Hệ thống quản lý tập tin

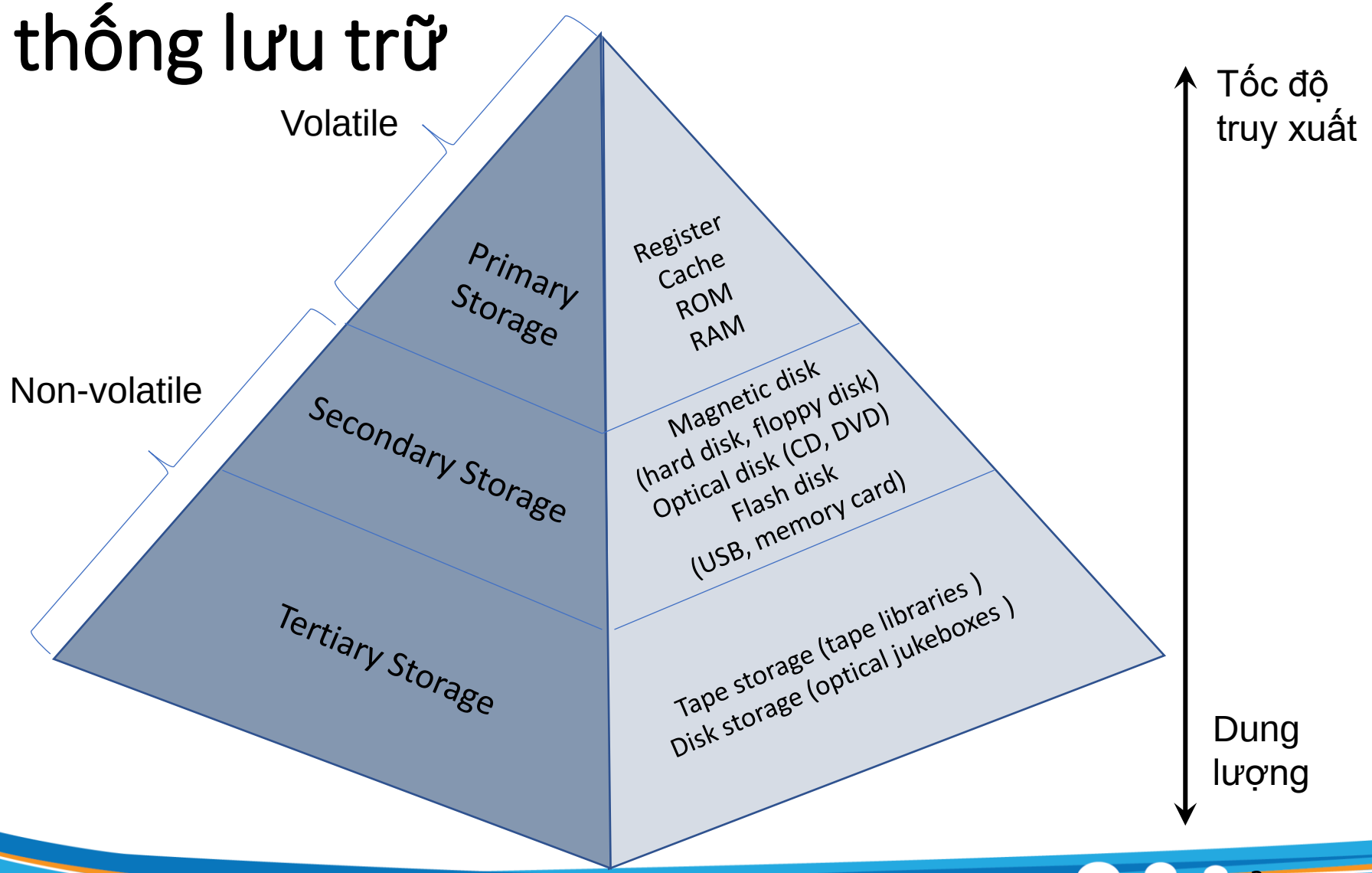
Môn học: Hệ Điều Hành

fit@hcmus
KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Nội dung

- Trình bày cấu tạo đĩa từ.
- Trình bày các khái niệm liên quan hệ thống tập tin.
- Trình bày một số vấn đề khi cài đặt hệ thống quản lý tập tin trên đĩa.
- Trình bày mô hình tổ chức hệ thống tập tin của một số hệ điều hành thông dụng.
- Hệ thống tập tin FAT.

Phân cấp hệ thống lưu trữ

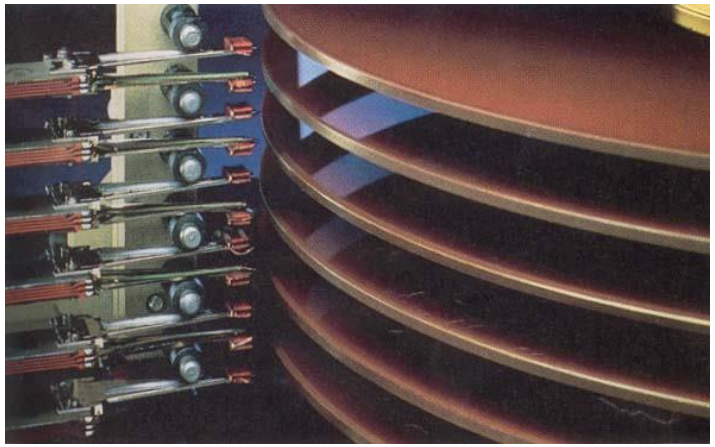


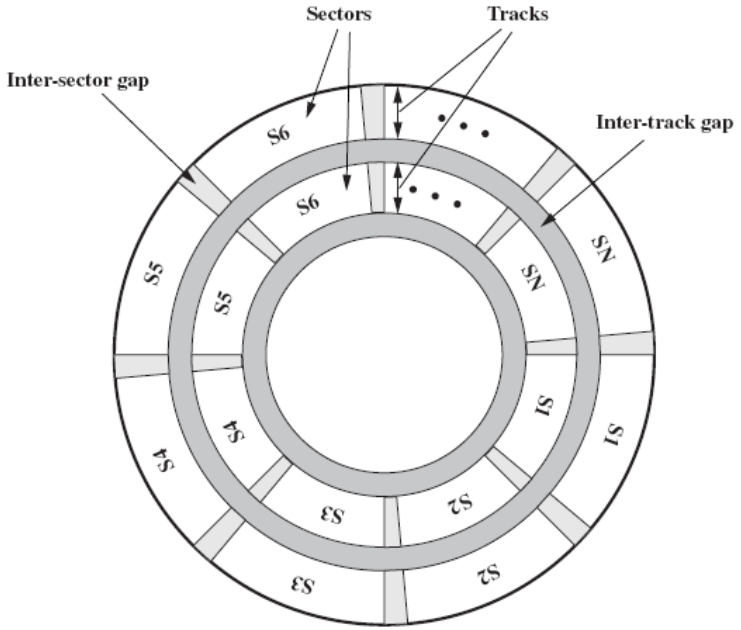
Đĩa từ

- Đĩa từ - là những đĩa phẳng bằng thủy tinh hay bằng kim loại cứng được phủ từ để lưu dữ liệu.

Cấu trúc vật lý

- Gồm nhiều lớp hình tròn, mỗi lớp phủ từ 1 hoặc cả 2 mặt (side)
- Mỗi mặt có tương ứng 1 đầu đọc (head) để đọc hoặc ghi dữ liệu
- Mỗi mặt có nhiều đường tròn đồng tâm (track)
- Mỗi đường tròn được chia nhỏ thành các cung tròn (sector), thông thường mỗi cung chứa 4096 điểm từ (~ 4096 bit = 512 byte)
- Mỗi lần đọc/ghi ít nhất 1 sector (512 byte)





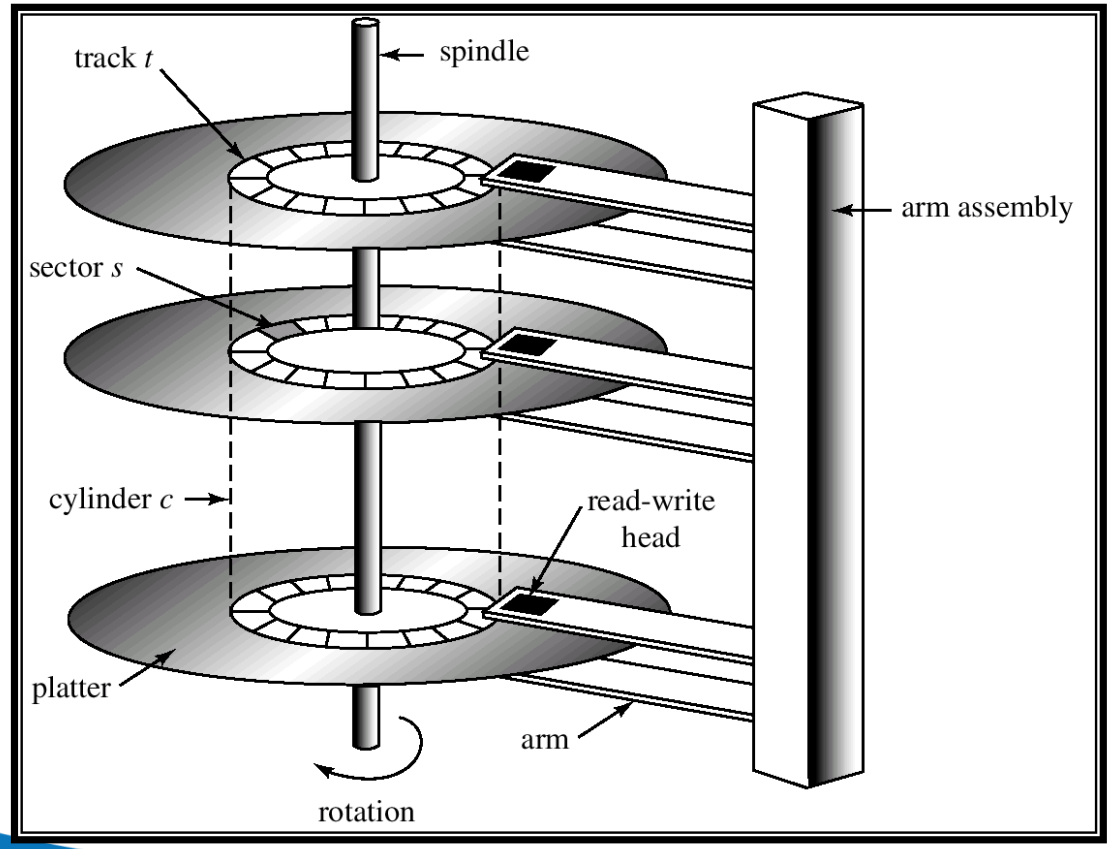
Truy xuất mức vật lý

- Để truy xuất 1 sector cần phải chỉ ra vị trí của sector đó. Vị trí sector được thể hiện bằng 3 thông số: chỉ số sector, track và head.
  - Head được đánh số từ trên xuống bắt đầu từ 0.
  - Track được đánh số theo thứ tự từ ngoài vào bắt đầu từ 0.
  - Sector được đánh số bắt đầu từ 1 theo chiều ngược với chiều quay của đĩa.
- Địa chỉ sector vật lý có ký hiệu: (sector, track, head).
- Hàm truy xuất mức vật lý trong C for DOS:

  int biosdisk (int cmd, int drive, int head, int track, int sector, int nsects, void *buffer)
- Hàm truy xuất mức vật lý trong C for Windows, Linux ???

Cơ chế đọc đĩa

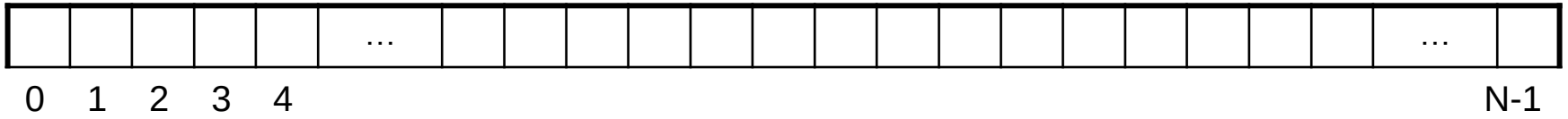
- Access time = Seek time + Rotational time + Read time



Tổ chức logic

- Do truy xuất mức vật lý phải dùng đến 3 tham số rất bất tiện nên tổ chức logic được đưa ra để dễ hiểu, dễ thao tác, dễ tính toán hơn.
- Cylinder: là tập các track có cùng bán kính (cùng số hiệu) trên tất cả các mặt.

  → Nhận xét: truy xuất sector theo từng cylinder sẽ đảm bảo *sau khi truy xuất sector K thì truy xuất sector K+1 là nhanh hơn so với tất cả các sector khác.*
- Tổ chức logic là một dãy sector được đánh chỉ số theo theo từng cylinder, bắt đầu từ 0.



- Mỗi lần truy xuất (đọc/ ghi đĩa) chỉ có thể thực hiện trên N sector liên tiếp (N>=1).
- Hàm truy xuất mức logic trong C for DOS:

  int absread (int drive, int nsects, long lsect, void *buffer)

  int abswrite (int drive, int nsects, long lsect, void *buffer)
- Hàm truy xuất mức logic trong C for Windows, Linux ???

Sector vật lý ↔ Sector logic

- Sector vật lý → Sector logic

  l = t*st*hd + h*st + s -1

- Sector logic → Sector vật lý

  s = (l mod st) + 1

  t = l div (st * hd)

  h = (l div st) mod hd

Trong đó:

l : chỉ số sector logic

h : chỉ số head

t : chỉ số track

s : chỉ số sector vật lý

st : số sector /track

th : số track /side (head)

hd : tổng số side (head)

Đĩa mềm 1.44 MB

- Có 2 head /disk, 80 track /head, 18 sector /track.
- Dung lượng đĩa:

  2 head/disk * 80 track/head * 18 sector/track

  = 2880 sector/disk

  = 0.5 KB/sector * 2880 sector/disk = 1440 KB/disk (~ 1.44 MB)
- Sector logic có chỉ số từ 0 đến 2879 và tương ứng với sector vật lý như sau:

Sector Logic	Sector vật lý (Sector, Track, Head)
0	(1, 0, 0)
1	(2, 0, 0)
…	…
17	(18, 0, 0)
18	(1, 0, 1)
19	(2, 0,1)
…	…
35	(18, 0, 1)
36	(1, 1, 0)
37	(2, 1,0)
…	…

Bài tập

1. Một đĩa cứng có 16 head, mỗi mặt có 684 track, và mỗi track có 18 sector thì sẽ có kích thước là bao nhiêu Megabyte ?
2. Cho biết sector vật lý (head 0, track 19, sector 6) tương ứng với sector logic nào trên đĩa mềm 1.44MB ?
   A. 347
   B. 348
   C. 689
   D. 690

Một số khái niệm

- Tập tin,
- Thư mục,

Bộ nhớ ngoài & Tập tin

- Một số hạn chế của bộ nhớ trong:
  - Không lưu trữ dữ liệu lâu dài.
  - Không chứa lượng thông tin lớn.

