

**ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**-Khoa Công nghệ thông tin-**

****

**BÀI TẬP LỚN**

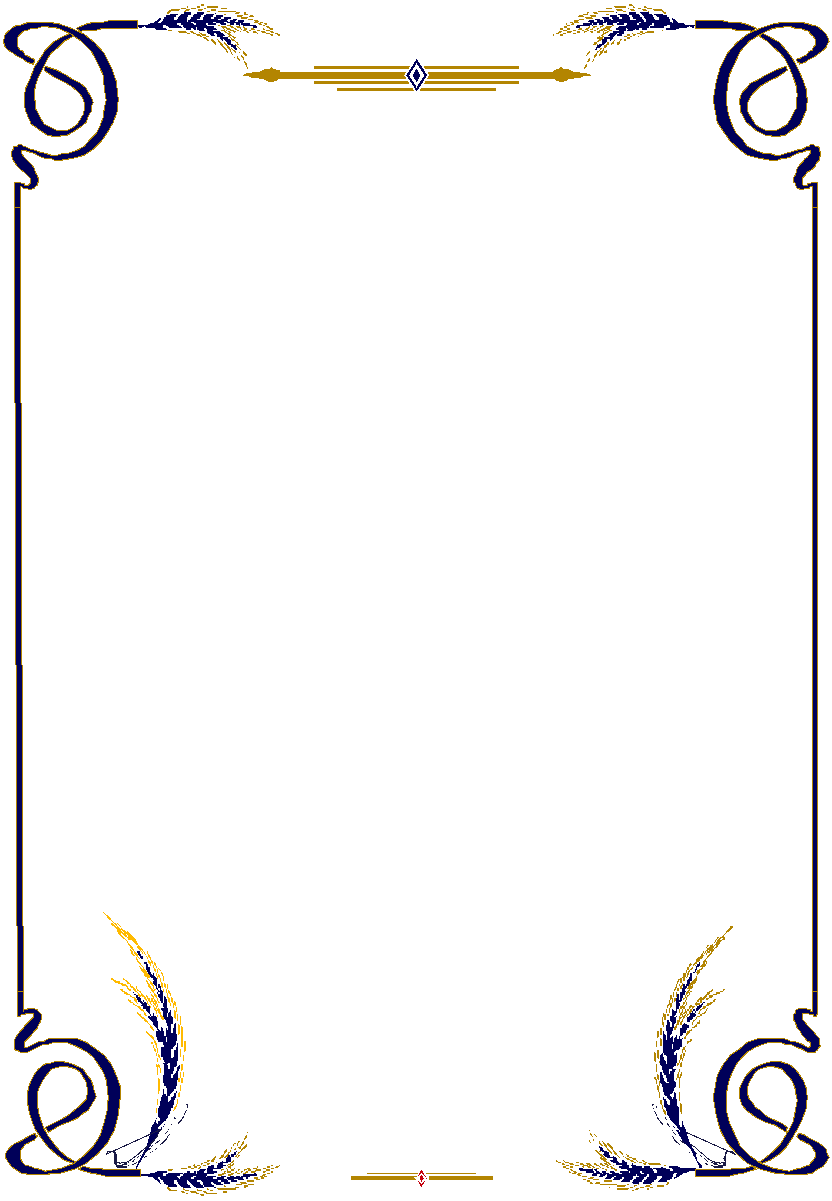
**Môn: NGUYÊN LÝ HỆ ĐIỀU HÀNH**

**Đề tài: Nguyên cứu và tìm hiểu về quản lý bộ nhớ ngoài trong**

**HĐH Windows**

**Giáo viên hướng dẫn: Nguyễn Tuấn Tú**

**Nhóm sinh viên thực hiện: Nhóm 5**



**ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**-Khoa Công nghệ thông tin-**

****

**BÀI TẬP LỚN**

**Môn: NGUYÊN LÝ HỆ ĐIỀU HÀNH**

**Đề tài: Nguyên cứu và tìm hiểu về quản lý bộ nhớ ngoài trong HĐH Windows**

**Giáo viên hướng dẫn: Nguyễn Tuấn Tú**

**Sinh viên thực hiện: Nguyễn Đức An**

**Phạm Minh Chiến**

**Cao Mạnh Cường**

**Nguyễn Sỹ Nam Điền**

**Vũ Trung Đức**

**Mục Lục**

[LỜI NÓI ĐẦU](#_heading=h.1fob9te)

