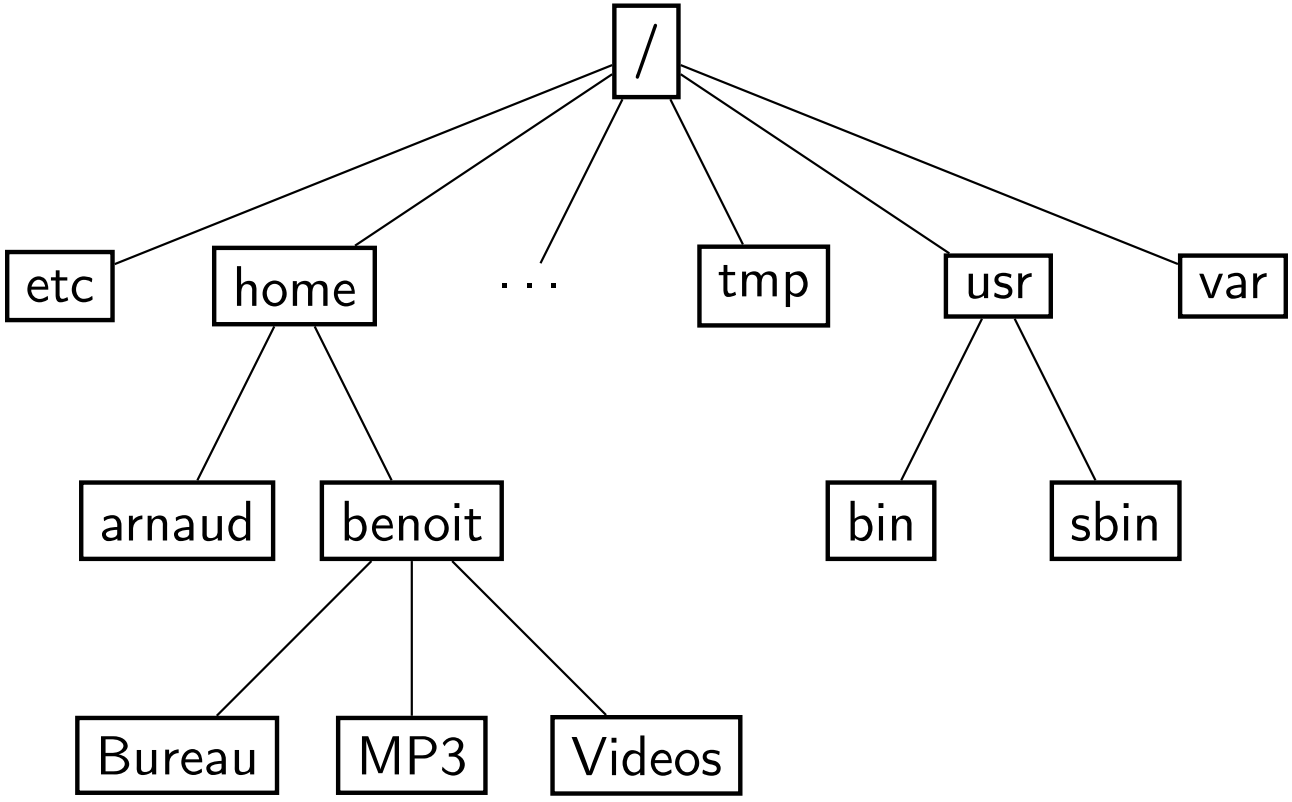
Un système de fichier (*FS* pour *Filesystem*) a pour objectif :

- d'organiser les fichiers sur un support (disque, clé USB, ...)
- de définir comment stocker et accéder à leur contenu
- de stocker des propriétés et informations diverses sur ces fichiers pour permettre leur traitement (accès, sécurité, ...)

Arbres

Organisation en arborescence

Les fichiers sont organisés en arborescence, dont la racine est /



Organisation en arborescence

Structure arborescente

- La racine est /
- Chaque répertoire peut contenir des fichiers ou des sous-répertoires
- Une seul arborescence générale

Identification des répertoires

- Chaque répertoire possède un chemin unique depuis la racine, qui définit son nom. Par exemple :
  - /home/benoit/MP3
  - /tmp
  - /usr/sbin

Question

Vous voulez savoir si vous êtes le descendant de Napoleon, et si oui trouver la lignée qui vous connecte. De quelle information (minimale) devez-vous disposer sur n'importe quel individu ? (deux méthodes possibles)

Réponse :

Question

Vous voulez savoir si vous êtes le descendant de Napoleon, et si oui trouver la lignée qui vous connecte. De quelle information (minimale) devez-vous disposer sur n'importe quel individu ? (deux méthodes possibles)

Réponse :

Question

Vous voulez savoir si vous êtes le descendant de Napoleon, et si oui trouver la lignée qui vous connecte. De quelle information (minimale) devez-vous disposer sur n'importe quel individu ? (deux méthodes possibles)

Réponse :

1. Pour chaque individu, connaitre la liste de ses enfants

Question

Vous voulez savoir si vous êtes le descendant de Napoleon, et si oui trouver la lignée qui vous connecte. De quelle information (minimale) devez-vous disposer sur n'importe quel individu ? (deux méthodes possibles)

Réponse :

1. Pour chaque individu, connaitre la liste de ses enfants
2. Pour chaque individu, connaître ses parents

Organisation en arborescence

Remarque (A noter pour la suite)

- En plus de contenir des fichiers et des sous-répertoires, un répertoire contient également son répertoire parent (! ?)
- Par ailleurs, un répertoire se contient lui-même (? !)
- Pour un répertoire, son répertoire parent est identifié par '`..`'
- Un répertoire peut s'identifier lui-même par '`.`'
- Ces liens ne sont pour l'instant pas présentés
- Les fichiers commençant par un '`.`' sont des fichiers cachés

Répertoire de départ

Répertoire personnel (ou répertoire maison)

- Chaque utilisateur possède un répertoire personnel
- C'est son répertoire de départ lorsqu'il lance un shell
- Le caractère ~ désigne ce répertoire pour l'utilisateur en cours
- Il est aussi contenu dans la variable d'environnement `$HOME`
- Enfin, la séquence ~*user1* désigne le répertoire personnel de l'utilisateur `user1`
- Pour visualiser le contenu de ces variables : `echo`

```
1  benoit$ echo ~
2  /home/benoit
3
4  benoit$ echo $HOME
5  /home/benoit
6
7  benoit$ echo ~arnaud
8  /home/arnaud
```

Navigation dans l'arborescence

Notion de répertoire de travail (ou répertoire courant)