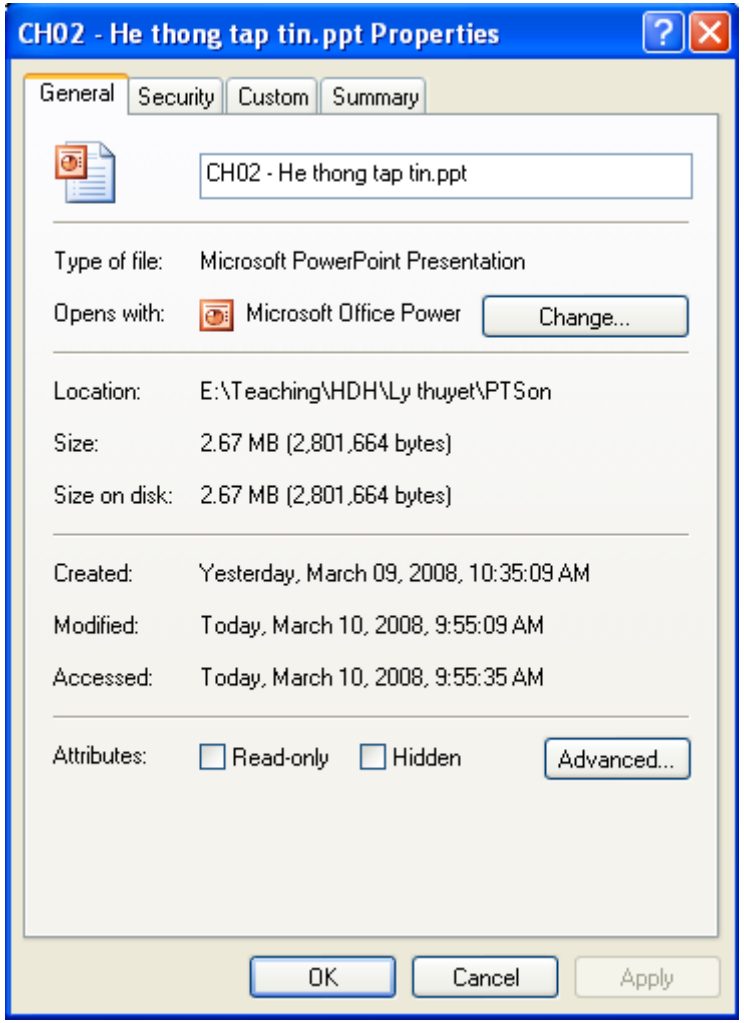
  → Cần các thiết bị lưu trữ ngoài(bộ nhớ ngoài) để lưu trữ dữ liệu.
- Tuy nhiên, có nhiều loại thiết bị lưu trữ ngoài (đĩa từ, CD/DVD, USB, thẻ nhớ,…); đa dạng về cấu trúc, khả năng lưu trữ, phương thức truy xuất, tốc độ truy xuất.
- HĐH cung cấp cái nhìn logic và đồng nhất về việc lưu trữ thông tin.
  - Trừu tượng hóa thông tin vật lý thành đơn vị lưu trữ logic – tập tin.

Tập tin

- Tập tin là gì ?
  - Lưu trữ tập hợp các thông tin có liên quan với nhau.
  - Là một đơn vị lưu trữ luận lý che tổ chức vật lý của các thiết bị lưu trữ ngoài.
  - Thường bao gồm 2 thành phần:
    - Thuộc tính.
    - Nội dung.
  - Mỗi hệ thống tập tin có cách thức tổ chức tập tin khác nhau.

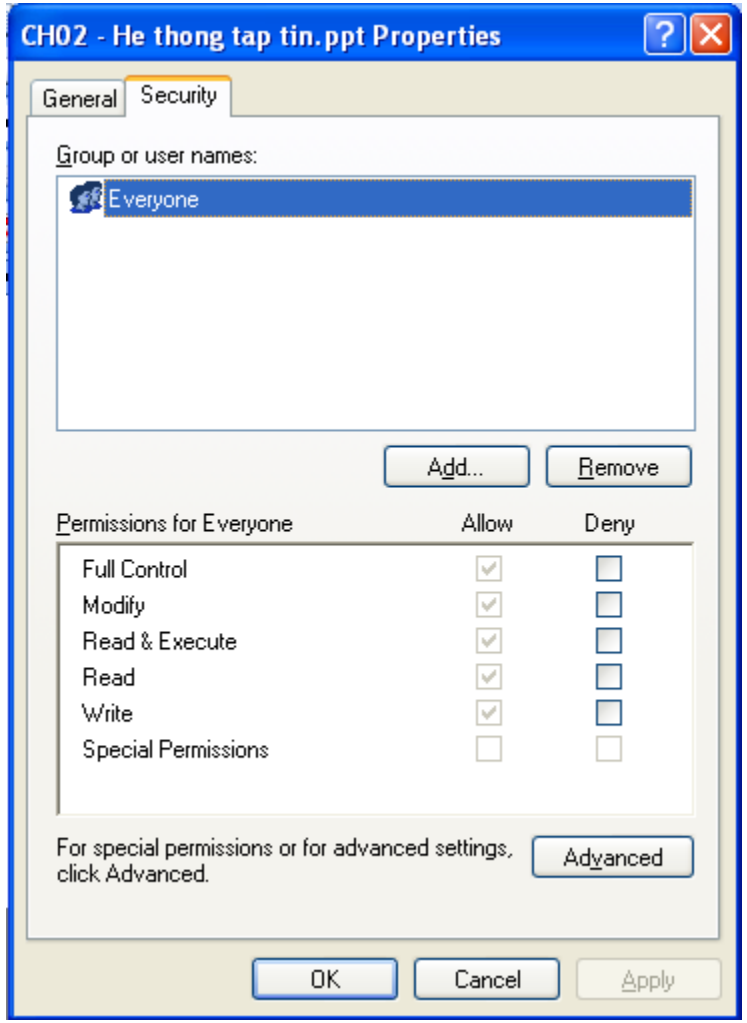
Thuộc tính tập tin

- Thuộc tính của tập tin trên các hệ thống tập tin khác nhau sẽ khác nhau, nhưng thường gồm các thuộc tính sau:
  - *Tên (tên + phần mở rộng)*
  - *Người sở hữu*
  - *Thuộc tính trạng thái: chỉ đọc, ẩn,…*
  - *Kích thước*
  - *Ngày giờ (tạo, truy cập, thay đổi)*
  - *Thuộc tính bảo vệ*
  - *Vị trí lưu trữ trên đĩa*



Cơ chế bảo vệ tập tin

- Người tạo /sở hữu tập tin có quyền kiểm soát:
  - *Ai* (người dùng /nhóm người dùng) có quyền *gì* trên tập tin.
    - Đọc
    - Ghi
    - Thực thi
    - Thêm
    - Xóa
    - Liệt kê
    - Một số quyền đặc biệt khác



Thao tác trên tập tin

- Một số thao tác cơ bản trên tập tin:
  - *Tạo*
  - *Xóa*
  - *Đọc*
  - *Ghi*
  - *Định vị (seek)*
  - *Xóa nội dung (truncate)*
  - *Mở*
  - *Đóng*
- Một số thao tác khác: sao chép, di chuyển, đổi tên, …

Một số tính chất khác của tập tin

- Cấu trúc tập tin – do HĐH hay chương trình ứng dụng quyết định:
  - Không cấu trúc.
  - Có cấu trúc.
- Loại tập tin:
  - Tập tin văn bản (text file): chứa các dòng văn bản, cuối dùng có ký hiệu kết thúc dòng (end line).
  - Tập tin nhị phân (binary file): là tập tin có cấu trúc.
- Truy xuất tập tin:
  - Tuần tự - Phải đọc từ đầu tập tin đến vị trí mong muốn, có thể quay lui (rewind).
  - Ngẫu nhiên - Có thể di chuyển (seek) đến đúng vị trí cần đọc.

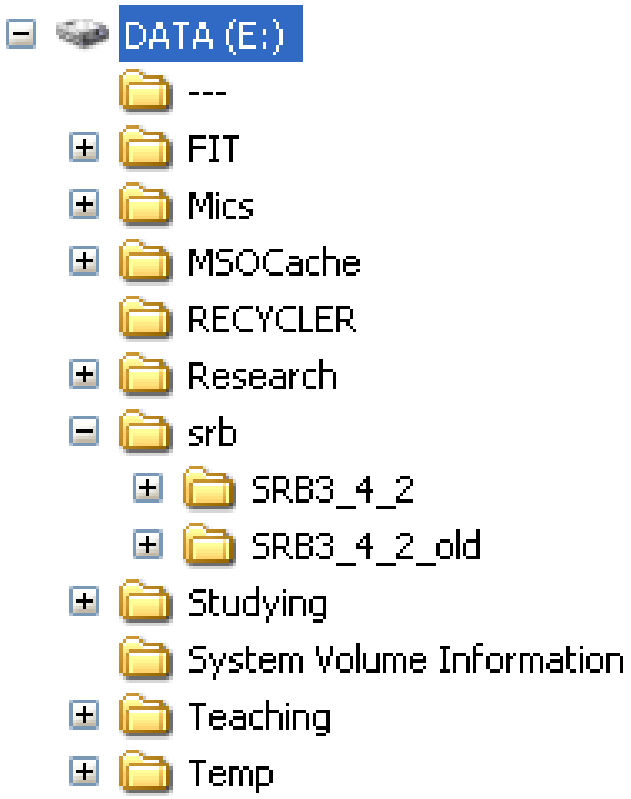
Thư mục

- Thư mục là một loại tập tin đặc biệt, giúp tổ chức có hệ thống các tập tin trên hệ thống lưu trữ ngoài.
  - Thuộc tính của thư mục tương tự của tập tin.
  - Nội dung của thư mục: quản lý các tập tin,thư mục con của nó.
    - Một cấp: đơn giản nhất, tất cả tập tin trên hệ thống cùng thư mục.
    - Hai cấp: mỗi người dùng có 1 thư mục riêng.
    - Cây phân cấp: được sử dụng phổ biến hiện nay.
- Một số thao tác trên thư mục:
  - Tạo
  - Xóa
  - Mở
  - Đóng
  - Liệt kê nội dung thư mục
  - Tìm kiếm tập tin
  - Duyệt hệ thống tập tin

Một số vấn đề tổ chức hệ thống tập tin

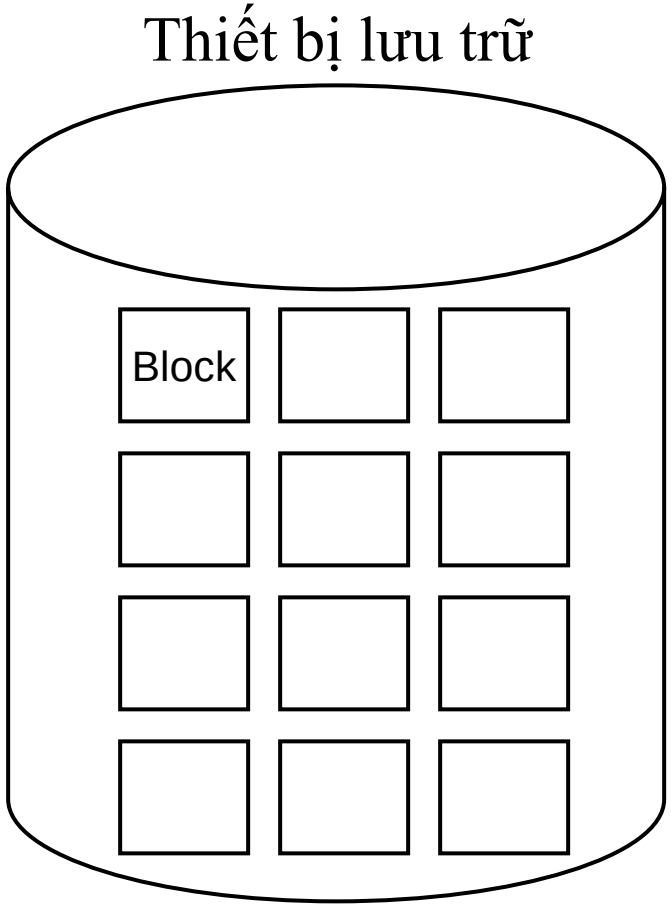
- Tổ chức thư mục.
- Tổ chức tập tin.
- Quản lý đĩa trống.
- Tổ chức hệ thống tập tin trên đĩa từ.
- Tổ chức hệ thống tập tin trong bộ nhớ.
- Kết buộc hệ thống tập tin.

Vấn đề


DATA (E:)

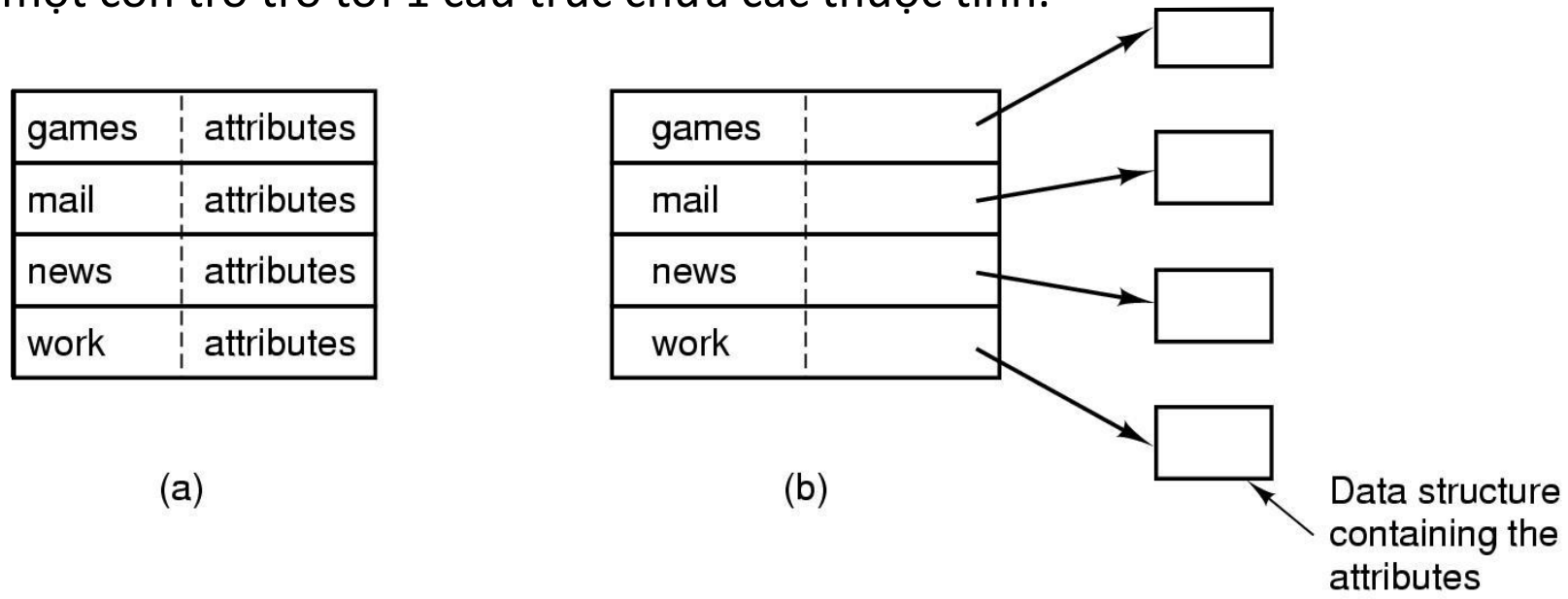
FIT
Mics
MSOCache
RECYCLER
Research
srb
SRB3_4_2
SRB3_4_2_old
Studying
System Volume Information
Teaching
Temp

???