[Chương 1: Tổng quan về bộ nhớ ngoài 1](#_heading=h.3j2qqm3)

1.1.Cấu trúc vật lý.................................................................................................................1

1.1.1.Đĩa từ (Platter).................................................................................................1

1.1.2.Các rảnh từ (Track)..........................................................................................1

1.1.3.Sector...............................................................................................................2

1.1.4.Cylinder...........................................................................................................2

1.1.5.Đầu đọc/ghi (Read Write Heads)....................................................................2

1.1.6.Cần di chuyển đọc/ghi (Disk Controler)..........................................................3

[Chương 2:Các dạng lưu trữ dữ liệu trên hệ điều hành Windows 4](#_heading=h.1y810tw)

2.1.Lưu trữ cơ bản (Basic Stroage) 4

2.2.Lưu trữ động (Dynamic Storage) 5

[2.2.1.Spanned Volume 5](#_heading=h.1ci93xb)

[2.2.2.Simple Volume 6](#_heading=h.1ci93xb)

[2.2.3.Striped Volume (RAID-0) 6](#_heading=h.1ci93xb)

[2.2.4.Mirror Volume(RAID-1) 7](#_heading=h.1ci93xb)

[2.2.5.RAID-5 Volume 8](#_heading=h.1ci93xb)

[Chương 3:Chương trình quản lý bộ nhớ ngoài Disk Manager 8](#_heading=h.1y810tw)

3.1.Xem thuộc tính của đĩa 8

[3.2.Xem thuộc tính của phân vùng hoặc đĩa cục bộ 9](#_heading=h.2xcytpi)

[Chương 4:Quản lý không gian nhớ tự do tỏng hệ điều hành 9](#_heading=h.1y810tw)

4.1.Quản lý bộ nhớ bằng phương pháp liệt kê (free list) 9

4.2.Quản lí bộ nhớ bằng phương pháp lập nhóm (Grouping) 10

4.3.Phương pháp đếm (Counting) 10

[Chương 5:Cấp phát không gian nhớ tự do trong hệ điều hành Windows 1](#_heading=h.1y810tw)2

5.1.Cấp phát kề (Contiguous) 12

5.2.Cấp phát liên kết (Linked) 13

5.3.Cấp phát theo chỉ số (Index) 14

[Chương 6: Lập lịch cho đĩa từ trong hệ điều hành Windows 1](#_heading=h.3rdcrjn)7

6.1.Khái niệm về lập lịch cho đĩa 17

6.2.Các thuật toán lập lịch cho đĩa...................................................................................19

6.2.1.First come first Served(FCFS) 19

6.2.2.Shortest Seek time First 20

6.2.3.Scan 21

6.2.4.C-Scan 21

6.2.5.Look 22

6.2.6.C-Look 23

[6.3.](#_heading=h.44sinio) Quản lý lỗi 24

[6.4.](#_heading=h.2p2csry) RAM Disks 23

[6.5.](#_heading=h.2jxsxqh) Interleave 25

[Kết Luận 2](#_heading=h.z337ya)6

# LỜI NÓI ĐẦU

Ngày nay, với sự phát triển vượt bậc của công nghệ, máy tính đã trở thành một phần không thể thiếu trong cuộc sống của chúng ta. Chúng ta có thể dùng máy tính để học, để làm việc, để giải trí,… Cho dù mục đích có là gì đi chăng nữa thì yêu cầu cấp thiết đối với một chiếc máy tính là phải đủ dung lượng. Do vậy, để hỗ trợ cho bộ nhớ trong của mỗi máy tính, chúng ta có thêm những bộ nhớ ngoài cực kì đa dạng để phục vụ nhu cầu lưu trữ lượng thông tin với dung lượng cực kì lớn.

Vậy bộ nhớ ngoài là gì? Từ những chiếc USB vô cùng quen thuộc với chúng ta đến những thiết bị to lớn, phức tạp hơn, đều có cùng chung cách quản lý. Và để làm rõ vấn đề này chúng em xin trình bày về cách quản lý bộ nhớ ngoài trong hệ điều hành Window ở bài tập lớn dưới đây.