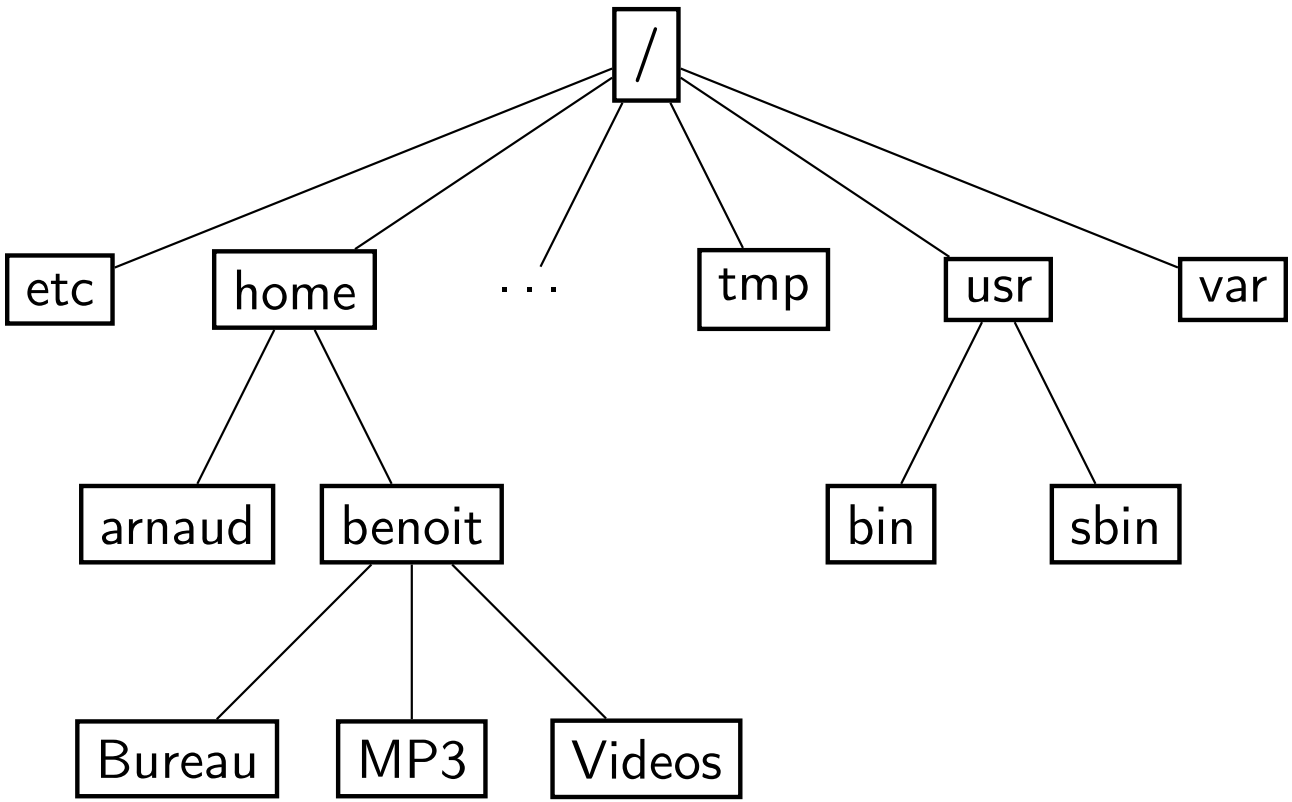
- Répertoire de travail : répertoire dans lequel l'utilisateur se situe à un instant donné
- Un utilisateur est toujours dans un répertoire de travail
- Affichage du répertoire de travail : commande `pwd`, ou variable d'environnement `$PWD` (visualisé avec commande `echo`)

```
1  benoit$ echo $PWD
2  /home/benoit
3
4  benoit$ pwd
5  /home/benoit
```

Navigation dans l'arborescence

Changer de répertoire (de travail) : commande `cd`

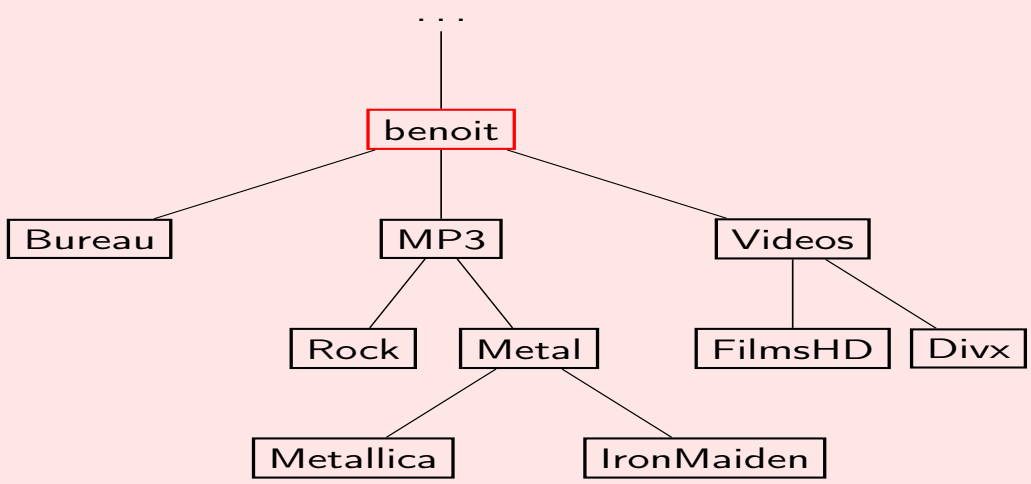
- commande `cd` (*change directory*) + *`chemin`* du nouveau répertoire : permet de changer de répertoire
  - chemin absolu, cad défini depuis la racine `/`
  - ou relatif, cad défini à partir du répertoire de travail et de caractères joker, tel que `~`, ou encore :
    - '`..`' : désigne le répertoire parent d'un répertoire
    - '`.`' : désigne le répertoire courant (lui-même)
  - Si pas de chemin précisé : retour au répertoire personnel
  - `cd -` : retour au répertoire de travail précédent


/
etc
home
...
tmp
usr
var
arnaud
benoit
bin
sbin
Bureau
MP3
Videos

```
1  benoit$ pwd
2  /home/benoit
3
4  benoit$ cd /
5  benoit$ pwd
6  /
7
8  benoit$ cd /usr/bin
9  benoit$ pwd
10 /usr/bin
11
12 benoit$ cd ~arnaud
13 benoit$ pwd
14 /home/arnaud
15
16 benoit$ cd ../benoit/MP3
17 benoit$ pwd
18 /home/benoit/MP3
19
20 benoit$ cd -
21 /home/arnaud
```

Question

On vous propose l'arborescence suivante :



Quel sera le répertoire de travail, si l'on tape les commandes suivantes à partir du répertoire personnel de benoit (en rouge) ?

1. `cd ./MP3/Metal/Metallica`
2. `cd ./Videos/./../Divx/./.`
3. `cd Videos/../MP3/./Metal/./../Metallica/./../.`
4. `cd ./Bureau/./../.Bureau/./../Rock`

Navigation dans l'arborescence

Lister le contenu d'un répertoire : commande `ls`

- commande `ls` (*list directory*) : affiche le contenu (fichier et sous-répertoires) d'un répertoire
- peut prendre des arguments : répertoires dont on souhaite lister le contenu.
- sans argument : liste le contenu du répertoires de travail

```
1  benoit$ pwd
2  /home/benoit
3
4  benoit$ ls
5  Bureau  MP3   Videos
6
7  benoit$ ls /usr
8  bin   sbin
9
10 benoit$ ls  ..  /
11 ..:
12 arnaud  benoit
13
14 /:
15 etc   home  tmp   usr var
```

Quelques répertoires usuels sous UNIX

Nom	**Contenu**
/	Racine du système d'exploitation
/home	Contient les répertoires des utilisateurs
/bin	Commandes essentielles aux utilisateurs
/usr/bin	La plupart des commandes des utilisateurs
/sbin	Commandes essentielles avec droits administrateur requis
/usr/sbin	La plupart des commandes avec droits administrateur requis
/tmp	Répertoire pour fichier temporaires
/var	Pour fichier dans lesquels le système écrit périodiquement
/var/log	Monitoring de l'activité du système
/etc	Fichiers de configuration
/usr	Programmes, librairies, documentation, en-têtes, ...
/mnt	Pour rattacher des clés USB, disques externes, ...(parfois /media)

Pourquoi arbres ?