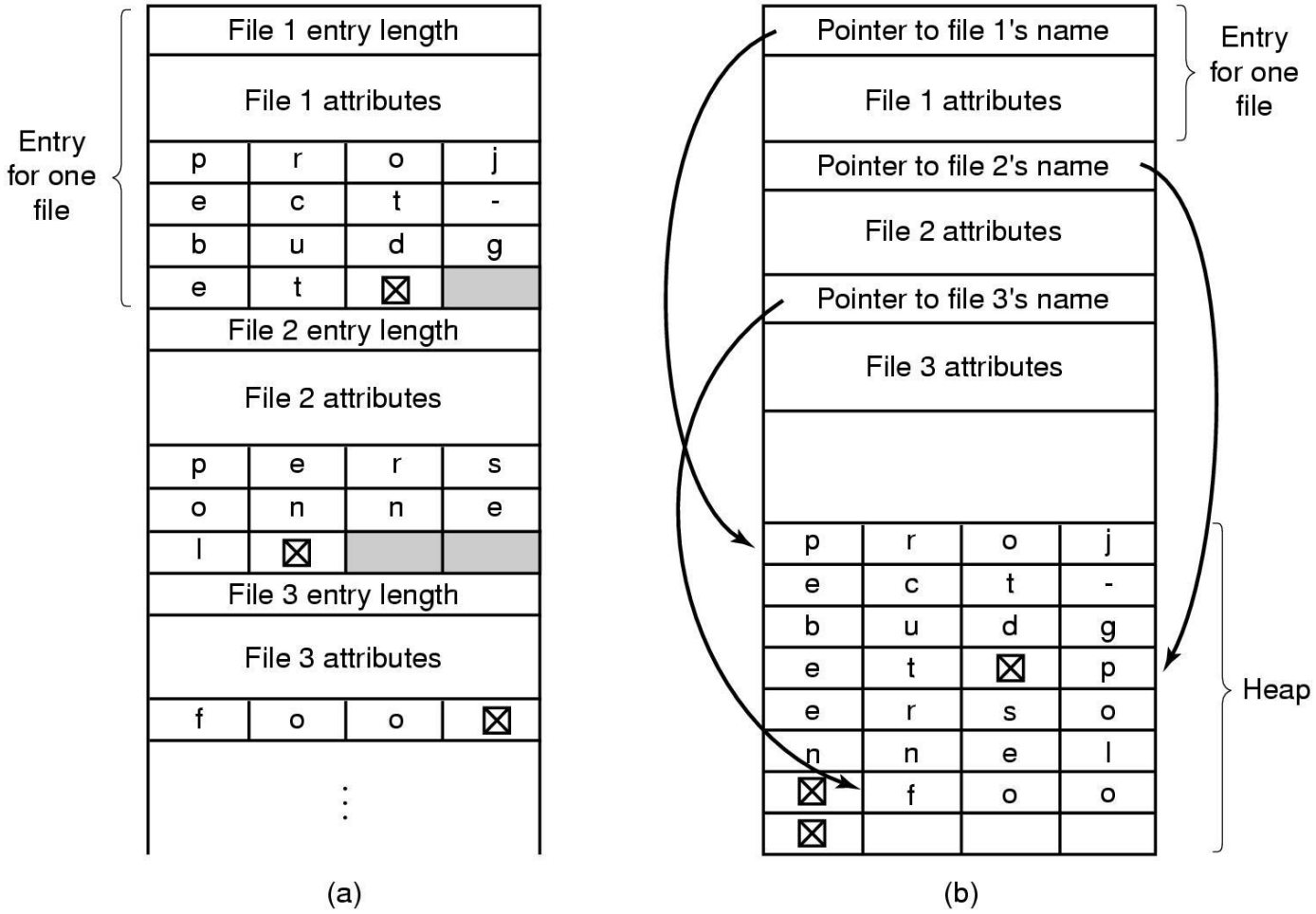

Thiết bị lưu trữ
Block

Tổ chức thư mục

- Thường được tổ chức thành một bảng các phần tử (*directory entry*), gọi là bảng thư mục
- 2 cách tổ chức directory entry:
  - Entry chứa tên và các thuộc tính.
  - Entry chứa tên và một con trỏ trỏ tới 1 cấu trúc chứa các thuộc tính.



Vấn đề tên dài (long file name - LFN)



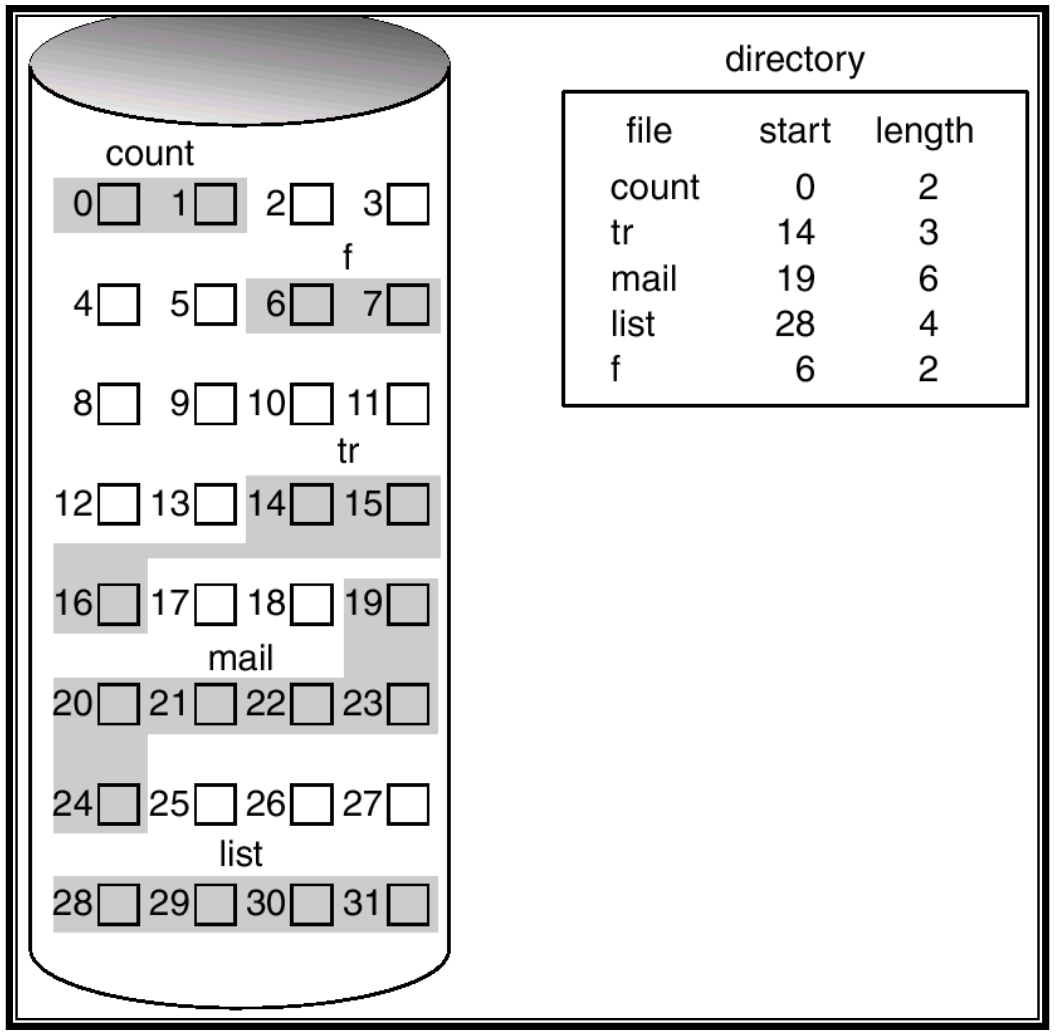
Tổ chức tập tin

- Mỗi tập tin lưu nội dung trên một số block (khối lưu trữ) của thiết bị lưu trữ.

  → Làm sao biết được tập tin đang chiếm những block nào ?
- Phương pháp cấp phát mô tả cách thức cấp phát các block cho các tập tin.
- Có 3 phương pháp cấp phát chính:
  - Cấp phát liên tục.
  - Cấp phát theo kiểu danh sách liên kết.
  - Cấp phát theo kiểu chỉ mục.

Cấp phát liên tục (1/2)

- Mỗi tập tin chiếm các block liên tục trên đĩa.
- Đơn giản, chỉ cần quản lý vị trí (chỉ số) block bắt đầu và chiều dài (số block).
- Hỗ trợ truy xuất tuần tự & truy xuất trực tiếp.
- Vấn đề External fragmentation.
- Vấn đề khi kích thước tập tin tăng.

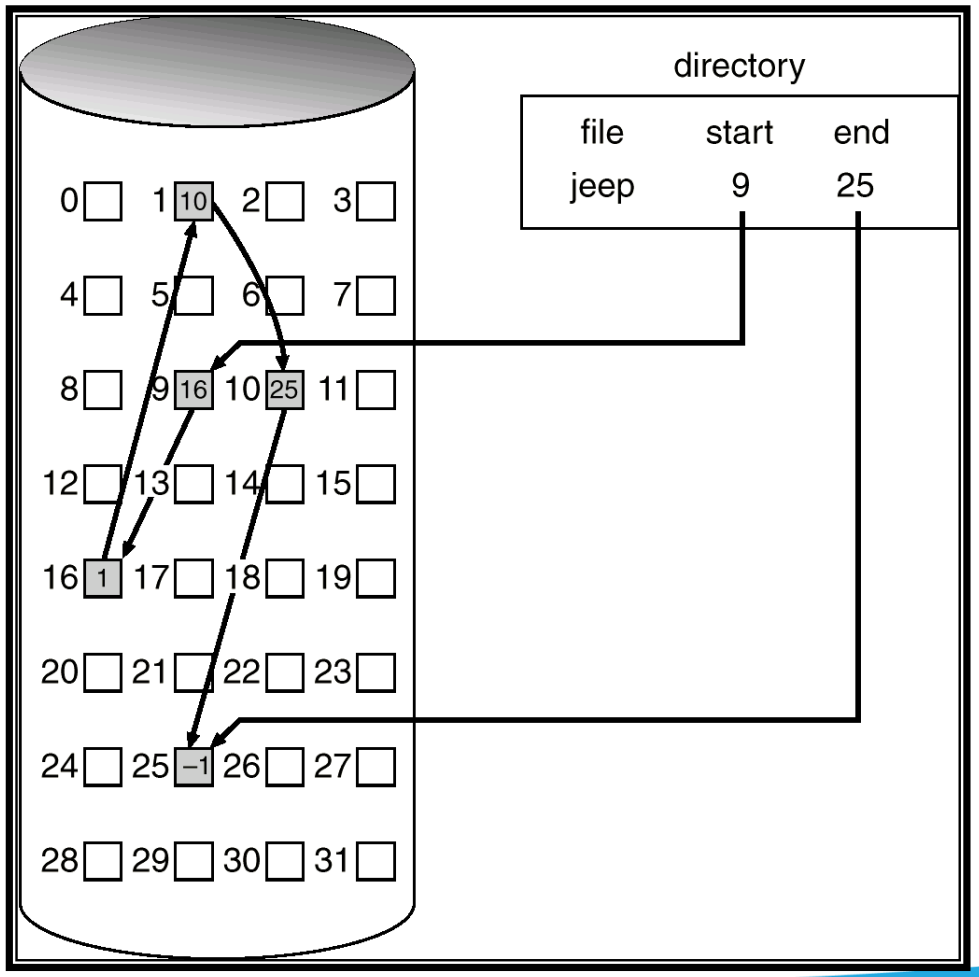


Cấp phát liên tục (2/2)

- Hệ thống tập tin cấp phát theo *extent*:
  - Extent là một tập các block liên tục.
  - Cấp phát cho tập tin theo từng extent.
  - Một tập tin có thể chiếm một hoặc nhiều extent không liên tục nhau.
  - Kích thước các extent có thể khác nhau.
  - Cần quản lý 3 thông tin: vị trí block bắt đầu, số block và một con trỏ trỏ tới block đầu tiên của extent kế tiếp.
  - Vấn đề Internal fragmentation và External fragmentation.

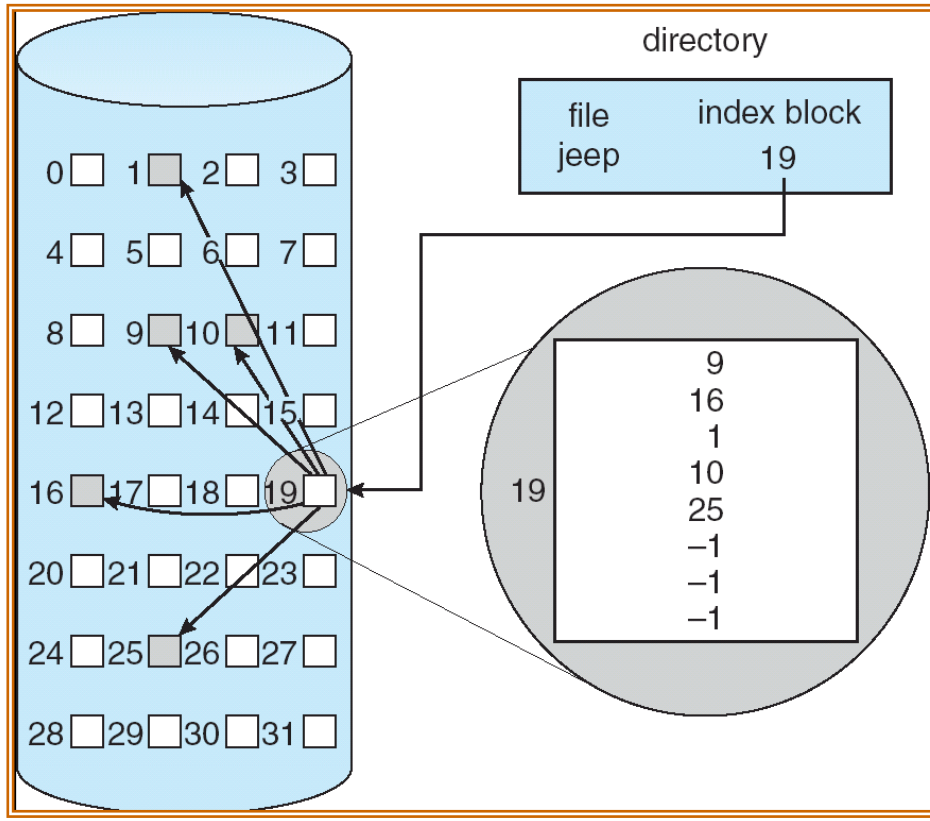
Cấp phát theo kiểu danh sách liên kết

- Mỗi tập tin chiếm một tập các block theo kiểu danh sách liên kết.
- Mỗi block sẽ chứa thông tin về địa chỉ của block kế tiếp.
- Các block có thể nằm rãi rác trên đĩa.
- Chỉ hỗ trợ truy xuất tuần tự.
- Đơn giản, chỉ cần quản lý vị trí (chỉ số) block bắt đầu.
- Không bị External fragmentation.
- Tốn chi phí lưu địa chỉ block kế tiếp.