Nhóm sinh viên thực hiện!

**CHƯƠNG 1: TỔNG QUAN VỀ BỘ NHỚ NGOÀI**

* 1. **Cấu trúc vật lý**
     1. **Đĩa từ (Platter)**
* Đĩa từ chỉ sử dụng được sau khi đã định dạng, đó là việc tổ chức, sắp xếp các vùng lưu trữ thông tin trên đĩa.
* Về mặt vật lý đĩa từ được chia thành:
* Rãnh từ (track): Là các vùng vòng tròn đồng tâm, có bề dày xác định dùng để

ghi từ, các rãnh cách nhau bởi vành hẹp không được từ hóa.

* Cung từ (sector): Mỗi rãnh từ được chia thành các cung (sector), mỗi sector = 512 byte, các sector được đánh số.
* Liên cung (Cluster): Một nhóm các sector liên tiếp, thường 2/4/8 sector.

Từ trụ (Cylinder): Các track có cùng bán kính tạo thành một từ trụ (Cylinder).



*Hình 1.1: Cấu trúc vật lý của đĩa từ*

**1.1.2. Các rãnh từ (Track)**

Trên một mặt làm việc của đĩa từ chia ra nhiều vòng tròn đồng tâm thành các **track**.

Track có thể được hiểu đơn giản giống các rãnh ghi dữ liệu giống như các [đĩa nhựa](https://vi.wikipedia.org/w/index.php?title=%C4%90%C4%A9a_nh%E1%BB%B1a&action=edit&redlink=1) (ghi âm nhạc trước đây) nhưng sự cách biệt của các rãnh ghi này không có các gờ phân biệt và chúng là các vòng tròn đồng tâm chứ không nối tiếp nhau thành dạng xoắn trôn ốc như [đĩa nhựa](https://vi.wikipedia.org/w/index.php?title=%C4%90%C4%A9a_nh%E1%BB%B1a&action=edit&redlink=1). Track trên ổ đĩa cứng không cố định từ khi sản xuất, chúng có thể thay đổi vị trí khi định dạng cấp thấp ổ đĩa (*low format*).

Khi một ổ đĩa cứng đã hoạt động quá nhiều năm liên tục, khi kết quả kiểm tra bằng các phần mềm cho thấy xuất hiện nhiều khối hư hỏng (bad block) thì có nghĩa là phần cơ của nó đã rơ rão và làm việc không chính xác như khi mới sản xuất, lúc này thích hợp nhất là [format cấp thấp](https://vi.wikipedia.org/wiki/%E1%BB%94_%C4%91%C4%A9a_c%E1%BB%A9ng#Format_c%E1%BA%A5p_th%E1%BA%A5p) cho nó để tương thích hơn với chế độ làm việc của phần cơ.

**1.1.3. Sector:**

Trên *track* chia thành những phần nhỏ bằng các đoạn hướng tâm thành các **sector**. Các sector là phần nhỏ cuối cùng được chia ra để chứa dữ liệu. Theo chuẩn thông thường thì một sector chứa dung lượng 512 [byte](https://vi.wikipedia.org/wiki/Byte).

Số sector trên các track là khác nhau từ phần rìa đĩa vào đến vùng tâm đĩa, các ổ đĩa cứng đều chia ra hơn 10 vùng mà trong mỗi vùng có số sector/track bằng nhau.

Bảng sau cho thấy các khu vực với các thông số khác nhau và sự ảnh hưởng của chúng đến tốc độ truyền dữ liệu của ổ cứng Các khu vực ghi dữ liệu của ổ đĩa cứng Hitachi Travelstar 7K60 2,5".

**1.1.4. Cylinder:**

Tập hợp các track cùng bán kính ở các mặt đĩa khác nhau tạo thành cylinder (trụ). Trên đĩa hai mặt, một cylinder sẽ bao gồm rãnh 1 của mặt trên trên và rãnh 1 của mặt dưới. Trên các đĩa cứng sắp xếp cái này chồng lên cái kia, một cylinder gồm các rãnh trên cả hai mặt của tất cả các đĩa. Trên một ổ đĩa cứng có nhiều cylinder bởi có nhiều track trên mỗi mặt của đĩa từ.