Essayez de faire chez vous !

```
man hier
```

Organisation en arborescence

Rattachement des périphériques de stockage

- Un disque logique (clé USB, DVD, ...) = sous-arborescence
- Rattachée à l'arborescence générale lorsque périphérique actif
- Généralement : rattaché dans répertoire `/mnt` ou `/media`
- Montage de partition : `mount` + point de montage
  - Permet, par ex., de rattacher l'arboresence décrivant le système de fichier de la clé USB à l'arborescence du système.
  - Cette opération se fait généralement automatiquement quand le système détecte l'insertion d'une clé USB
  - + ajout raccourci sur le bureau

Système de fichiers

Représentation des fichiers et de l'arborescence

Objectifs du système de fichier

L'objectif d'un système de fichier est de définir les méthodes et mécanismes principaux permettant de :

- Stocker les fichiers : données mais également informations complémentaires, appelées méta-données : propriétaire, date de création, etc ...
- Organiser ces derniers selon une arborescence
- définir comment cette arborescence est codée

Plusieurs systèmes de fichiers existants : FAT16, FAT32, NTFS, ext2, ext3, wbfs,. . .chacun avec ses propres caractéristiques

Adressage d'un espace disque

Notion de blocs (anglais : clusters)

Sur la quasi-totalité des supports (disques durs, clés USB) :

- Espace de stockage divisé en blocs d'octets de taille identique
- Taille de ces blocs définie par le système de fichier
- Découpage en blocs fixé lors du formatage du disque
- Données d'un fichier : contenues sur un ou plusieurs blocs
- Visualisation des blocs et de leur état par divers logiciels

Remarque

- Chaque bloc possède un identifiant unique : adresse / numéro
- Cette adresse a une taille fixe, exprimée en bits (ex. 16 bits)
- Conséquence : on ne peut pas avoir un nombre de blocs illimité

Adressage d'un espace disque

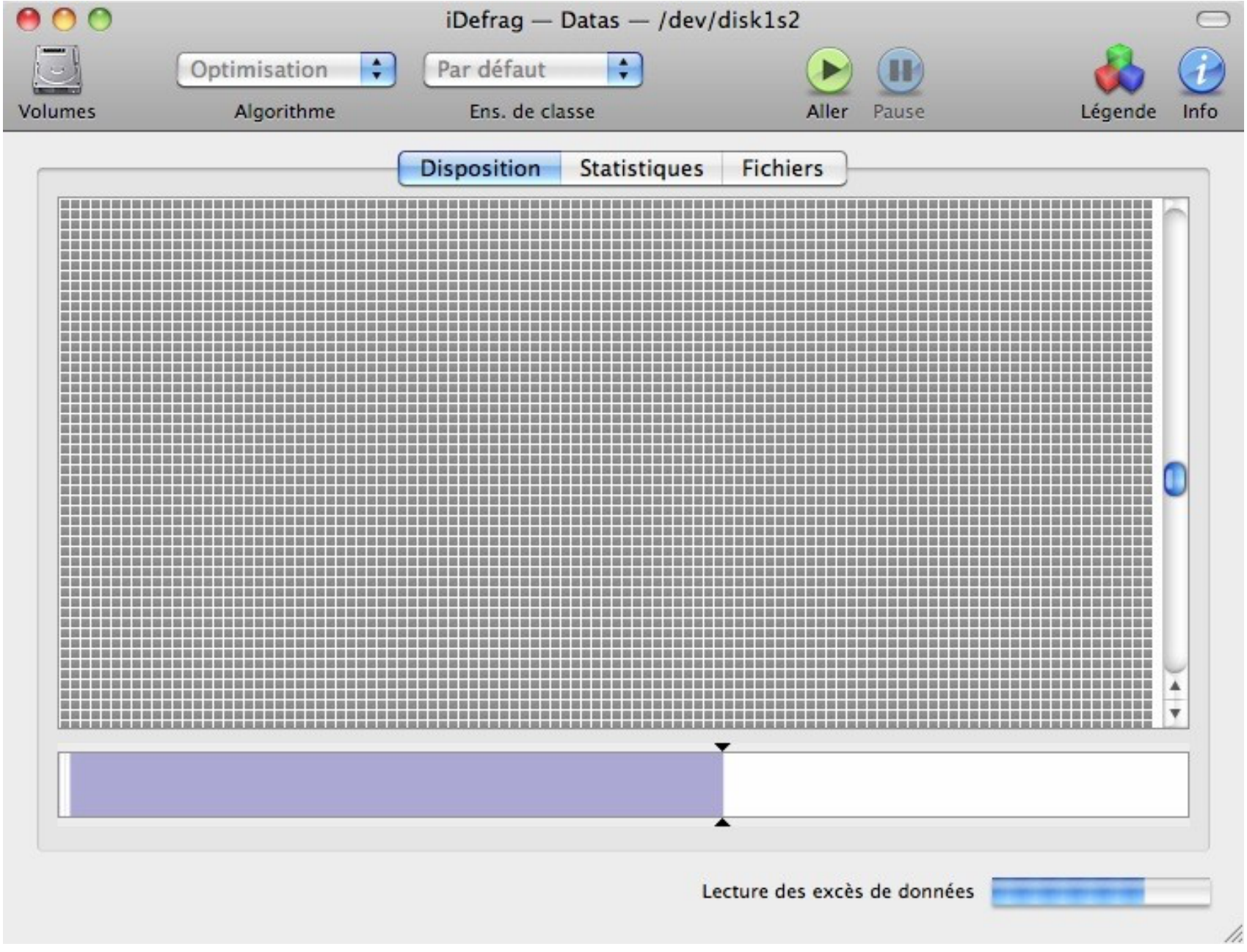
Notion de blocs (anglais : clusters)

Sur la quasi-totalité des supports (disques durs, clés USB) :

- Espace de stockage divisé en blocs d'octets de taille identique
- Taille de ces blocs définie par le système de fichier
- Découpage en blocs fixé lors du formatage du disque
- Données d'un fichier : contenues sur un ou plusieurs blocs
- Visualisation des blocs et de leur état par divers logiciels

Remarque

- Chaque bloc possède un identifiant unique : adresse / numéro
- Cette adresse a une taille fixe, exprimée en bits (ex. 16 bits)
- Conséquence : on ne peut pas avoir un nombre de blocs illimité


iDefrag — Datas — /dev/disk1s2
Volumes
Optimisation
Algorithme
Par défaut
Ens. de classe
Aller
Pause
Légende
Info
Disposition
Statistiques
Fichiers
Lecture des excès de données

pause

Question

Soit un disque dur formaté avec un système de fichier particulier, et ayant les caractéristiques suivantes :

- Le disque est de taille 4 To
- La taille de chaque bloc est de 64 Ko
- Taille d'une adresse de bloc : 20 bits

Quel est le problème ?

Tout l'espace ne peut pas être adressé !

- Taille d'une adresse : 20 bits $\Rightarrow 2^{20}$ adresses de blocs possibles
- $\Rightarrow 2^{20} \times 64$ Ko $\equiv 67$ Go utilisables au maximum

pause

Question

Soit un disque dur formaté avec un système de fichier particulier, et ayant les caractéristiques suivantes :

- Le disque est de taille 4 To
- La taille de chaque bloc est de 64 Ko
- Taille d'une adresse de bloc : 20 bits

Quel est le problème ?

Tout l'espace ne peut pas être adressé !

- Taille d'une adresse : 20 bits $\Rightarrow 2^{20}$ adresses de blocs possibles
- $\Rightarrow 2^{20} \times 64$ Ko $\equiv 67$ Go utilisables au maximum

Organisation des fichier en blocs

Relation entre fichiers et blocs 1/2

- En fonction de sa taille, un fichier occupe un ou plusieurs blocs
- Chaque bloc utilisé l'est pour un et un seul fichier
- Taille réelle du fichier vs espace occupé sur le disque
- Besoin de stocker la taille réelle d'un fichier, et pas juste le nombre de blocs occupés
- Importance de la taille d'un bloc sur l'espace perdu, le nombre de blocs d'un fichier, la vitesse d'accès, ...

Organisation des fichier en blocs

Relation entre fichiers et blocs 2/2

- Si tous les blocs qui constituent un fichier sont consécutifs, le fichier est dit défragmenté
- Sinon : fichier fragmenté
- Multiples causes à la fragmentation d'un disque
- Défragmentation de disque : réordonner les contenus des blocs afin que pour un fichier donné, les blocs qui le composent soient consécutifs

Remarque

Pour un fichier donné, il faudra stocker la liste de tous les blocs qui le composent. Problème difficile.