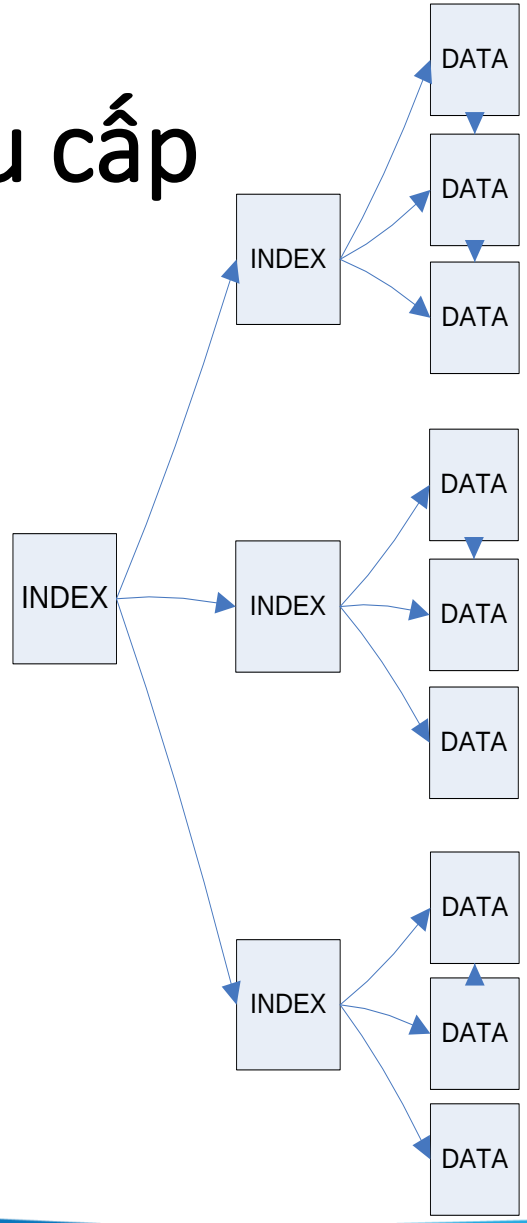


Cấp phát theo kiểu chỉ mục

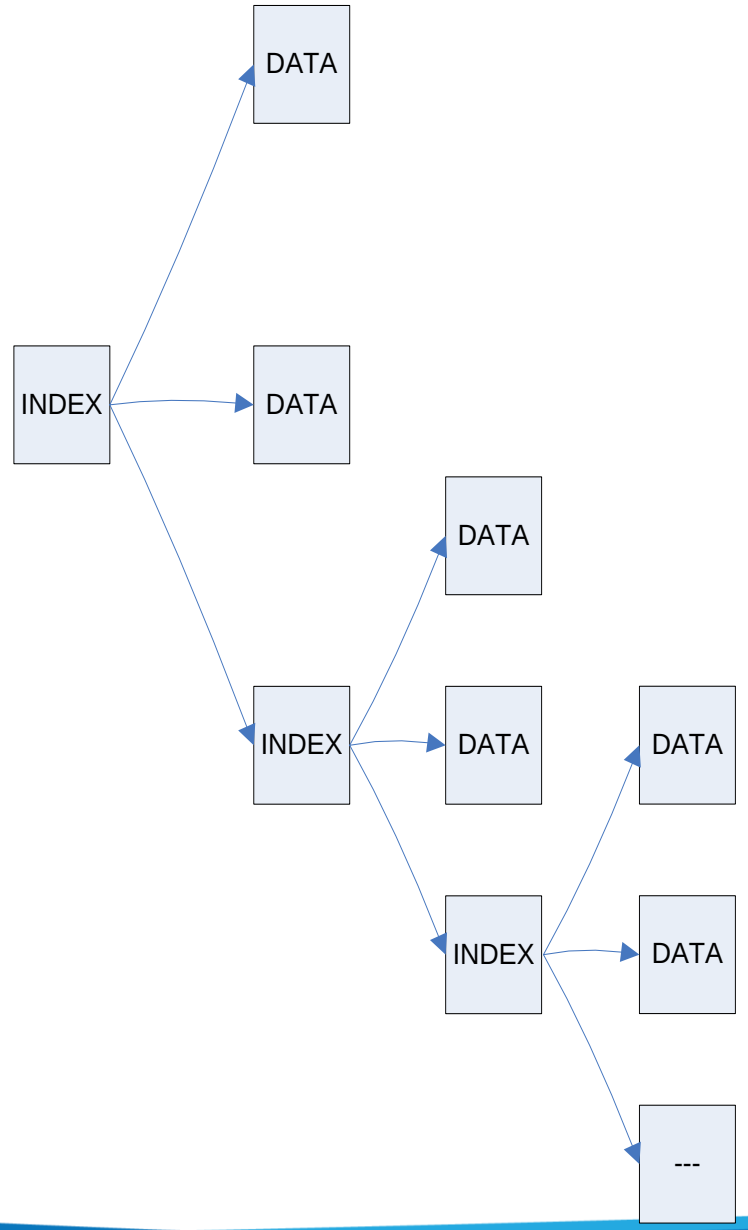
- Gồm một hoặc nhiều block làm bảng chỉ mục chứa địa chỉ của các block dữ liệu.
- Hỗ trợ truy xuất tuần tự & truy xuất trực tiếp.
- Tốn không gian đĩa để lưu các block chỉ mục.
- Không bị External fragmentation.
- Một số mô hình mở rộng:
  - Mô hình chỉ mục nhiều cấp.
  - Mô hình chỉ mục kết hợp danh sách liên kết.
  - Mô hình chỉ mục nhiều cấp kết hợp danh sách liên kết.



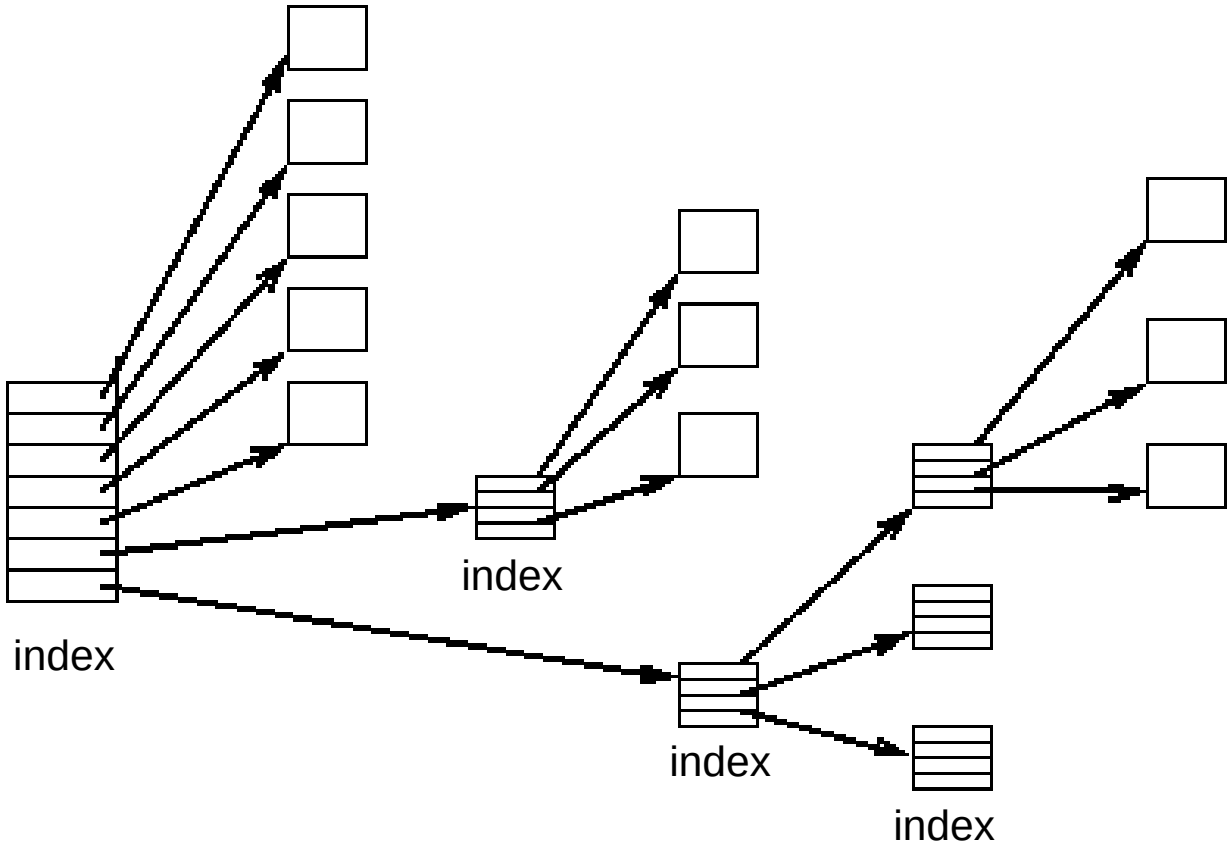
Mô hình chỉ mục nhiều cấp



Mô hình chỉ mục kết hợp danh sách liên kết

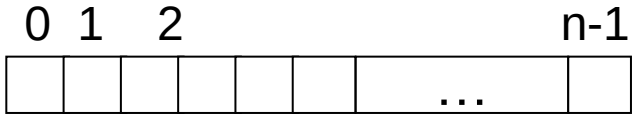


Mô hình chỉ mục nhiều cấp kết hợp danh sách liên kết



Quản lý không gian đĩa trống

- Bit vector (Bit map):
  - Mỗi block được biểu diễn bằng 1 bit.



bit[*i*] = { 0 ⇒ block[*i*] trống ; 1 ⇒ block[*i*] đã dùng

  - Bit vector tốn không gian đĩa. Ví dụ:

kích thước 1 block = 2^{12} bytes

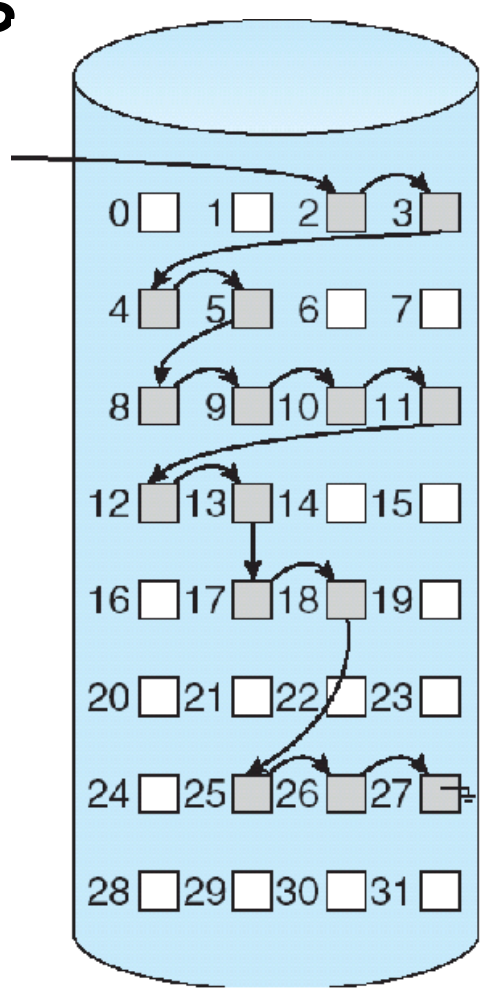
kích thước đĩa = 2^{30} bytes (1 gigabyte)

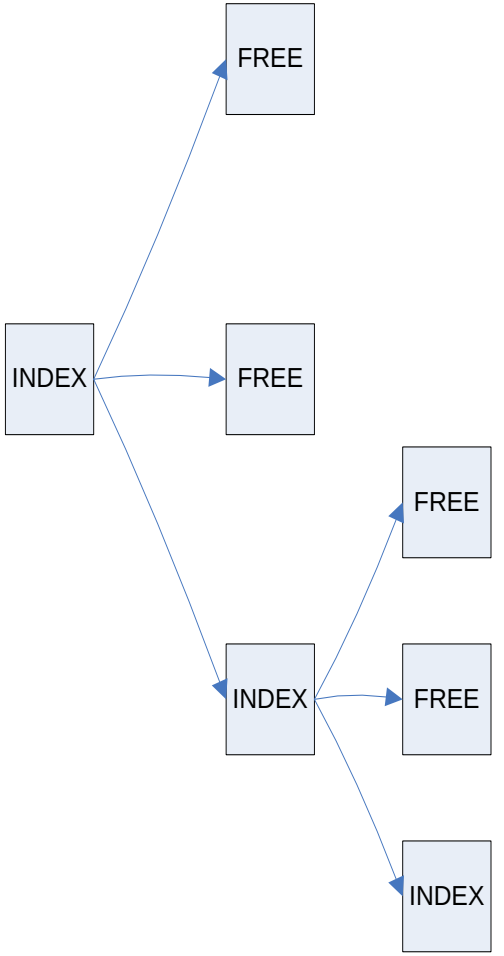
→ $n = 2^{30}/2^{12} = 2^{18}$ bits (or 32K bytes)

  - HĐH Macintosh.

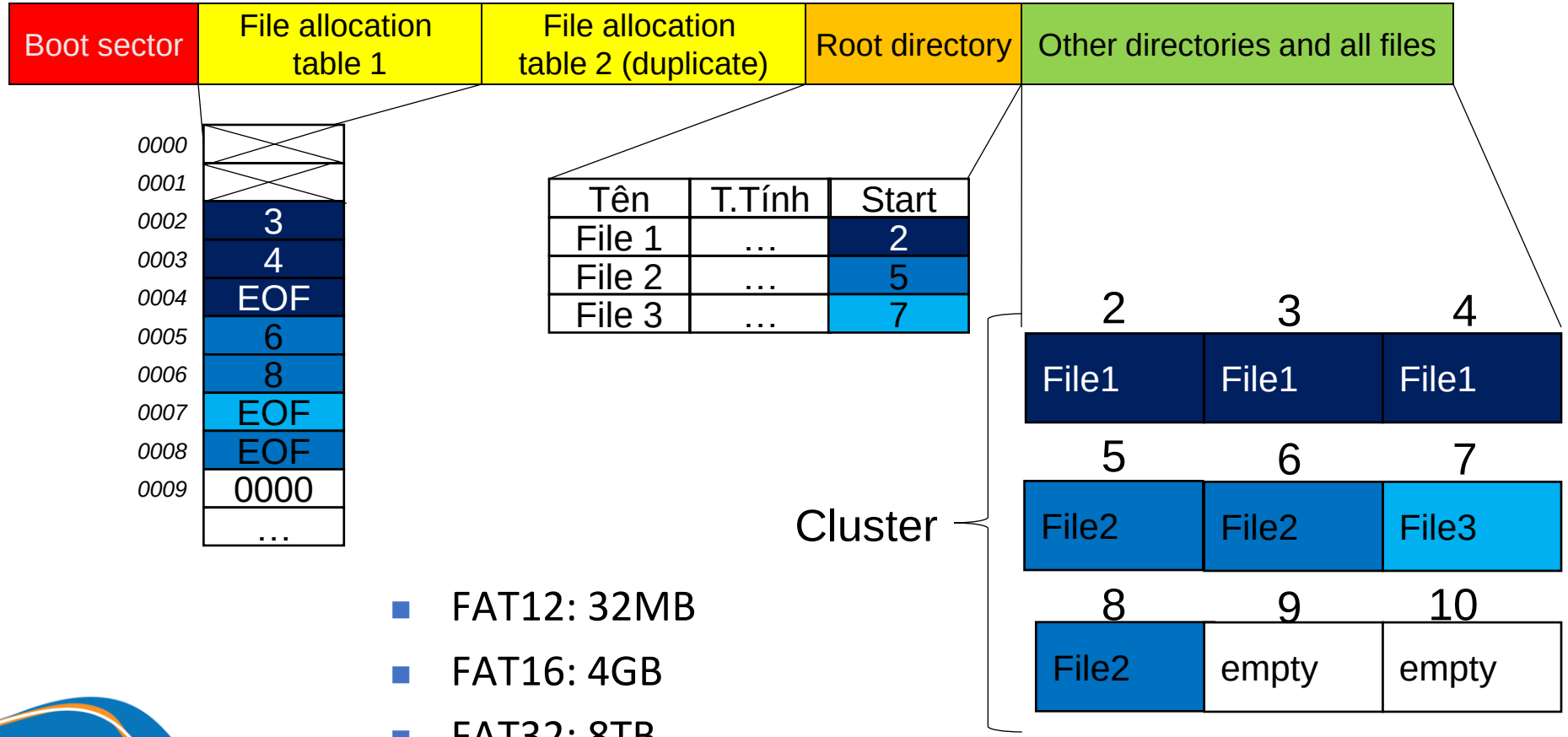
Quản lý không gian đĩa trống

- Danh sách liên kết:
  - Chi phí duyệt danh sách cao.
  - Không tốn không gian đĩa.
- Grouping:
  - Chứa danh sách các block trống.
  - Dễ tìm một lượng lớn các block trống.
- Counting:
  - Chứa địa chỉ block trống đầu tiên và số lượng các block trống liên tục tiếp theo.