**1.1.5. Đầu đọc/ghi (Read Write Heads)**

Đầu đọc đơn giản được cấu tạo gồm lõi [ferit](https://vi.wikipedia.org/wiki/Ferit) (trước đây là lõi [sắt](https://vi.wikipedia.org/wiki/S%E1%BA%AFt)) và cuộn dây (giống như [nam châm](https://vi.wikipedia.org/wiki/Nam_ch%C3%A2m) điện). Gần đây các [công nghệ](https://vi.wikipedia.org/wiki/C%C3%B4ng_ngh%E1%BB%87) mới hơn giúp cho ổ đĩa cứng hoạt động với mật độ xít chặt hơn như: chuyển các hạt từ sắp xếp theo phương [vuông góc](https://vi.wikipedia.org/wiki/Vu%C3%B4ng_g%C3%B3c) với bề mặt đĩa nên các đầu đọc được thiết kế nhỏ gọn và phát triển theo các ứng dụng công nghệ mới.

Đầu đọc trong đĩa cứng có công dụng đọc [dữ liệu](https://vi.wikipedia.org/wiki/D%E1%BB%AF_li%E1%BB%87u) dưới dạng từ hoá trên bề mặt đĩa từ hoặc từ hoá lên các mặt đĩa khi ghi dữ liệu.

Số đầu đọc ghi luôn bằng số mặt hoạt động được của các đĩa cứng, có nghĩa chúng nhỏ hơn hoặc bằng hai lần số đĩa (nhỏ hơn trong trường hợp ví dụ hai đĩa nhưng chỉ sử dụng 3 mặt).

**1.1.6. Cần di chuyển đầu đọc/ghi (Disk controler)**

Cần di chuyển đầu đọc/ghi là các thiết bị mà đầu đọc/ghi gắn vào nó. Cần có nhiệm vụ di chuyển theo phương song song với các đĩa từ ở một khoảng cách nhất định, dịch chuyển và định vị chính xác đầu đọc tại các vị trí từ mép đĩa đến vùng phía trong của đĩa (phía trục quay).

Các cần di chuyển đầu đọc được di chuyển đồng thời với nhau do chúng được gắn chung trên một trục quay (đồng trục), có nghĩa rằng khi việc đọc/ghi [dữ liệu](https://vi.wikipedia.org/wiki/D%E1%BB%AF_li%E1%BB%87u) trên bề mặt (trên và dưới nếu là loại hai mặt) ở một vị trí nào thì chúng cũng hoạt động cùng vị trí tương ứng ở các bề mặt đĩa còn lại.

Sự di chuyển cần có thể thực hiện theo hai phương thức:

* Sử dụng [động cơ bước](https://vi.wikipedia.org/wiki/%C4%90%E1%BB%99ng_c%C6%A1_b%C6%B0%E1%BB%9Bc) để truyền [chuyển động](https://vi.wikipedia.org/wiki/Chuy%E1%BB%83n_%C4%91%E1%BB%99ng).
* Sử dụng [cuộn cảm](https://vi.wikipedia.org/wiki/Cu%E1%BB%99n_c%E1%BA%A3m) để di chuyển cần bằng lực từ.

**Vì sao phải quản lý bộ nhớ ngoài ?**

* Khi lưu trữ chương trình hoặc dữ liệu, các hệ thống máy tính cần sử dụng bộ nhớ ngoài( đĩa từ, băng từ,…)
* Nhiệm vụ chính của hệ điều hành phải đảm bảo các chức năng sau:

1. Quản lý không gian nhớ tự do trên bộ nhớ ngoài (Free space mange).
2. Cấp phát không gian nhớ tự do ( Allocations Methods).
3. Cung cấp các khả năng định vị bộ nhớ ngoài.
4. Lập lịch cho bộ nhớ ngoài (Disk scheduling).