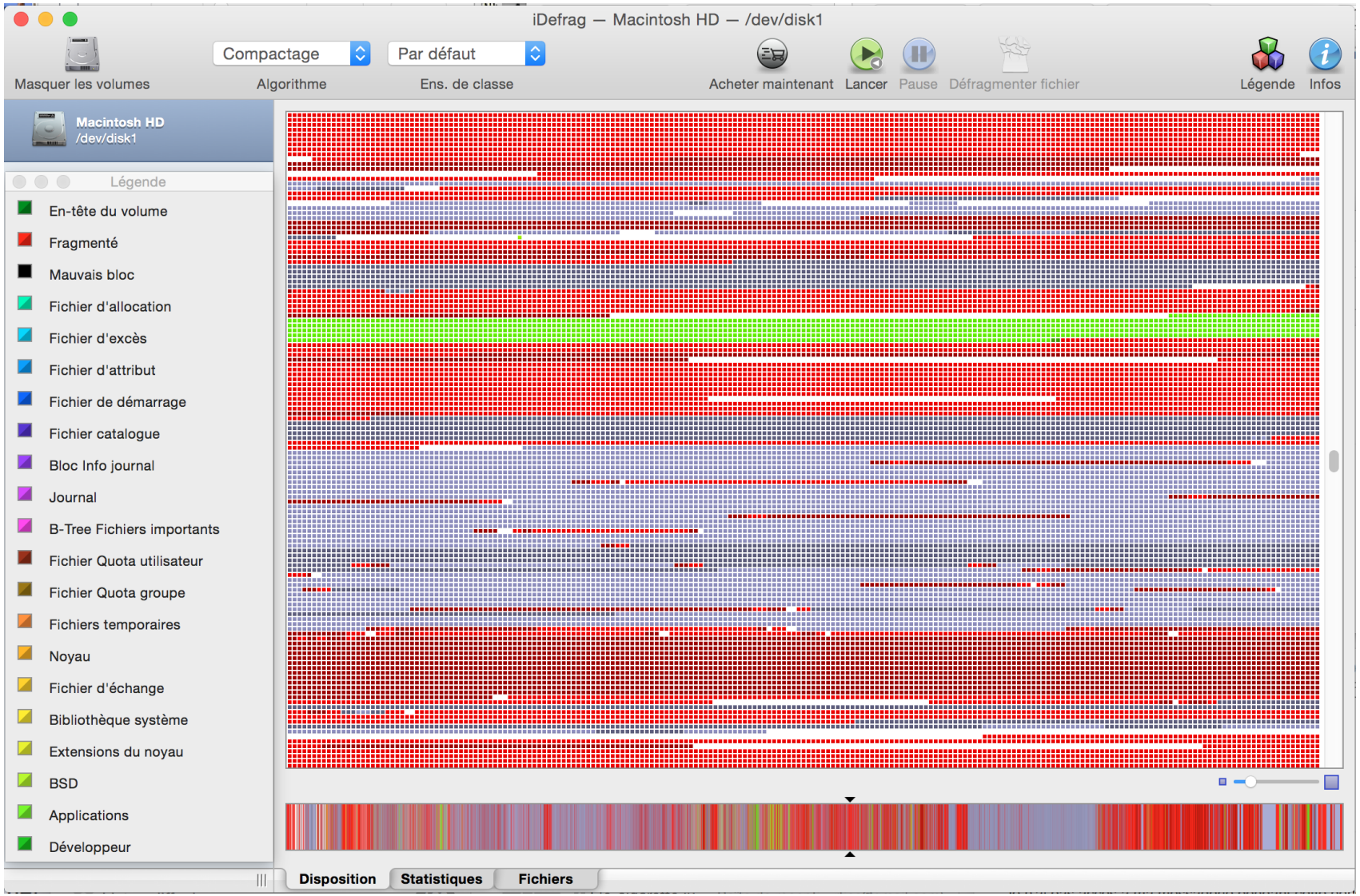
Organisation des fichier en blocs

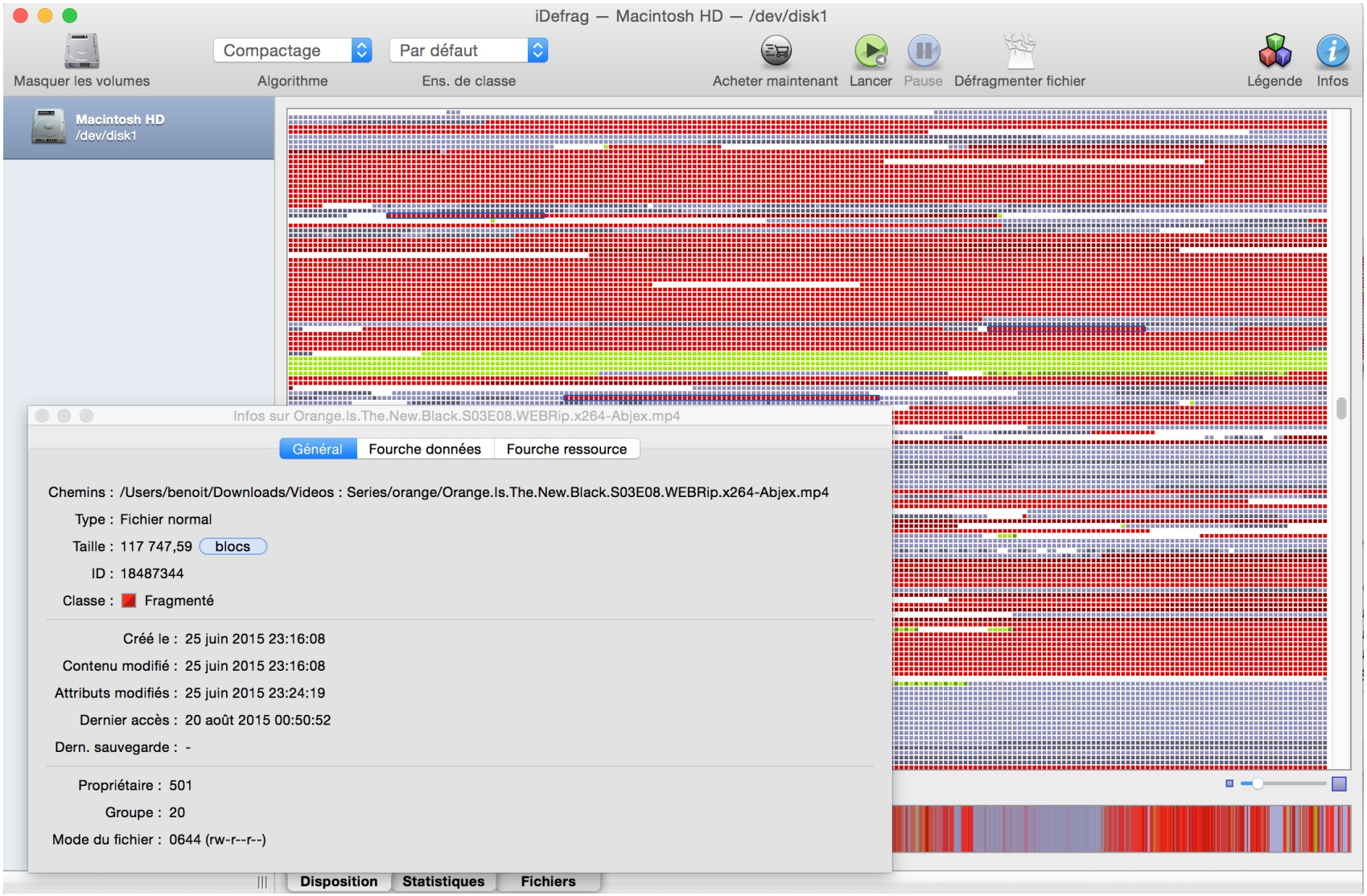
Relation entre fichiers et blocs 2/2

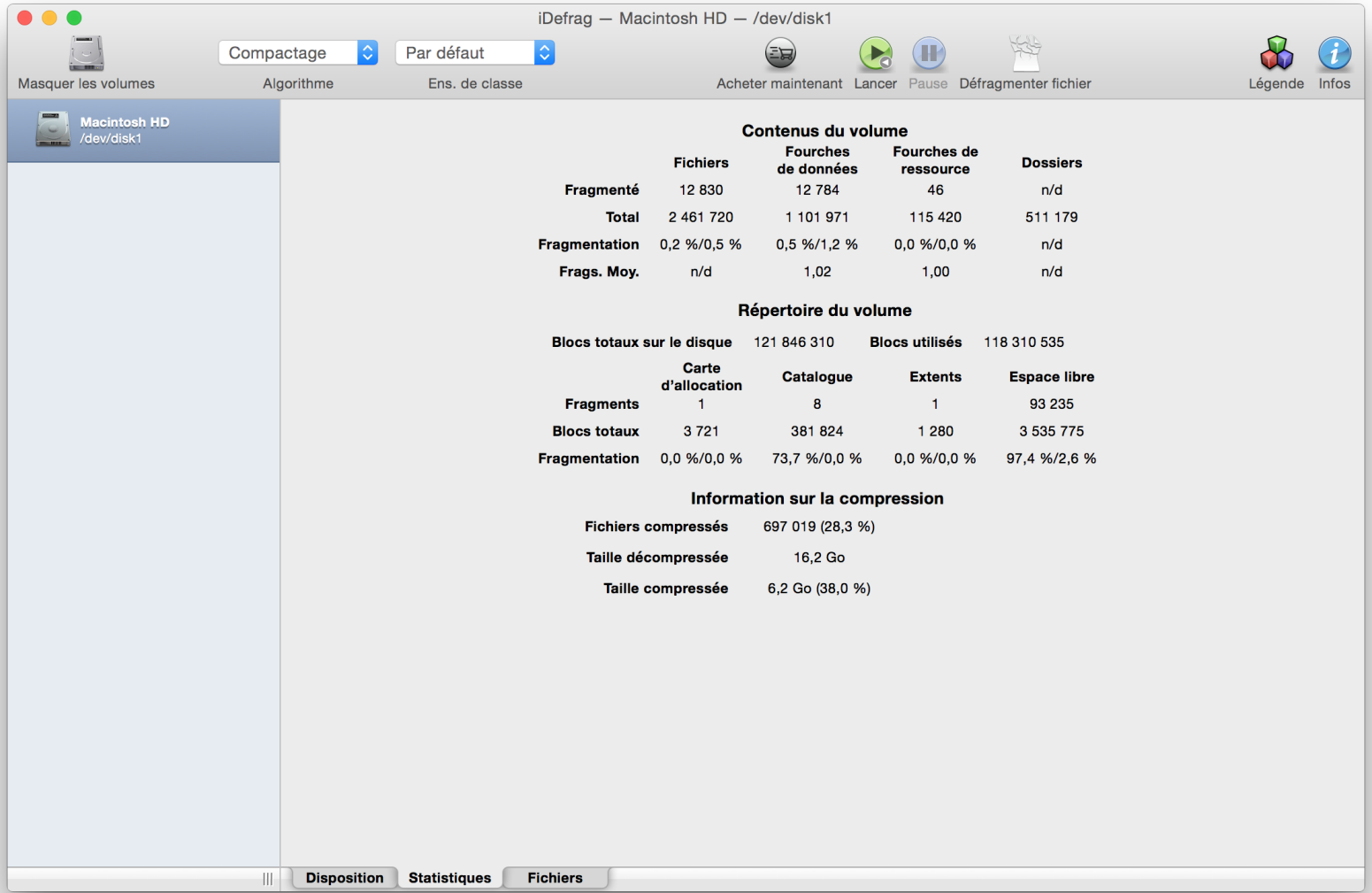
- Si tous les blocs qui constituent un fichier sont consécutifs, le fichier est dit défragmenté
- Sinon : fichier fragmenté