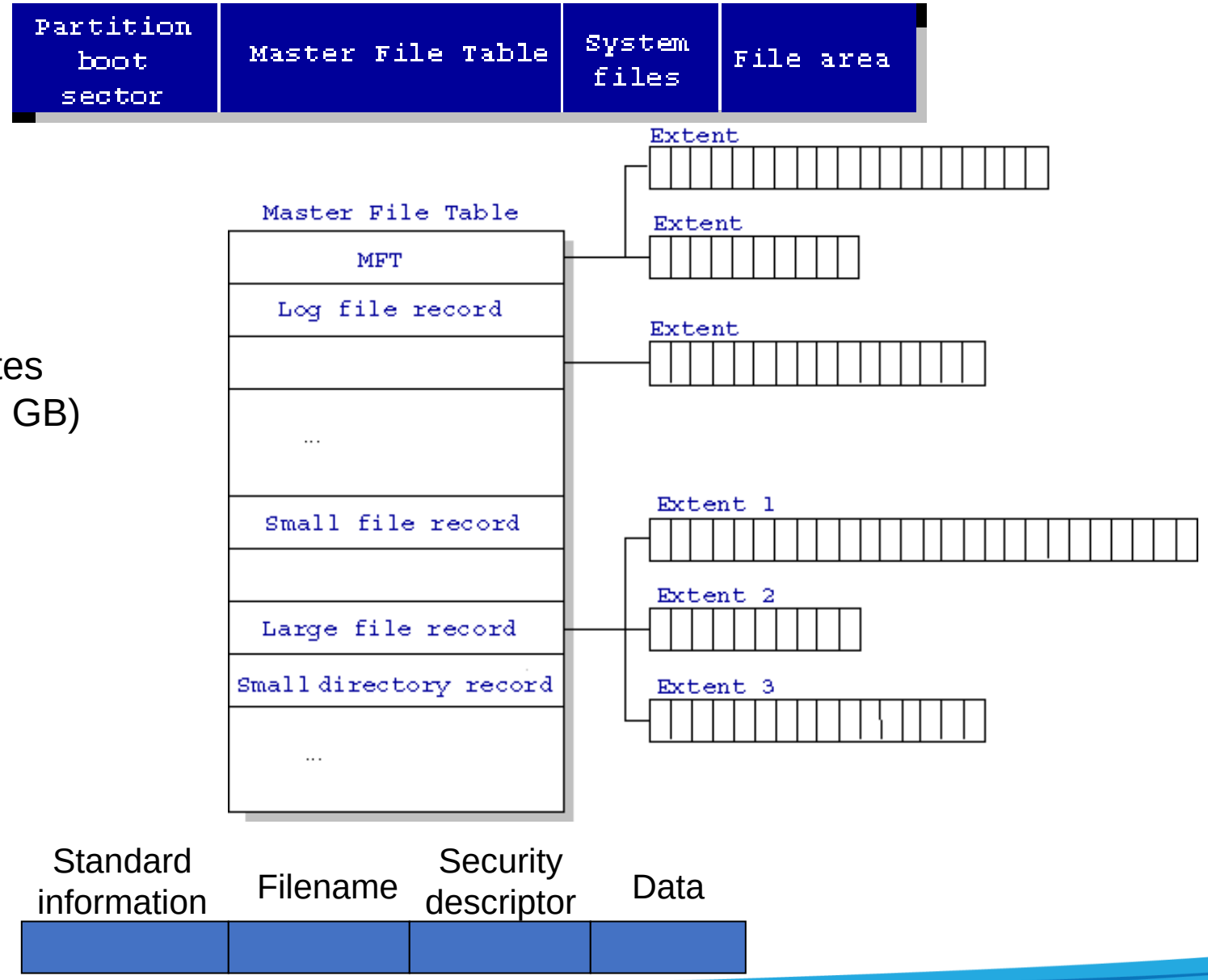
Hệ thống tập tin FAT (12, 16, 32) – File Allocation Table



- FAT12: 32MB
- FAT16: 4GB
- FAT32: 8TB

NTFS (New Technology File System)

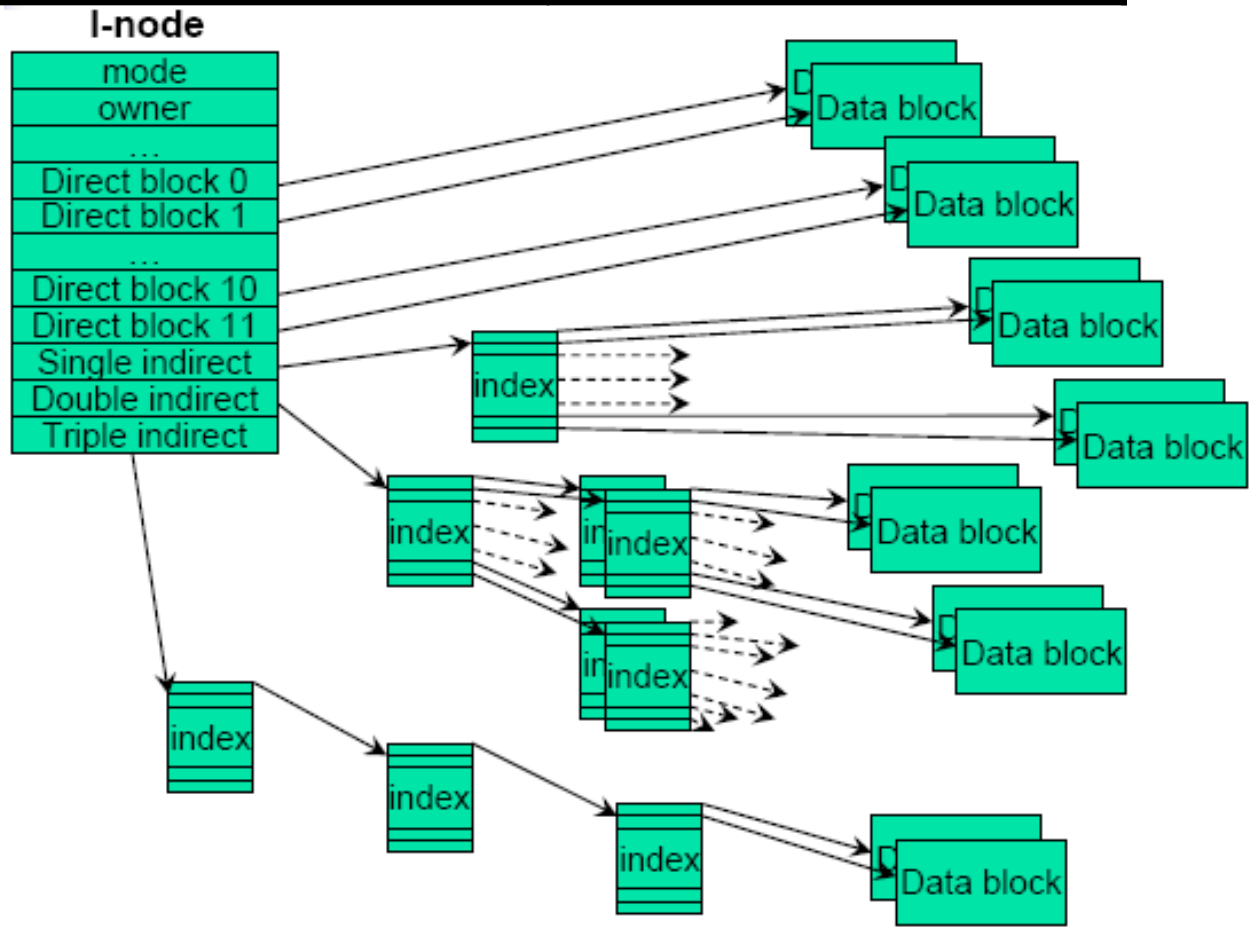
16 exabytes
(16 billion GB)



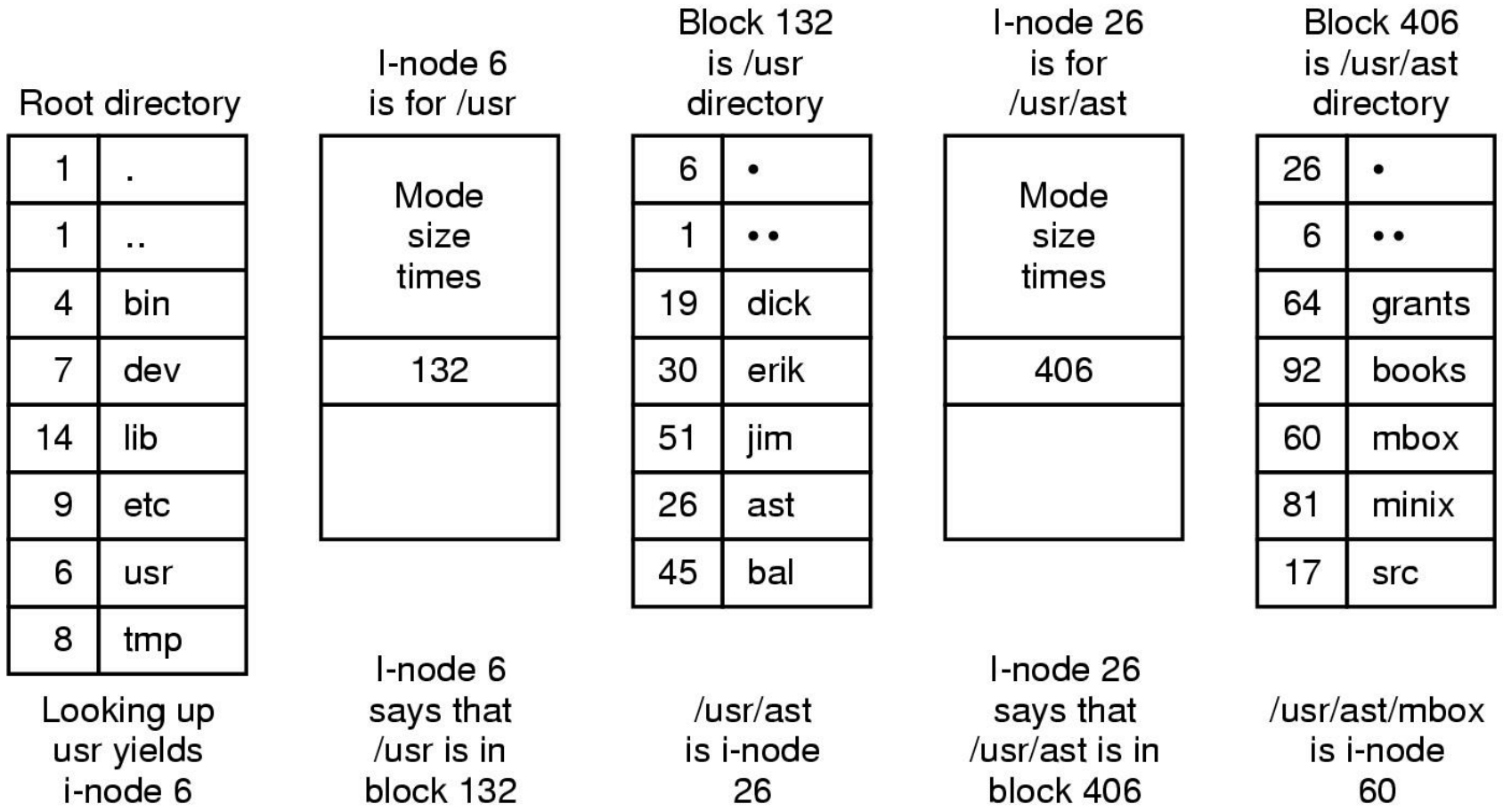
Hệ thống tập tin trên Unix /Linux – Cấu trúc I-node

boot block	super block	I-node	files and directories

- Gián tiếp cấp 1: cấp này trỏ tới 256 địa chỉ. Tổng 256KB
- Gián tiếp cấp 2: 256*256 = 65 MB
- Gián tiếp cấp 3: 256*256*256 = 16GB