**Chương 2 : Các dạng lưu trữ dữ liệu trên hệ điều hành Windows**

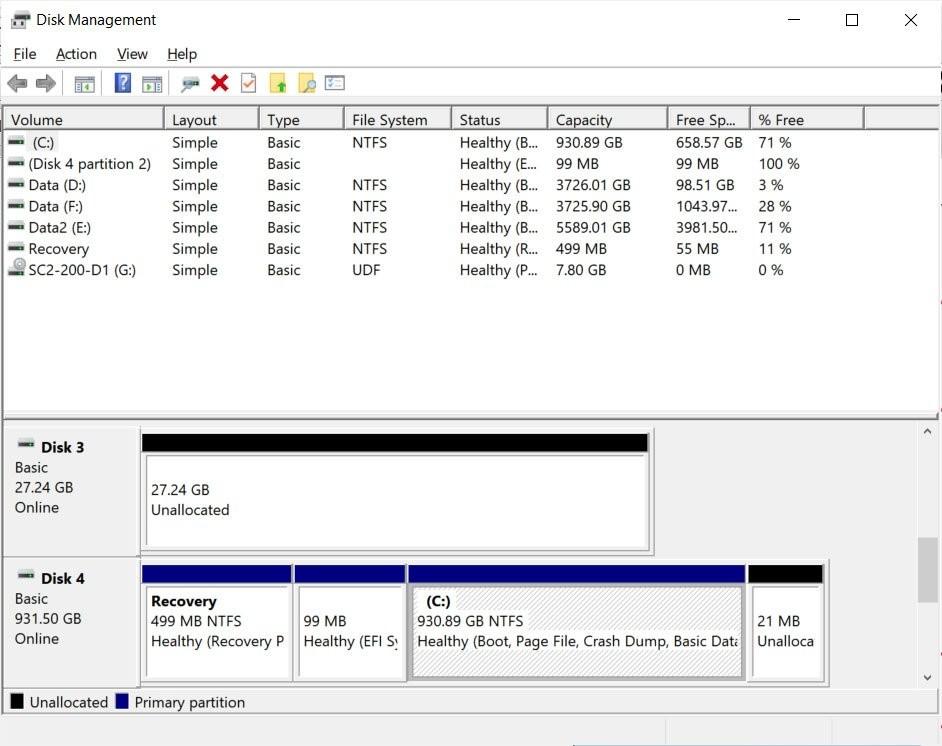
Hệ điều hành Windows hỗ trợ hai loại đĩa lưu trữ chủ yếu là ***Basic*** (cơ bản) và ***Dynamic*** (động).

**2.1. Lưu trữ cơ bản ( Basic storage).**

Gồm các phân vùng cơ bản (Partition Primary), hay còn gọi là phân vùng chính, và phân vùng mở rộng (Extended Partition). Phân vùng tạo ra đầu tiên trên đĩa được gọi là phân vùng chính và toàn bộ không gian cấp cho phân vùng sẽ được sử dụng trọn vẹn. Mỗi ổ đĩa vật lý có thể tạo tối đa bốn phân vùng chính hoặc ba phân vùng chính và một phân vùng mở rộng . Với phân vùng mở rộng, ta có thể tạo ra tùy ý số phân vùng logic khác.

Trên ổ cứng có một vùng nhỏ dùng để ghi bảng phân vùng ổ đĩa (Disk partition table). Đây là nơi hệ điều hành sẽ đọc để theo dõi cách thức phân chia đang tồn tại trên ổ đĩa. Bảng phân vùng ổ đĩa có độ lớn 64 byte chia làm 4 mục, các thông tin về mỗi phân vùng chính được ghi trên một mục chiếm 16 byte, ổ cứng vật lý chỉ có thể chia làm 4 phân vùng cũng là lý do đó. Tại 1 thời điểm chỉ có một phân vùng được nhận quyền khởi động, đó là phân vùng chứa hệ điều hành dùng để khởi động máy.

Một Basic storage có hai cách quản lý phân vùng là MBR disks (Master Boot Record) và GPT disks (GUIDs Partition Table).Đối với MBR disks, chúng ta có thể tạo được nhiều nhất 4 phần vùng chính (Primary Partition), hoặc 3 phân vùng chính và một phân vùng mở rộng (Extended Partion) và trong phân vùng mở rộng (Extended Partion) ta có thể tạo vô hạn các ổ đĩa luận lý (Logical Drive).Đối với GPT disks, chúng ta có thể tạo lên đến 128 phân vùng chính (Primary Partition). Bởi vì GPT disks không giới hạn 4 phân vùng chính nên chúng ta không cần tạo phân vùng mở rộng (Extended Partion) hay các ổ đĩa luận lý (Logical Drive)