- Multiples causes à la fragmentation d'un disque
- Défragmentation de disque : réordonner les contenus des blocs afin que pour un fichier donné, les blocs qui le composent soient consécutifs

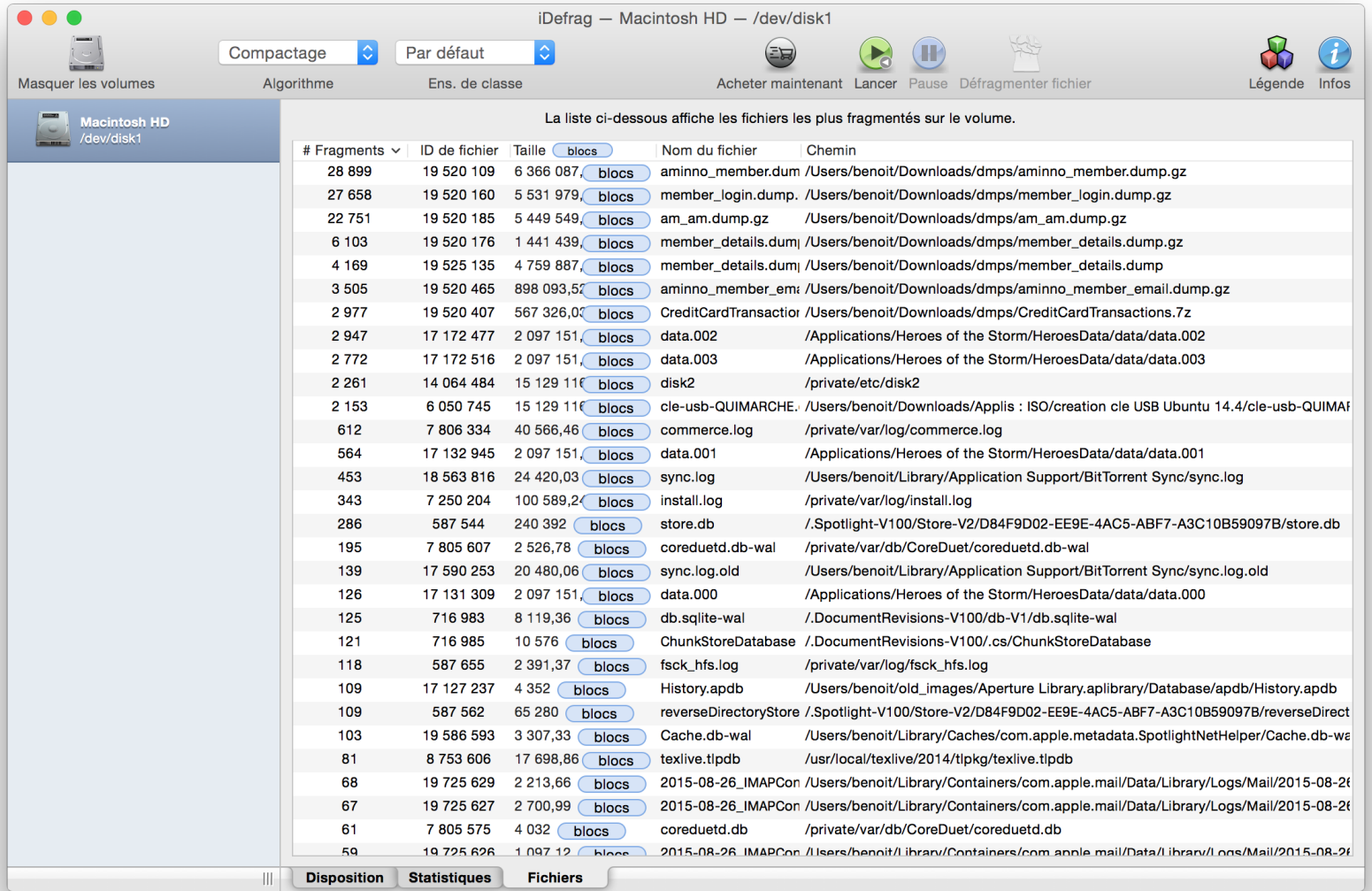
Remarque

Pour un fichier donné, il faudra stocker la liste de tous les blocs qui le composent. Problème difficile.