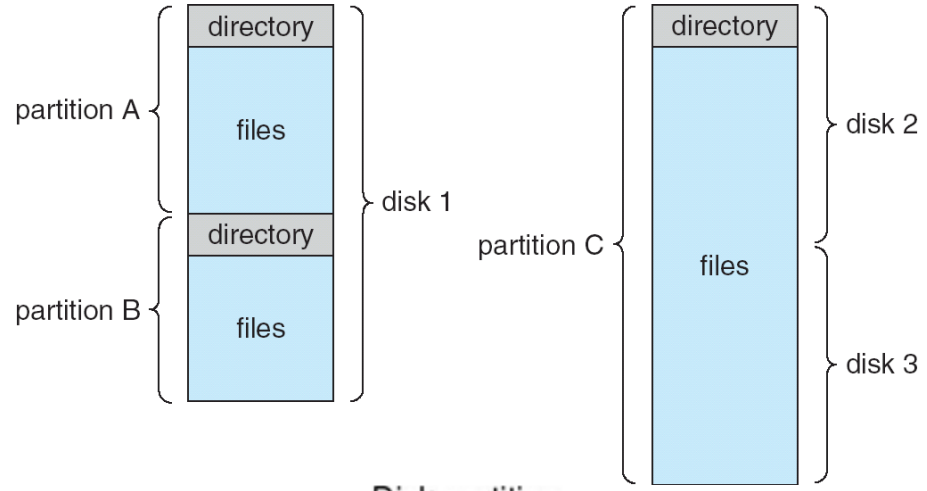
Hệ thống tập tin trên UNIX V7

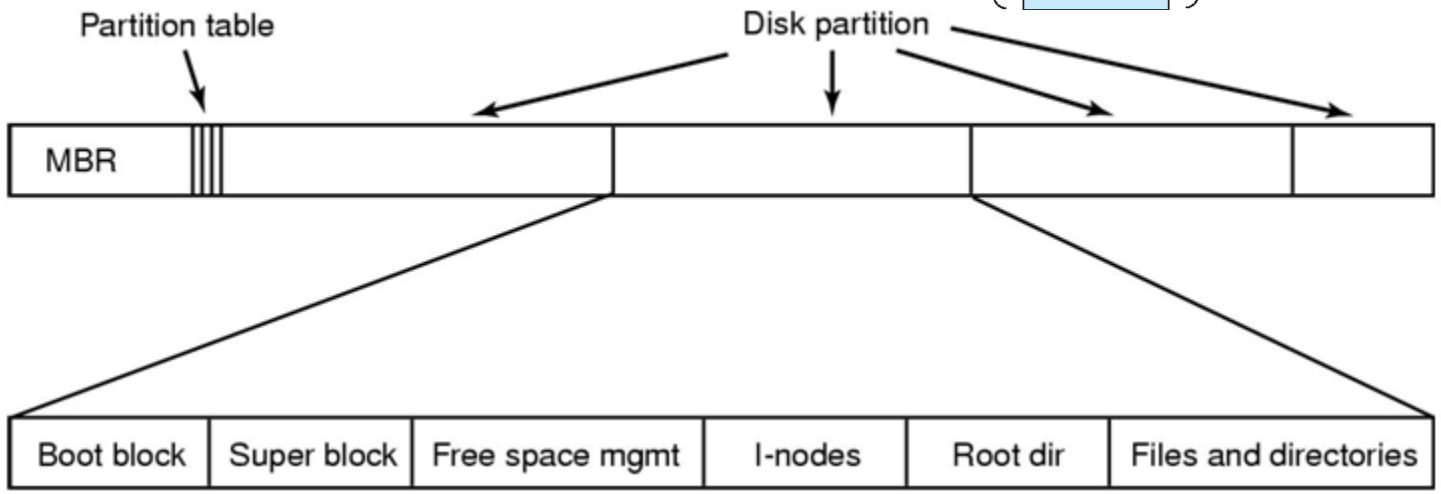


Tổ chức hệ thống tập tin trên đĩa từ

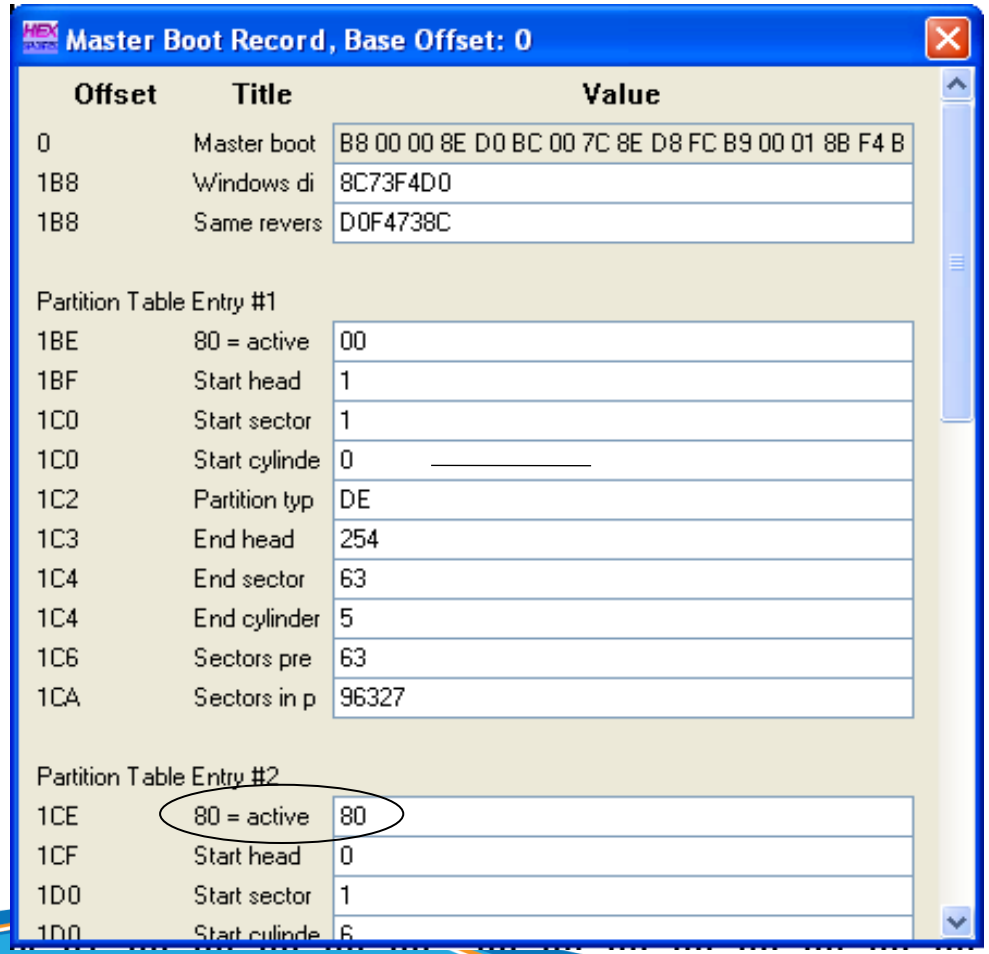
- Master Boot Record (MBR): thường nằm tại sector logic 0, kích thước 512 bytes.
- Phân vùng (Partition):
  - Primary.
  - Extended.

  Tối đa 4 phân vùng.
- Boot block + Super block (Boot sector).
  - Chứa các thông số quan trọng của phân vùng.
  - Chứa một đoạn chương trình nhỏ để nạp HĐH khi khởi động máy.





Master Boot Record



- Đoạn chương trình để giúp khởi động hệ thống.
- Bảng mô tả thông tin các phân vùng logic.
  - TYPE-ID = 0x07 : NTFS
  - TYPE-ID = 0x83 : Linux
  - TYPE-ID = 0x00 : Không sử dụng.
- Thông tin nhận diện MBR.

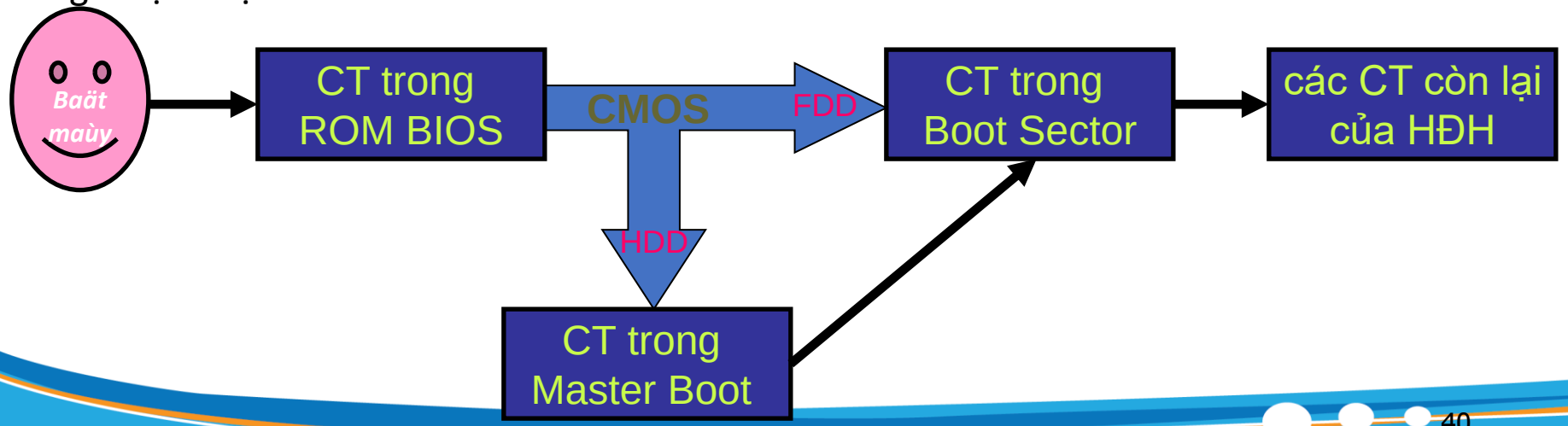
Address		Description		Size in bytes
Hex	Dec			
0000	0	Code Area		max. 446
01B8	440	Optional Disk signature		4
01BC	444	Usually Nulls; 0x0000		2
01BE	446	**Table of primary partitions** (Four 16-byte entries, IBM Partition Table scheme)		64
01FE	510	55h	MBR signature; 0xAA55[1]	2
01FF	511	AAh		
MBR, total size: 446 + 64 + 2 =				512

Quá trình khởi động hệ thống từ đĩa từ

1. POST (Power-On Self-Test)
2. Tải MBR để đọc thông tin bảng phân vùng.

   Tìm phân vùng “active”.

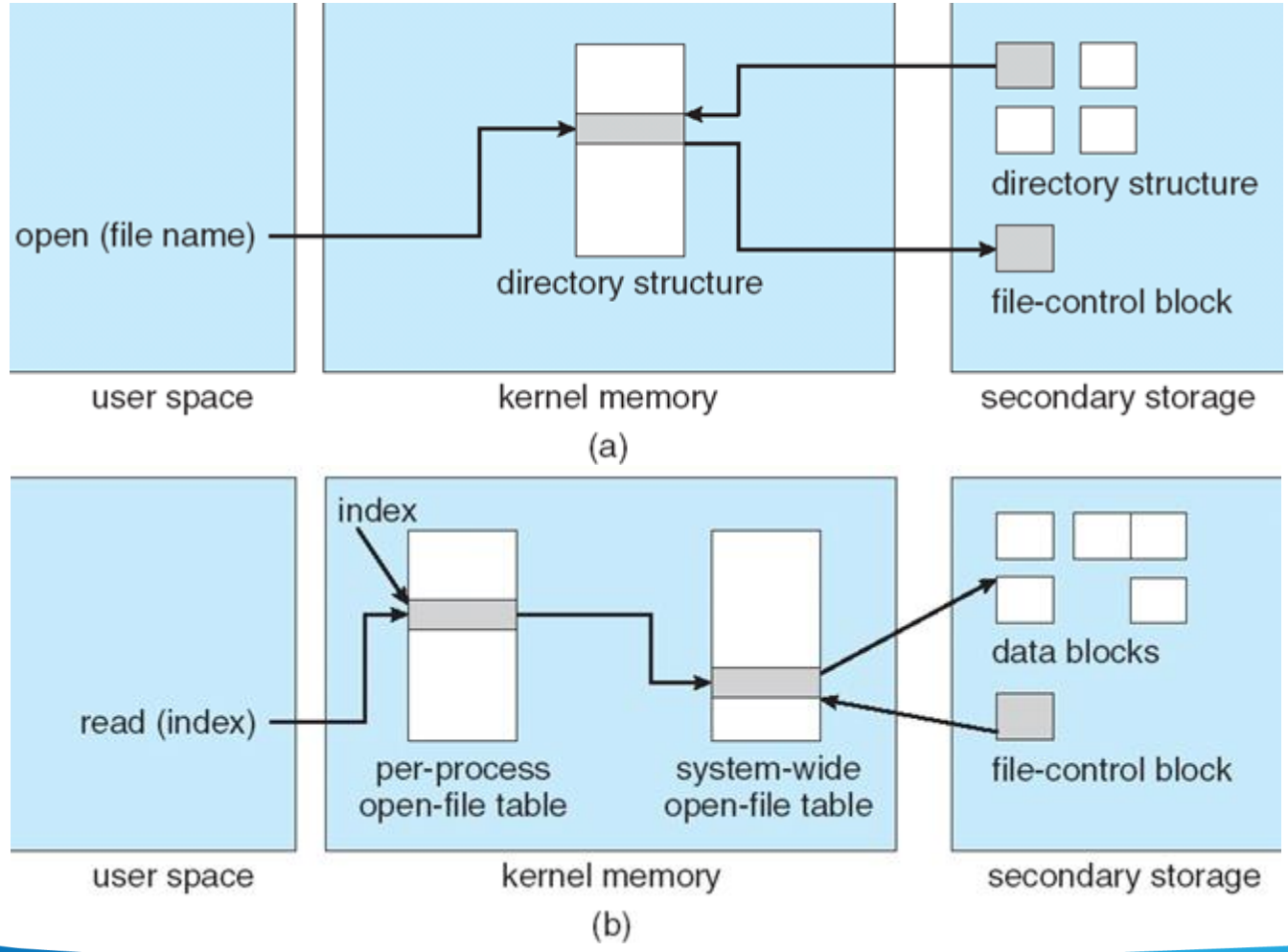
   Nếu không tìm thấy phân vùng “active”, MBR có thể tải một boot loader và chuyển điều khiển cho nó. Boot loader này sẽ cho phép chọn HĐH trên một phân vùng
3. Chuyển quyền điều khiển về cho đoạn mã chương trình nằm trong Boot Sector của phân vùng được chọn
4. Tải HĐH tại phân vùng được chọn



Tổ chức hệ thống tập tin trong bộ nhớ chính

- Vấn đề:
  - Thao tác với nhiều tập tin tại một thời điểm ?
  - Thao tác trên cùng một tập tin tại một thời điểm ?
- Các thông tin cần lưu trữ trong bộ nhớ:
  - Mounted Volume Table – Danh sách các volume được sử dụng trên hệ thống.
  - Directory Structure – Thông tin các thư mục mới được sử dụng
    - Con trỏ trỏ tới volume tương ứng.
  - System-wide open-file Table – Danh sách các tập tin đang được mở trên hệ thống.
    - Con trỏ tập tin, định vị tập tin trên đĩa.
    - Quyền truy cập.
    - Biến đếm tập tin đang mở.
  - Per-process open-file Table – Danh sách các tập tin mà tiến trình đang thao tác.
    - Con trỏ trỏ tới tập tin đang mở tương ứng trong system-wide open-file table.

Tổ chức hệ thống tập tin trong bộ nhớ chính



Kết buộc (Mount) hệ thống tập tin

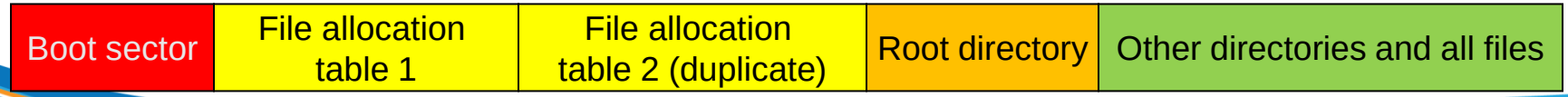
- Một hệ thống tập tin phải được kết buộc (mount) trước khi có thể truy xuất (giống như tập tin phải được mở trước khi sử dụng).
- Các HĐH thường phát hiện và tự động kết buộc các hệ thống tập tin tồn tại trên hệ thống.
  - Windows kết buộc hệ thống tập tin vào ổ đĩa.
  - Linux kết buộc hệ thống tập tin vào một thư mục.
- Một số HĐH cung cấp lệnh để thực hiện việc kết buộc hệ thống tập tin.
  - Ví dụ: lệnh *mount* (Linux).