**Hình 2.1. Quản lý đĩa cứng trên Windows**

**2.2. Lưu trữ động (Dynamic storage)**

Đĩa lưu trữ động được chia thành các phân vùng động. Phân vùng động không chứa phân vùng hoặc ổ đĩa logic và chỉ có thế truy cập được trên hệ điều hành Windows server 2003 và Windows 2000. Windows server 2003 và Windows 2000 hỗ trợ 5 loại phân vùng động là **spanned, simple, striped, mirrored và RAID-5**

* + Ưu điểm của công nghệ lưu động so với công nghệ lưu trữ căn bản là:
    - * Cho phép nhiều ổ đĩa vật lý để tạo thành các ổ đĩa logic (Volume).
      * Cho phép ghép nhiều vùng trống không liên tục trên nhiều đĩa cứng vật lý để tạo ổ đĩa logic.
      * Có thể tạo ra các ổ đĩa logic có khả năng dung lỗi cao và tắng tốc truy xuất
  + Nhược điểm của đĩa lưu trữ động là chỉ được hỗ trợ bởi một số hệ điều hành của Microsoft, ngoài ra nó đòi hỏi phải có từ hai ổ cứng trở lên, trong một số trường hợp các ổ cứng phải giống hệt nhau về kích thước… do đó đĩa lưu trữ động chủ yếu chỉ sử dụng trong các hệ thống lớn, trong các máy chủ, hoặc trong các nơi có đòi hỏi cấp thiết về vấn đề an toàn và tốc độ xử lý dữ liệu

**2.2.1. Spanned volume**

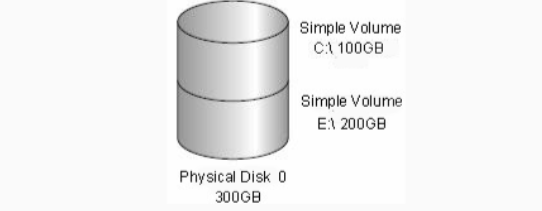
Bao gồm 1 hoặc nhiều đĩa lưu trữ động ( tối đa là 32 đĩa ). Sử dụng khi người dùng muốn tăng kích thước phân vùng. Dữ liệu được ghi lên vùng theo thứ tự và hết đĩa này đến đĩa khác. Thông thường người quản trị sử dụng phân vùng *spanned* khi ổ đĩa đang sử dụng trong phân vùng sắp bị đầy và muốn tăng kích thước của phân vùng bằng cách bổ sung thêm một đĩa khác.

Do dữ liệu được ghi tuần tự nên phân vùng loại này không tăng hiệu năng sử dụng. Nhược điểm chính của phân vùng *spanned* là nếu một đĩa bị hỏng thì toàn bộ dữ liệu trên phân vùng sẽ không thể truy xuất được. Dữ liệu trên *spanned volume* được chép phân bổ trên 2 ổ cứng vật lý trở lên, các ổ cứng không nhất thiết phải giống nhau và chúng được ghép lại thành một Volume duy nhất, tuy vậy chúng không có khả năng đáp ứng vấn đề an toàn dữ liệu (Fault Tolerangcing), và tốc độ xử lý dữ liệu (Load Balancing), vì chưa có sự thay đổi về cơ chế (dữ liệu được chép đầy trên span volume ở disk 1 mới chép sang các disk còn lại).

**2.2.2. Simple volume**

Đây là một “phân vùng” thông thường. Nếu loại đĩa là “Cơ bản”, điều này sẽ tạo ra một phân vùng thực.Phân vùng *simple* không gian lấy từ một đĩa dynamic duy nhất. Không gian đĩa này có thể liên tục hoặc không liên tục. Hình sau minh hoạ một đĩa vật lý được chia thành hai volume đơn giản.

Dữ liệu trên simple volume chỉ được lưu trữ trên 1 ổ cứng vật lý, do đó vấn đề an toàn dữ liệu (Fault Tolerancing), và tăng tốc độ xử lý (Load Balancing) không được đảm bảo, khi ổ cứng vật lý hỏng, thì dữ liệu có nguy cơ bị mất.