iDefrag — Macintosh HD — /dev/disk1
Masquer les volumes
Compactage
Algorithme
Par défaut
Ens. de classe
Acheter maintenant
Lancer
Pause
Défragmenter fichier
Légende
Infos
Macintosh HD
/dev/disk1
Légende
En-tête du volume
Fragmenté
Mauvais bloc
Fichier d'allocation
Fichier d'excès
Fichier d'attribut
Fichier de démarrage
Fichier catalogue
Bloc Info journal
Journal
B-Tree Fichiers importants
Fichier Quota utilisateur
Fichier Quota groupe
Fichiers temporaires
Noyau
Fichier d'échange
Bibliothèque système
Extensions du noyau
BSD
Applications
Développeur
Disposition
Statistiques
Fichiers


iDefrag — Macintosh HD — /dev/disk1
Masquer les volumes
Compactage
Algorithme
Par défaut
Ens. de classe
Acheter maintenant
Lancer
Pause
Défragmenter fichier
Légende
Infos
Macintosh HD
/dev/disk1
Infos sur Orange.Is.The.New.Black.S03E08.WEBRip.x264-Abjex.mp4
Général
Fourche données
Fourche ressource
Chemins : /Users/benoit/Downloads/Videos : Series/orange/Orange.Is.The.New.Black.S03E08.WEBRip.x264-Abjex.mp4
Type : Fichier normal
Taille : 117 747,59 blocs
ID : 18487344
Classe : Fragmenté
Créé le : 25 juin 2015 23:16:08
Contenu modifié : 25 juin 2015 23:16:08
Attributs modifiés : 25 juin 2015 23:24:19
Dernier accès : 20 août 2015 00:50:52
Dern. sauvegarde : -
Propriétaire : 501
Groupe : 20
Mode du fichier : 0644 (rw-r--r--)
Disposition
Statistiques
Fichiers


iDefrag — Macintosh HD — /dev/disk1
Masquer les volumes
Compactage
Algorithme
Par défaut
Ens. de classe
Acheter maintenant
Lancer
Pause
Défragmenter fichier
Légende
Infos
Macintosh HD
/dev/disk1
Contenus du volume
Fichiers
Fourches de données
Fourches de ressource
Dossiers
Fragmenté 12 830 12 784 46 n/d
Total 2 461 720 1 101 971 115 420 511 179
Fragmentation 0,2 %/0,5 % 0,5 %/1,2 % 0,0 %/0,0 % n/d
Frags. Moy. n/d 1,02 1,00 n/d
Répertoire du volume
Blocs totaux sur le disque 121 846 310
Blocs utilisés 118 310 535
Carte d'allocation
Catalogue
Extents
Espace libre
Fragments 1 8 1 93 235
Blocs totaux 3 721 381 824 1 280 3 535 775
Fragmentation 0,0 %/0,0 % 73,7 %/0,0 % 0,0 %/0,0 % 97,4 %/2,6 %
Information sur la compression
Fichiers compressés 697 019 (28,3 %)
Taille décompressée 16,2 Go
Taille compressée 6,2 Go (38,0 %)
Disposition
Statistiques
Fichiers


iDefrag — Macintosh HD — /dev/disk1
Masquer les volumes
Compactage
Algorithme
Par défaut
Ens. de classe
Acheter maintenant
Lancer
Pause
Défragmenter fichier
Légende
Infos
Macintosh HD
/dev/disk1
La liste ci-dessous affiche les fichiers les plus fragmentés sur le volume.
Fragments
ID de fichier
Taille
blocs
Nom du fichier
Chemin
28 899 19 520 109 6 366 087, blocs aminno_member.dum /Users/benoit/Downloads/dmps/aminno_member.dump.gz
27 658 19 520 160 5 531 979, blocs member_login.dump. /Users/benoit/Downloads/dmps/member_login.dump.gz
22 751 19 520 185 5 449 549, blocs am_am.dump.gz /Users/benoit/Downloads/dmps/am_am.dump.gz
6 103 19 520 176 1 441 439, blocs member_details.dum /Users/benoit/Downloads/dmps/member_details.dump.gz
4 169 19 525 135 4 759 887, blocs member_details.dum /Users/benoit/Downloads/dmps/member_details.dump
3 505 19 520 465 898 093,52 blocs aminno_member_ema /Users/benoit/Downloads/dmps/aminno_member_email.dump.gz
2 977 19 520 407 567 326,03 blocs CreditCardTransactior /Users/benoit/Downloads/dmps/CreditCardTransactions.7z
2 947 17 172 477 2 097 151, blocs data.002 /Applications/Heroes of the Storm/HeroesData/data/data.002
2 772 17 172 516 2 097 151, blocs data.003 /Applications/Heroes of the Storm/HeroesData/data/data.003
2 261 14 064 484 15 129 11 blocs disk2 /private/etc/disk2
2 153 6 050 745 15 129 11 blocs cle-usb-QUIMARCHE. /Users/benoit/Downloads/Applis : ISO/creation cle USB Ubuntu 14.4/cle-usb-QUIMAI
612 7 806 334 40 566,46 blocs commerce.log /private/var/log/commerce.log
564 17 132 945 2 097 151, blocs data.001 /Applications/Heroes of the Storm/HeroesData/data/data.001
453 18 563 816 24 420,03 blocs sync.log /Users/benoit/Library/Application Support/BitTorrent Sync/sync.log
343 7 250 204 100 589,24 blocs install.log /private/var/log/install.log
286 587 544 240 392 blocs store.db /.Spotlight-V100/Store-V2/D84F9D02-EE9E-4AC5-ABF7-A3C10B59097B/store.db
195 7 805 607 2 526,78 blocs coreduetd.db-wal /private/var/db/CoreDuet/coreduetd.db-wal
139 17 590 253 20 480,06 blocs sync.log.old /Users/benoit/Library/Application Support/BitTorrent Sync/sync.log.old
126 17 131 309 2 097 151, blocs data.000 /Applications/Heroes of the Storm/HeroesData/data/data.000
125 716 983 8 119,36 blocs db.sqlite-wal /.DocumentRevisions-V100/db-V1/db.sqlite-wal
121 716 985 10 576 blocs ChunkStoreDatabase /.DocumentRevisions-V100/.cs/ChunkStoreDatabase
118 587 655 2 391,37 blocs fsck_hfs.log /private/var/log/fsck_hfs.log
109 17 127 237 4 352 blocs History.apdb /Users/benoit/old_images/Aperture Library.aplibrary/Database/apdb/History.apdb
109 587 562 65 280 blocs reverseDirectoryStore /.Spotlight-V100/Store-V2/D84F9D02-EE9E-4AC5-ABF7-A3C10B59097B/reverseDirect
103 19 586 593 3 307,33 blocs Cache.db-wal /Users/benoit/Library/Caches/com.apple.metadata.SpotlightNetHelper/Cache.db-wa
81 8 753 606 17 698,86 blocs texlive.tlpdb /usr/local/texlive/2014/tlpkg/texlive.tlpdb
68 19 725 629 2 213,66 blocs 2015-08-26_IMAPCon /Users/benoit/Library/Containers/com.apple.mail/Data/Library/Logs/Mail/2015-08-2
67 19 725 627 2 700,99 blocs 2015-08-26_IMAPCon /Users/benoit/Library/Containers/com.apple.mail/Data/Library/Logs/Mail/2015-08-2
61 7 805 575 4 032 blocs coreduetd.db /private/var/db/CoreDuet/coreduetd.db
59 19 725 626 1 097 12 blocs 2015-08-26_IMAPCon /Users/benoit/Library/Containers/com.apple.mail/Data/Library/Logs/Mail/2015-08-2
Disposition
Statistiques
Fichiers

Codage de l'arborescence sur les systèmes UNIX

Table d'attributs des fichiers

Table multi-colonnes où chaque ligne correspond à un fichier.