Hệ thống tập tin FAT

- Giới thiệu hệ thống tập tin FAT
- Vùng Boot Sector
- Bảng thư mục gốc (RDET)
- Bảng FAT
- Vùng dữ liệu
- Bảng thư mục con
- Các thao thác

Giới thiệu hệ thống tập tin FAT

- FAT là hệ thống tập tin được sử dụng trên HĐH MS-DOS và Windows 9x (trên Windows họ NT có thêm hệ thống NTFS).
- Có 3 loại FAT:
  - FAT12.
  - FAT16.
  - FAT32.
- Tổ chức thành 2 vùng:
  - Vùng hệ thống:
    - Vùng Boot Sector.
    - Bảng FAT.
    - Bảng thư mục gốc (có thể nằm trên vùng dữ liệu).
  - Vùng dữ liệu



Vùng Boot Sector

- Gồm một số sector đầu tiên của phân vùng (partition), trong đó:
  - Sector đầu tiên (Boot Sector):
    - Chứa các thông số quan trọng của phân vùng.
    - Chứa một đoạn chương trình nhỏ để nạp HĐH khi khởi động máy.
  - Các sector còn lại (nếu có):
    - Chứa các thông tin hỗ trợ cho việc xác định tổng số cluster trống & tìm kiếm cluster trống được hiệu quả.
    - Chứa một sector bản sao của Boot sector.

fit@hcmus

Boot Sector của FAT12 và FAT16

Offset (hex)	Số byte	Ý nghĩa
0	3	Lệnh nhảy đến đầu đoạn mã Boot (qua khỏi vùng thông số)
3	8	Tên công ty /version của HĐH
B	2	Số byte của sector, thường là 512
D	**1**	**Số sector của cluster (S_C)**
E	**2**	**Số sector trước bảng FAT (S_B)**
10	**1**	**Số lượng bảng FAT (N_F)**, thường là 2
11	**2**	**Số Entry của RDET (S_R)**, thường là 512 với FAT16
13	**2**	**Số sector của volume (S_V)**, bằng 0 nếu $S_V > 65535$
15	1	Kí hiệu loại volume
16	**2**	**Số sector của FAT (S_F)**
18	2	Số sector của track
1A	2	Số lượng đầu đọc (side)
1C	4	Khoảng cách từ nơi mô tả vol đến đầu vol
20	**4**	**Kích thước volume (nếu số 2 byte tại offset 13h là 0)**
24	1	Ký hiệu vật lý của đĩa chứa vol (0 : mềm, 80h: cứng)
25	1	Dành riêng
26	1	Ký hiệu nhận diện HĐH
27	4	SerialNumber của Volume
2B	B	Volume Label
36	8	Loại FAT, là chuỗi "FAT12" hoặc "FAT16"
3E	1CF	Đoạn chương trình Boot nạp tiếp HĐH khi khởi động máy
1FE	2	Dấu hiệu kết thúc BootSector /Master Boot (luôn là AA55h)

Boot Sector của FAT32

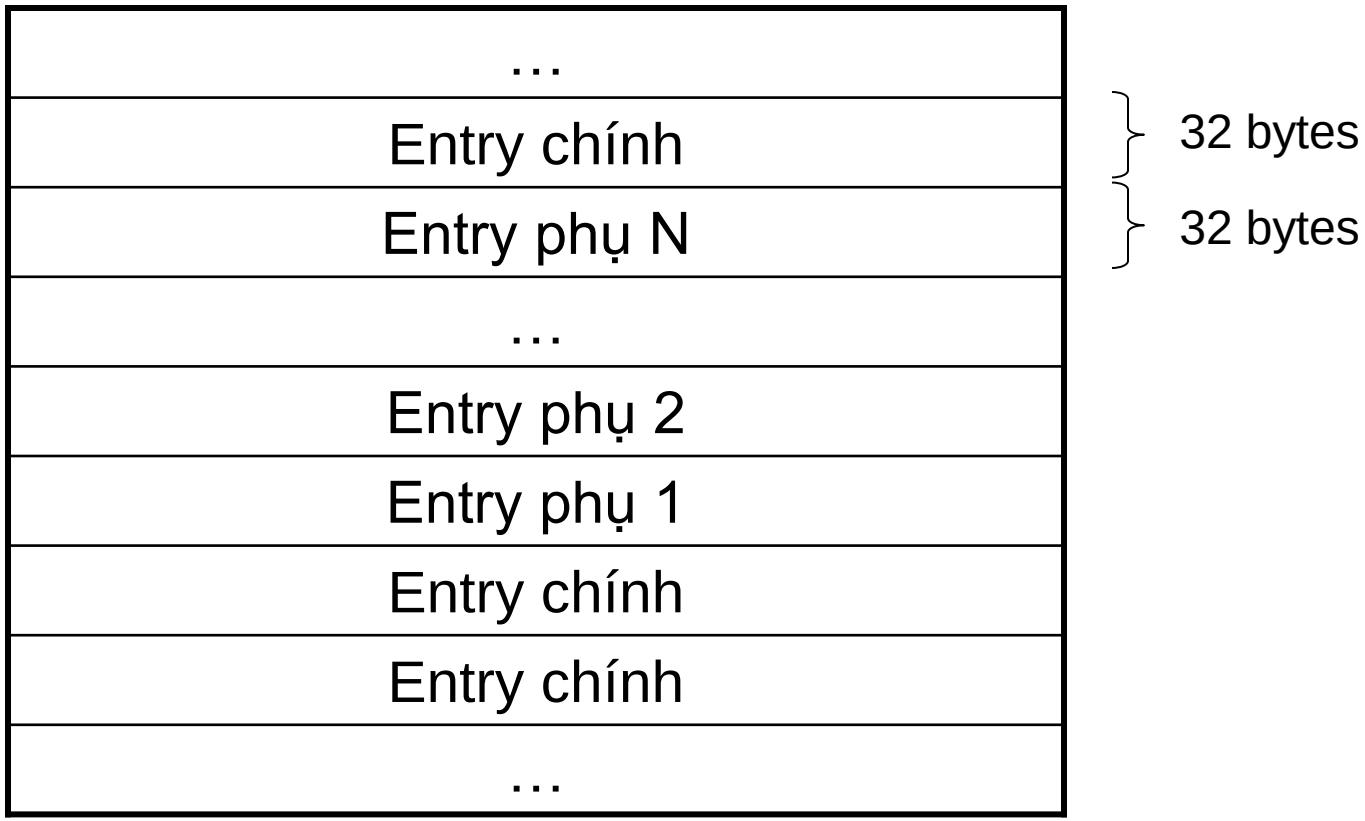
Offset	Số byte	Nội dung
0	3	Jump_Code: lệnh nhảy qua vùng thông số (như FAT)
3	8	OEM_ID: nơi sản xuất – version, thường là "MSWIN4.1"
B	2	Số byte trên Sector, thường là 512 (như FAT)
D	**1**	**S_C: số sector trên cluster (như FAT)**
E	**2**	**S_B: số sector thuộc vùng Bootsector (như FAT)**
10	**1**	**N_F: số bảng FAT, thường là 2 (như FAT)**
11	2	Không dùng, thường là 0 (số entry của RDET – với FAT)
13	2	Không dùng, thường là 0 (số sector của vol – với FAT)
15	1	Loại thiết bị (F8h nếu là đĩa cứng - như FAT)
16	2	Không dùng, thường là 0 (số sector của bảng FAT – với FAT)
18	2	Số sector của track (như FAT)
1A	2	Số lượng đầu đọc (như FAT)
1C	4	Khoảng cách từ nơi mô tả vol đến đầu vol (như FAT)
20	**4**	**S_V: Kích thước volume** (như FAT)
24	**4**	**S_F: Kích thước mỗi bảng FAT**
28	2	bit 8 bật: chỉ ghi vào bảng FAT active (có chỉ số là 4 bit đầu)
2A	2	Version của FAT32 trên vol này
2C	**4**	**Cluster bắt đầu của RDET**
30	**2**	**Sector chứa thông tin phụ (về cluster trống), thường là 1**
32	**2**	**Sector chứa bản lưu của Boot Sector**
34	C	Dành riêng (cho các phiên bản sau)
40	1	Kí hiệu vật lý của đĩa chứa vol (0 : mềm, 80h: cứng)
41	1	Dành riêng
42	1	Kí hiệu nhận diện HĐH
43	4	SerialNumber của Volume
47	B	Volume Label
52	**8**	**Loại FAT, là chuỗi "FAT32"**
5A	1A4	Đoạn chương trình khởi tạo & nạp HĐH khi khởi động máy
1FE	2	Dấu hiệu kết thúc BootSector /Master Boot (luôn là AA55h)

Bảng thư mục gốc (RDET – Root Directory Entry Table)

- Nằm trên vùng hệ thống (FAT12 & FAT16) hoặc nằm trên vùng dữ liệu (FAT32).
- Gồm một dãy các phần tử (gọi là entry), mỗi phần tử có kích thước 32 bytes chứa các thông tin của 1 tập tin hoặc một thư mục.

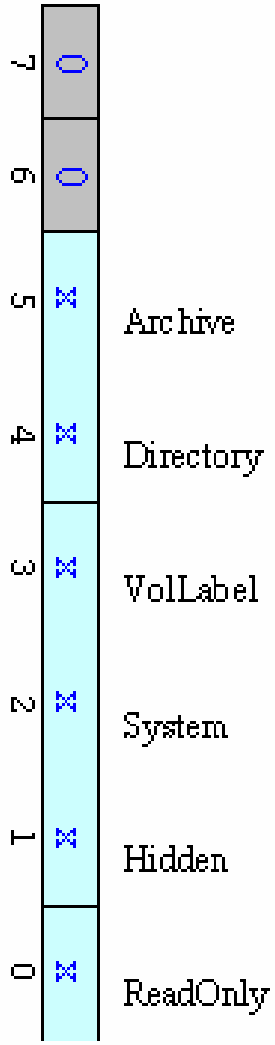
- Thông tin của mỗi tập tin/ thư mục có thể chiếm 1 hay nhiều entry.
- Byte đầu tiên của mỗi entry cho biết trạng thái của entry này.
  - 0 – entry trống.
  - E5h – tập tin chiếm entry này đã bị xóa.
  - Giá trị khác – đang chứa thông tin của tập tin/ thư mục.
- Có 2 loại entry:
  - Entry chính: chứa các thông tin của tập tin.
  - Entry phụ: chỉ chứa tên của tập tin.

Cấu trúc bảng thư mục gốc



Entry chính

Offset (hex)	Số byte	Ý nghĩa
0	8	Tên chính /tên ngắn - lưu bằng mã ASCII
8	3	Tên mở rộng – mã ASCII
B	1	Thuộc tính trạng thái (0.0.A.D.V.S.H.R)
C	1	Dành riêng
D	3	Giờ tạo (miligiây:7; giây:6; phút:6; giờ:5)
10	2	Ngày tạo (ngày: 5; tháng: 4; năm-1980: 7)
12	2	Ngày truy cập gần nhất (lưu như trên)
14	2	Cluster bắt đầu – phần Word (2Byte) cao
16	2	Giờ sửa gần nhất (giây/2:5; phút:6; giờ:5)
18	2	Ngày cập nhật gần nhất (lưu như trên)
1A	2	Cluster bắt đầu – phần Word thấp
1C	4	Kích thước của phần nội dung tập tin



Entry phụ

Offset	Số byte	Ý nghĩa
0	1	Thứ tự của entry (bắt đầu từ 1)
1	A (10d)	5 ký tự UniCode – bảng mã UTF16
B (11d)	1	Dấu hiệu nhận biết (luôn là **0F**h)
E (14d)	C (12d)	6 ký tự kế tiếp
1C (28d)	4	2 ký tự kế tiếp

Bảng FAT (1/2)

- Nằm trên vùng hệ thống.
- Thường có 2 bảng: 1 bảng chính và 1 bảng dự phòng.
- Lưu vị trí của các tập tin/ thư mục theo kiểu danh sách liên kết.

Giá trị	X	X	3	4	EOF	7	EOF	6
Phần tử	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*

- Kích thước mỗi phần tử FAT phụ thuộc vào loại FAT.
  - FAT12: kích thước mỗi phần tử là 12 bits ~ 1.5 bytes.
  - FAT16: kích thước mỗi phần tử là 16 bits ~ 2 bytes.
  - FAT32: kích thước mỗi phần tử là 32 bits ~ 4 bytes.

Bảng FAT (2/2)

- Phần tử thứ k trên bảng FAT (đánh số từ 0) cho biết trạng thái của cluster thứ k trên vùng dữ liệu (đánh số từ 2) → 2 phần tử đầu của bảng FAT không dung.

Trạng thái của cluster k trên vùng dữ liệu	Giá trị của phần tử k trên bảng FAT			Ghi chú
	FAT12	FAT16	FAT32	
Trống	0	0	0	= FREE
Hư	FF7	FFF7	0FFFFFF7	= BAD
Cluster cuối của file	FFF	FFFF	0FFFFFFF	= EOF
Chứa nội dung file	2 .. FEF	2 .. FFEF	2..0FFFFFEF	

- FAT 12 quản lý được tối đa 4078 (FEEh) cluster.
- FAT 16 quản lý được tối đa 65518 (FFEEh) cluster.
- Nếu số cluster quá 65518 thì dùng FAT 32.

Làm sao để truy xuất các phần tử FAT ? (1/2)

- Lưu trữ bảng FAT là dãy byte.

Giá trị	**F0**	**FF**	**FF**	**03**	**40**	**00**	**FF**	**7F**	**FF**	**AB**	**CD**	**EF**
Byte	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*

- Truy xuất theo FAT 32 (mỗi phần tử 4 bytes).

Giá trị	**F0**	**FF**	**FF**	**03**	**40**	**00**	**FF**	**7F**	**FF**	**AB**	**CD**	**EF**
Byte	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*

Giá trị	**03 FF FF F0**	**7F FF 00 40**	**EF CD AB FF**
Ptử FAT	0	1	2

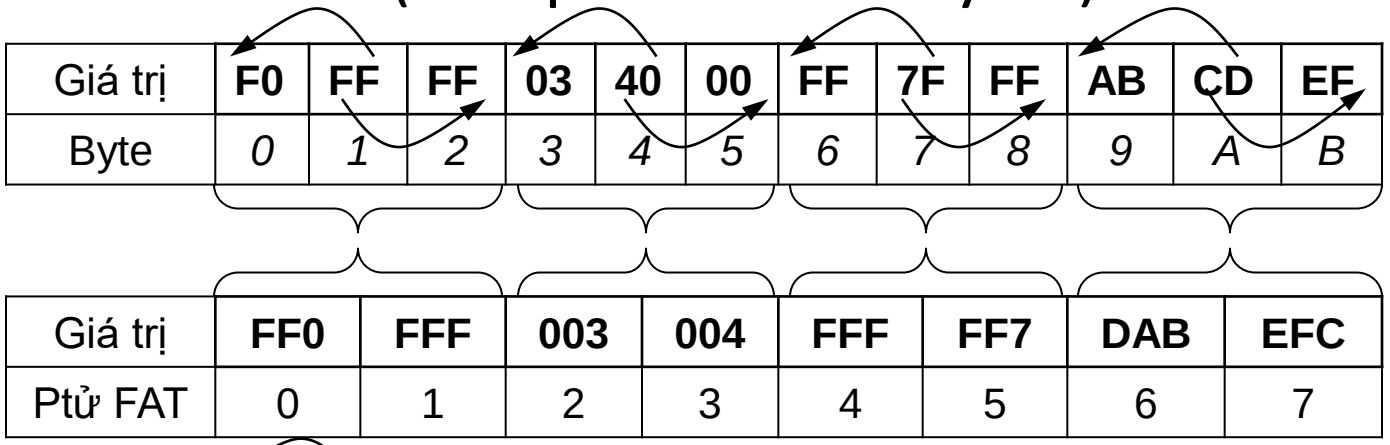
- Truy xuất theo FAT 16 (mỗi phần tử 2 bytes).