- Contient les informations générales sur les fichiers : propriétaire, taille, dates d'accès, ...
- Tous les fichiers représentés : fichiers réguliers, répertoires, ...
- Chaque fichier est identifié par un numéro unique : son inode
- Permet de récupérer la liste des blocs qui contiennent les données du fichier

Remarque

Le nom du fichier n'est pas stocké dans la table d'attributs ! ! ! !(?)

Codage de l'arborescence sur les systèmes UNIX

Table d'attributs des fichiers

Table multi-colonnes où chaque ligne correspond à un fichier.

- Contient les informations générales sur les fichiers : propriétaire, taille, dates d'accès, ...
- Tous les fichiers représentés : fichiers réguliers, répertoires, ...
- Chaque fichier est identifié par un numéro unique : son inode
- Permet de récupérer la liste des blocs qui contiennent les données du fichier

Remarque

Le nom du fichier n'est pas stocké dans la table d'attributs ! ! ! !(?)

Visualiser l'inode d'un fichier

Commande `ls` avec option `-i`

- Permet de visualiser l'inode d'un fichier
- Avec paramètres : inode du fichier paramètre
- Sans paramètres : inode de tous les fichiers du répertoire courant

```
1  benoit$ ls -i fichier.jpg
2  191 fichier.jpg
3
4  benoit$ ls -i
5  42  Bureau  675 MP3
6  180 Videos  191 Fichier.jpg
```

Liste des attributs couramment stockés dans la table d'attributs

- Inode
- Propriétaire (UID)
- Groupe (GID)
- Type de fichier
- Droits sur le fichier
- Taille du fichier
- Nombre de liens pointant sur le fichier
- Nombre de blocs utilisés
- Date de création / accès / modification
- Adresses des blocs du fichier
- et bien d'autres...

Exemple d'une table d'attribut des fichiers

Num inode	**Propriétaire**	**...**	**Type**	**Taille**	**...**
2	50 (Benoit)	...	Répertoire	260	...
4	50 (Benoit)	...	Répertoire	110	...
8	50 (Benoit)	...	Répertoire	254	...
15	50 (Benoit)	...	Répertoire	40	...
16	50 (Benoit)	...	Répertoire	110	...
23	50 (Benoit)	...	Répertoire	120	...
24	50 (Benoit)	...	Répertoire	120	...
42	50 (Benoit)	...	Répertoire	138	...
135	50 (Benoit)	...	Régulier	132796	...
180	50 (Benoit)	...	Répertoire	420	...
191	50 (Benoit)	...	Régulier	19420	...
249	50 (Benoit)	...	Régulier	3427900	...
675	50 (Benoit)	...	Répertoire	420	...
...					

Visualiser les informations de d'un fichier

Commande `stat`

- Affiche les informations d'un ou plusieurs fichiers
- Permet de spécifier un format des champs a afficher avec `-f`
  - %N : nom %HT : type de fichiers %i : inode
  - %b : nombre de blocs alloués %z : taille
- Plus d'info : `man stat`

```
1  benoit$ stat fichier.jpg
2  16777220 191 -rw-rw-rw- 1 benoit staff 0 19420 "Jul  6 02:59:59 2015" "Jul  6
       04:36:33 2015" "Jul  6 04:36:33 2015" "Jul  6 02:59:59 2015" 4096 8 0 fichier.
       jpg
3
4  benoit$ stat -f "\%N \%i : \%z" fichier.jpg
5  fichier.jpg 191 : 19420
6
7  benoit$ stat -f "Le fichier \%N de type '\%HT' occupe \%b bloc(s)" fichier.jpg
8  Le fichier fichier.jpg de type 'Regular File' occupe 48 bloc(s)
```

Codage de l'arborescence sur les systèmes UNIX

Codage de l'arborescence :

- Un répertoire est vu comme un fichier : il possède un numéro d'inode, et une entrée dans la table des attributs
- Quand un fichier est un répertoire :
  - Les données de ce répertoire sont la liste des fichiers réguliers et sous-répertoires qu'il contient
  - Données : liste d'éléments associant un numéro d'Inode (un fichier décrit dans la table d'attributs) à un **nom de fichier**

Exemple de données d'un fichier répertoire

Le fichier répertoire d'inode 42 contient les éléments suivants :

Num inode	Nom de fichier
42	.
23	..
135	vacances.jpg
249	EnterSandman.mp3

Codage de l'arborescence sur les systèmes UNIX

Codage de l'arborescence :

- Un répertoire est vu comme un fichier : il possède un numéro d'inode, et une entrée dans la table des attributs
- Quand un fichier est un répertoire :
  - Les données de ce répertoire sont la liste des fichiers réguliers et sous-répertoires qu'il contient
  - Données : liste d'éléments associant un numéro d'Inode (un fichier décrit dans la table d'attributs) à un **nom de fichier**

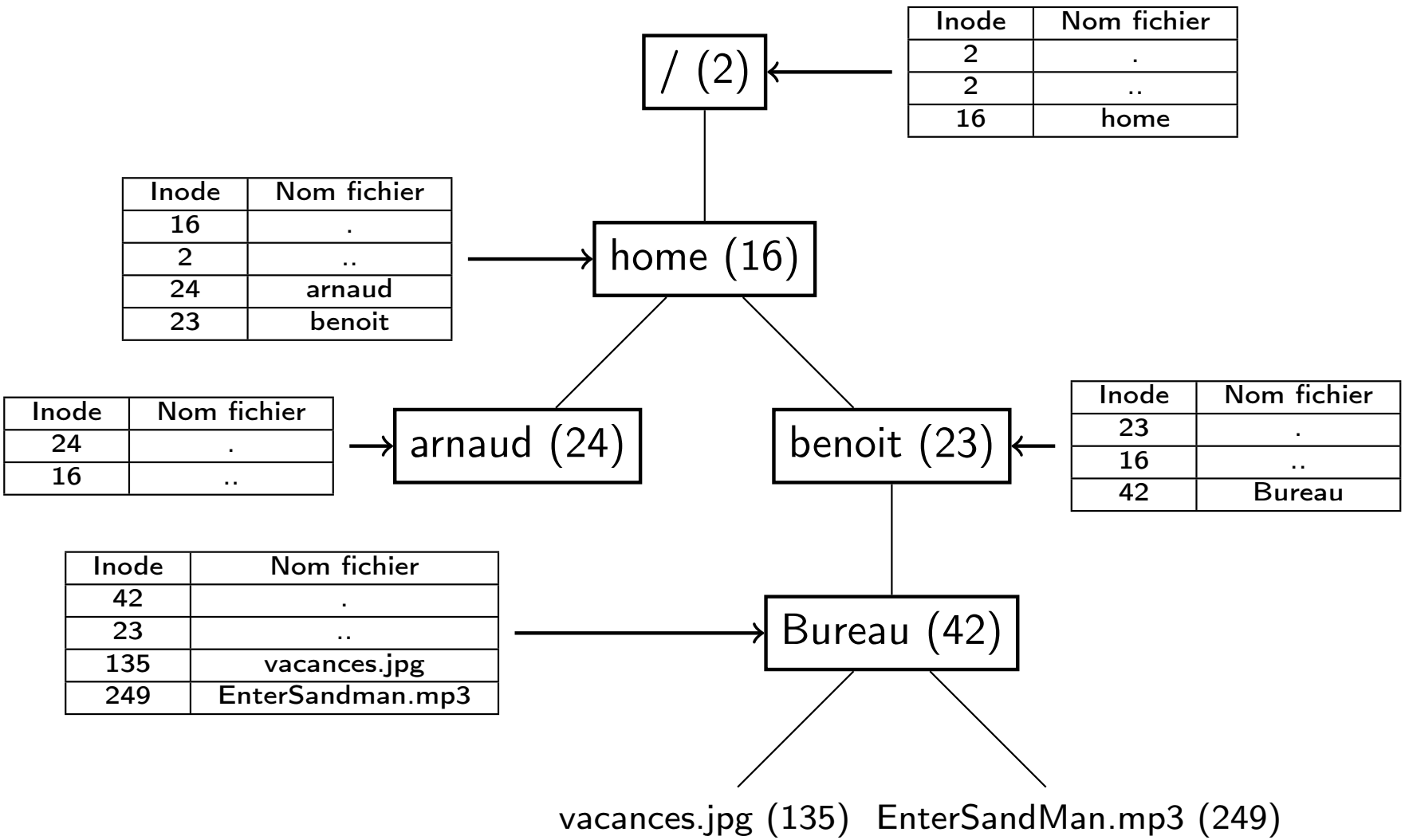
Exemple de données d'un fichier répertoire

Le fichier répertoire d'inode 42 contient les éléments suivants :

Num inode	Nom de fichier
42	.
23	..
135	vacances.jpg
249	EnterSandman.mp3

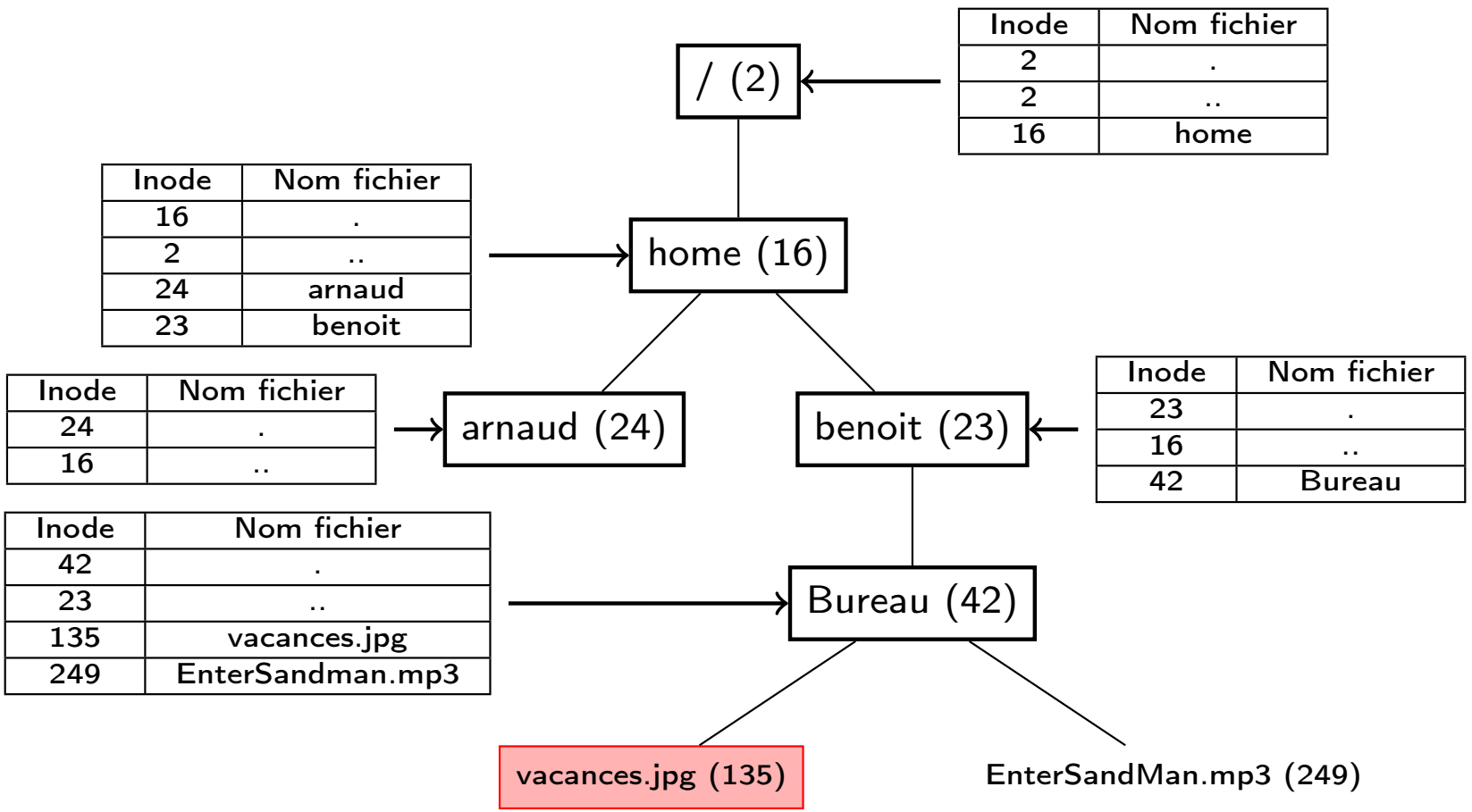
Conventions

- Chaque répertoire contient son propre numéro d'inode associé au nom '.'
- Chaque répertoire contient le numéro d'inode de son répertoire parent, associé au nom '..'
- Le répertoire racine / possède le numéro d'inode 2
- Le répertoire racine / est son propre parent


/ (2)
Inode | Nom fichier
2 | .
2 | ..
16 | home
home (16)
Inode | Nom fichier
16 | .
2 | ..
24 | arnaud
23 | benoit
arnaud (24)
Inode | Nom fichier
24 | .
16 | ..
benoit (23)
Inode | Nom fichier
23 | .
16 | ..
42 | Bureau
Bureau (42)
Inode | Nom fichier
42 | .
23 | ..
135 | vacances.jpg
249 | EnterSandman.mp3
vacances.jpg (135)
EnterSandMan.mp3 (249)

Remarque

Si un fichier (qu'il soit régulier ou répertoire) est contenu dans un répertoire, on dit qu'il possède un **lien physique** vers le répertoire.



Question

Sur le dernier schéma présenté, combien comptez-vous de liens physiques pour les fichiers suivants :

- vacances.jpg
- arnaud
- benoit
- home
- /

Question

Sur le dernier schéma présenté, combien comptez-vous de liens physiques pour les fichiers suivants :

- vacances.jpg : 1
- arnaud : 2
- benoit : 3
- home : 4
- / : 3

Questions

- Un répertoire vide contient combien de liens ?

Questions

- Un répertoire vide contient combien de liens ?
  - . — lui-même et
  - .. — son répertoire parent
- Si un répertoire X contient k répertoires, combien de liens physiques vers X existe-t-il dans le système de fichiers ?

Questions

- Un répertoire vide contient combien de liens ?
  - . — lui-même et
  - .. — son répertoire parent
- Si un répertoire X contient k répertoires, combien de liens physiques vers X existe-t-il dans le système de fichiers ?
  k + 2

Remarques

- Ajouter un fichier dans un répertoire revient simplement à ajouter un lien physique du fichier vers le répertoire
- Supprimer un fichier revient en fait à supprimer un lien physique du fichier vers le répertoire qui contenait le fichier

Questions ?