Giá trị	**F0**	**FF**	**FF**	**03**	**40**	**00**	**FF**	**7F**	**FF**	**AB**	**CD**	**EF**
Byte	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*

Giá trị	**FFF0**	**03FF**	**0040**	**7FFF**	**ABFF**	**EFCD**
Ptử FAT	0	1	2	3	4	5

Làm sao để truy xuất các phần tử FAT ? (2/2)

- Truy xuất theo FAT 12 (mỗi phần tử 1.5 bytes).



Giá trị	F0	FF	FF	03	40	00	FF	7F	FF	AB	CD	EF
Byte	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*

Giá trị	FF0	FFF	003	004	FFF	FF7	DAB	EFC
Ptử FAT	0	1	2	3	4	5	6	7

  - Phần tử chẵn: **F0 FF → FF0**
  - Phần tử lẻ: **FF FF → FFF**

- Công thức tương quan giữa phần tử thứ k và byte thứ i trên bảng FAT:

  i = k * <kích thước phần tử FAT>

Vùng dữ liệu

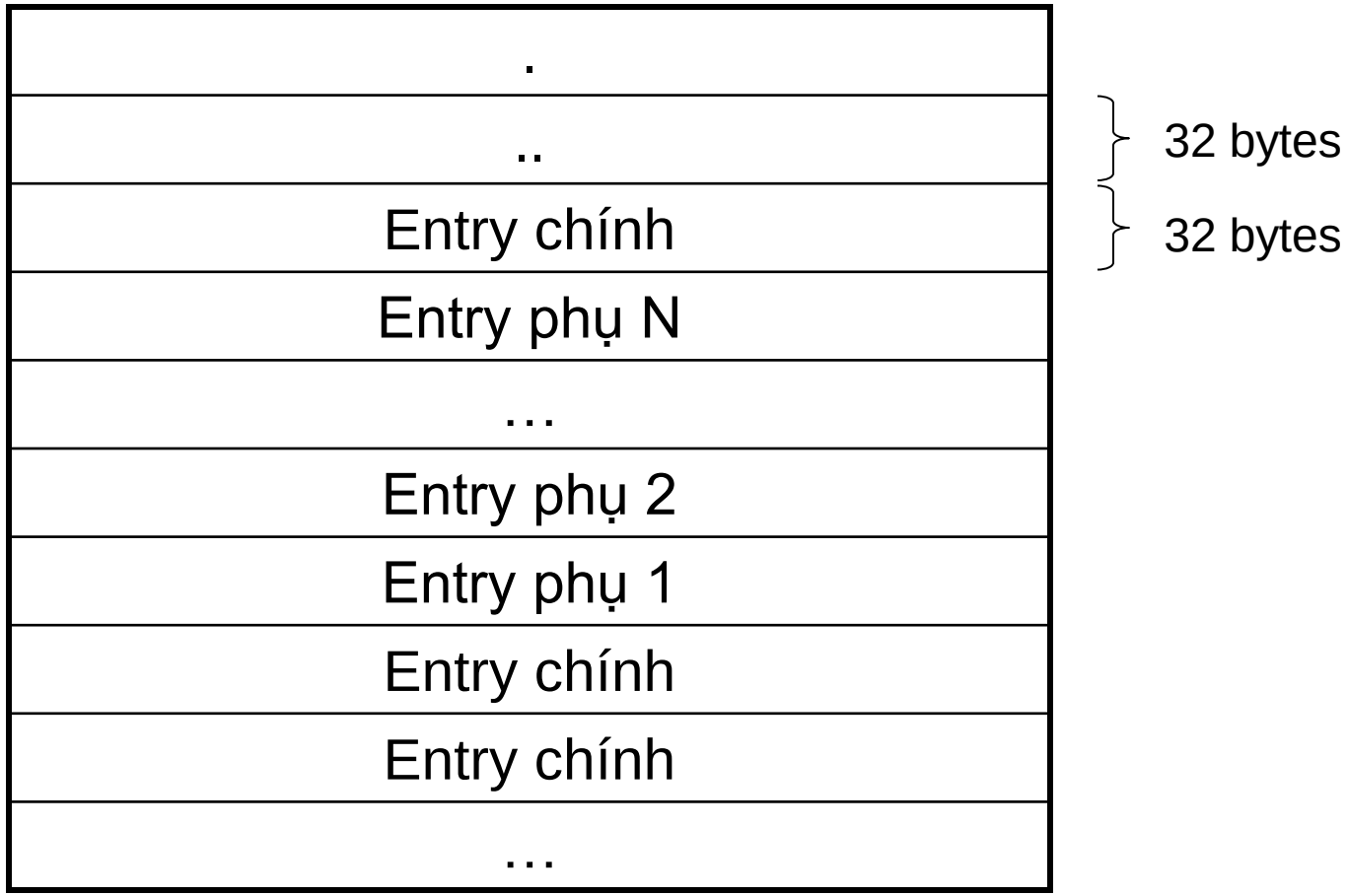
- Mỗi phần tử trên vùng dữ liệu, gọi là ***cluster***, có kích thước 2^n sector, tùy thuộc vào người dùng khi format.
- Cluster trên vùng dữ liệu đánh số từ 2.
- Công thức tương quan giữa cluster thứ k trên vùng dữ liệu và sector thứ i trên phân vùng.

$$i = S_B + S_F * N_F + [S_{RDET}] + (k - 2) * S_c$$

Bảng thư mục con (SDET – Sub Directory Entry Table)

- Chứa thông tin các tập tin/ thư mục con của một thư mục.
- Nằm trên vùng dữ liệu, có cấu trúc hoàn toàn giống bảng thư mục gốc.
- Mỗi SDET luôn có 2 entry ‘.‘ và ‘..‘ ở đầu bảng mô tả về chính thư mục này và thư mục cha của nó.

Cấu trúc bảng thư mục con



Một số thao tác trên hệ thống tập tin FAT (1/10)

- Đọc nội dung tập tin (TYPE):
  - Xác định entry chính trong bảng thư mục (RDET/ SDET) chứa thông tin của tập tin dựa vào phần tên và phần mở rộng (lưu ý trường hợp tên dài).
  - Từ entry chính tìm được, ta có được chỉ số cluster/ phần tử FAT đầu tiên.
  - Từ phần tử FAT đầu tiên này, vào bảng FAT, xác định các phần tử còn lại của tập tin, tương ứng có được các cluster của tập tin này → các sector của tập tin.
  - Đọc các sector nội dung của tập tin.

Một số thao tác trên hệ thống tập tin FAT (2/10)

- Liệt kê nội dung thư mục (DIR):
  - Xác định entry chính trong bảng thư mục (RDET/ SDET) chứa thông tin của thư mục dựa vào phần tên (lưu ý trường hợp tên dài).
  - Từ entry chính tìm được, ta có được chỉ số cluster/ phần tử FAT đầu tiên.
  - Từ phần tử FAT đầu tiên này, vào bảng FAT, xác định các phần tử còn lại của tập tin, tương ứng có được các cluster của tập tin này → các sector của tập tin.
  - Đọc các sector nội dung tìm được theo từng entry (32 bytes) và hiển thị thông tin của các tập tin và thư mục con của thư mục này.

Một số thao tác trên hệ thống tập tin FAT (3/10)

- Tạo tập tin (COPY CON):
  - Tìm đủ số entry trống liên tiếp nhau trên bảng thư mục (RDET/ SDET) để chứa thông tin của tập tin (lưu ý trường hợp tên dài).
  - Kiểm tra trên bảng FAT xem còn đủ số cluster trống để chứa nội dung của tập tin không.
  - Lưu thông tin của tập tin vào các entry trống tìm được.
  - Ghi giá trị vào các phần tử FAT trống tìm được theo dạng danh sách liên kết, đồng thời lưu nội dung tập tin vào các cluster tương ứng (theo chỉ số sector).

Một số thao tác trên hệ thống tập tin FAT (4/10)

- Tạo thư mục (MD):
  - Tìm đủ số entry trống liên tiếp nhau trên bảng thư mục (RDET/ SDET) để chứa thông tin của thư mục (lưu ý trường hợp tên dài).
  - Kiểm tra trên bảng FAT xem còn cluster trống nào để chứa nội dung của thư mục không.
  - Lưu thông tin của thư mục vào các entry trống tìm được.
  - Ghi giá trị kết thúc vào phần tử FAT trống tìm được, đồng thời tạo 2 thư mục “.” và “..” chiếm 2 entry đầu tiên trong cluster tương ứng.

Một số thao tác trên hệ thống tập tin FAT (5/10)

- Xóa tập tin (DELETE):
  - Xác định entry chính trong bảng thư mục (RDET/ SDET) chứa thông tin của tập tin dựa vào phần tên và phần mở rộng (lưu ý trường hợp tên dài).
  - Đặt giá trị E5h vào byte đầu tiên của entry chính và tất cả các entry phụ của tập tin (nếu có).
  - Từ entry chính tìm được, ta có được chỉ số cluster/ phần tử FAT đầu tiên. Vào bảng FAT, xác định được các phần tử còn lại của tập tin.
  - Đặt tất cả các phần tử FAT của tập tin về giá trị 0.
  - Lưu ý, hoàn toàn không thay đổi gì phần nội dung của tập tin.

Một số thao tác trên hệ thống tập tin FAT (6/10)

- Xóa thư mục (RD):
  - Thực hiện xóa đệ qui tất cả các tập tin và thư mục con từ cấp sâu nhất ra. Xóa thư mục rỗng tương tự như xóa tập tin.

Một số thao tác trên hệ thống tập tin FAT (7/10)

- Sao chép tập tin (COPY):
  - Tìm đủ số entry trống liên tiếp nhau trên bảng thư mục (RDET/ SDET) để chứa thông tin của tập tin đích (lưu ý trường hợp tên dài).
  - Kiểm tra trên bảng FAT xem còn đủ số cluster trống để chứa nội dung của tập tin đích không.
  - Copy thông tin (các entry) của tập tin nguồn sang các entry tìm được của tập tin đích.
  - Ghi giá trị vào các phần tử FAT trống tìm được theo dạng danh sách liên kết, đồng thời copy các sector nội dung tập tin nguồn vào các sector nội dung tương ứng tìm được của tập tin đích.

Một số thao tác trên hệ thống tập tin FAT (8/10)

- Di chuyển tập tin (MOVE):
  - Tìm đủ số entry trống liên tiếp nhau trên bảng thư mục (RDET/ SDET) để chứa thông tin của tập tin đích (lưu ý trường hợp tên dài).
  - Copy thông tin (các entry) của tập tin nguồn sang các entry tìm được của tập tin đích
  - Xóa thông tin của tập tin nguồn.

Một số thao tác trên hệ thống tập tin FAT (9/10)

- Đổi tên tập tin/ thư mục (REN):
  - Xác định entry chính trong bảng thư mục (RDET/ SDET) chứa thông tin của tập tin/ thư mục dựa vào phần tên và phần mở rộng (lưu ý trường hợp tên dài).
  - Nếu tên tập tin không cần thêm các entry phụ.
    - Cập nhật lại phần tên và phần mở rộng.
  - Nếu tên tập tin cần thêm các entry phụ.
    - Tìm đủ số entry trống liên tiếp nhau trên bảng thư mục (RDET/ SDET) để chứa thông tin của tập tin đích (lưu ý trường hợp tên dài).
    - Copy thông tin (các entry) của tập tin nguồn sang các entry tìm được của tập tin đích.

Một số thao tác trên hệ thống tập tin FAT (10/10)

- Quick format:
  - Giữ lại các thông số cũ của phân vùng.
  - Cập nhật lại trạng thái các cluster đang chứa dữ liệu thành trống và cho tất cả entry trên bảng thư mục gốc về trạng thái trống.
  - Chức năng này tương đương với việc xóa tất cả mọi tập tin & thư mục đang tồn tại trên phân vùng, nhưng thời gian thi hành rất nhanh, có thể nhanh hơn thời gian xóa một tập tin.
- Full format:
  - Các thông số của từng thành phần trên phân vùng sẽ được xác định lại.
  - Để tạo ra những dạng thức mới phù hợp hơn cho phân vùng. Chức năng này dĩ nhiên cũng được dùng cho những phân vùng chưa được định dạng.

Ví dụ

- Xét đĩa mềm 1.44MB (có 2880 sector), để các tập tin trên vol có thể truy xuất nhanh & an toàn hơn ta có thể cho SC = 4 (sector), SB = 1 (sector), SR = 32 (entry) = 2 (sector), nF = 2.

  Thay các giá trị trên vào đẳng thức SB + nF*SF +SR + SD = SV ta được
  1 + 2SF +2 + SD = 2880 (sector), hay 2SF + SD = 2877 (sector) (*)
  (*) $\Rightarrow$ SD < 2877 (sector) = 719.25 (cluster) (vì SC = 4 sector).
  $\Rightarrow$ Loại FAT tối ưu nhất (về kích thước) là **FAT12**, vì SD **< 4079** (cluster)

  - **Giả sử SF = 1** (sector): (*) $\Rightarrow$ SD = 2875 (sector) = 718.75 (cluster)
  $\Rightarrow$ Vùng dữ liệu có 718 cluster, nên bảng FAT phải có 718 + 2 = 720 phần tử,
  do đó SF = (720*1.5)/512 = 2.1x (sector)
  Bảng FAT phải chiếm 3 sector – mâu thuẫn với giả thiết SF = 1**.**
  Vậy kích thước bảng FAT của vol này không thể là 1 sector
  - **Giả sử SF = 2** (sector): tương tự, ta vẫn thấy mâu thuẫn, tức kích thước bảng FAT phải lớn hơn 2 sector.
  - **Giả sử SF = 3** (sector): (*) $\Rightarrow$ SD = 2871 (sector) = 717.75 (cluster).
  $\Rightarrow$ Vùng dữ liệu có 717 cluster, nên bảng FAT phải có 717 + 2 = 719 phần tử,
  do đó SF = (719*1.5)/512 = 2.1x (sector)
  $\Rightarrow$ Bảng FAT phải chiếm 3 sector – phù hợp với giả thiết SF = 3.
  Vậy kích thước bảng FAT của vol này là 3 sector.

Fragmentation

- Một bảng FAT gọi là bị phân mảnh nếu xảy ra ít nhất một trong 2 điều kiện:
  - Các phần tử FAT của 1 tập tin không liên tiếp nhau.
  - Các phần tử FAT của các tập tin không liên tiếp nhau.

→Truy xuất chậm.

→Defragmentation.