**Hình 2.2. Một đĩa vật lý được chia thành hai phân vùng đơn giản**

**2.2.3 Striped Volume (RAID-0)**

Lưu trữ dữ liệu lên các dãy (strip) bằng nhau trên một hoặc nhiều đĩa vật lý (tối đa là 32). Do dữ liệu được ghi tuần tự lên từng dãy, nên bạn có thể thi hành nhiều tác vụ I/O đồng thời, làm tăng tốc độ truy xuất dữ liệu. Thông thường, người quản trị mạng sử dụng volume striped để kết hợp dung lượng của nhiều ổ đĩa vật lý thành một đĩa logic đồng thời tăng tốc độ truy xuất.

Dữ liệu trên striped volume có thể được trao đổi cùng lúc trên 2 ổ cứng vật lý trở lên, dung lượng trên các ổ cứng vật lý của striped volume phải bằng nhau. Striped Volume có sự thay đổi trong cơ chế hoạt động, dữ liệu khi được chép trên striped được chia ra và chép đều trên các disk, vì thế striped đáp ứng được vấn đề tốc độ xử lý dữ liệu (Load Balancing), tuy nhiên striped không đáp ứng được vấn đề an toàn dữ liệu (Fault Tolerangcing), nếu một ổ đĩa bị hỏng thì dữ liệu trên toàn bộ volume mất giá trị

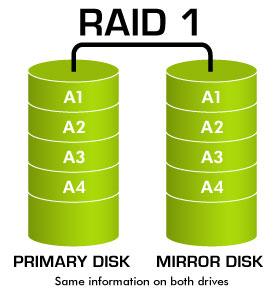


**Hình 2.3. Trao đổi dữ liệu trên các ổ cứng vật lý**

**2.2.4 Mirror volume (RAID-1)**

Là hai bản sao của một volume đơn giản. Bạn dùng một ổ đĩa chính và một ổ đĩa phụ. Dữ liệu khi ghi lên đĩa chính đồng thời cũng sẽ được ghi lên đĩa phụ. Volume dạng này cung cấp khả năng dung lỗi tốt. Nếu một đĩa bị hỏng thì ổ đĩa kia vẫn làm việc và không làm gián đoạn quá trình truy xuất dữ liệu.Nhược điểm của phương pháp này là bộ điều khiển đĩa phải ghi lần lượt lên hai đĩa, làm giảm hiệu năng

Để tăng tốc độ ghi đồng thời cũng tăng khả năng dung lỗi, bạn có thể sử dụng một biến thể của volume mirrored là duplexing. Theo cách này bạn phải sử dụng một bộ điều khiển đĩa khác cho ổ đĩa thứ hai.

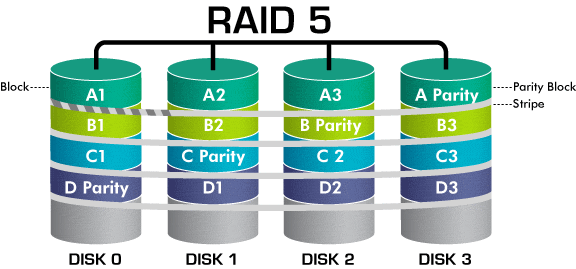


**Hình 2.4. Dữ liệu khi ghi lên đĩa chính đồng thời cũng sẽ được ghi lên đĩa phụ**

Nhược điểm chính của phương pháp này là chi phí cao. Để có một volume 4GB bạn phải tốn đến 8GB cho hai ổ đĩa.

**2.2.5. Volume RAID-5**

Raid-5 Volume là giải pháp kết hợp các loại volume (Striped Volume RAID-0, Mirror Volume RAID-1)mà ta đã đề cập ở trên. Raid-5 đáp ứng cho chúng ta cả 2 vấn đề an toàn dữ liệu (Fault Tolerangcing), và tăng tốc độ xử lý dữ liệu (Load Balancing). Để đáp ứng 2 vấn đề trên, Raid-5 đòi hỏi phải sử dụng 3 ổ đĩa cứng vật lý, và sử dụng thuật toán Parity (khi 1 trong 3 đĩa bị hỏng, thuật toán Parity sẽ tự chép những bit bị mất). Vì phải chứa thêm bit Parity nên dung lượng của Raid-5 Volume sẽ chỉ bằng 2/3 dung lượng ta cấu hình (1/3 còn lại là để chứa bit Parity). Ưu điểm chính của kỹ thuật này là khả năng dung lỗi cao và tốc độ truy xuất cao bởi sử dụng nhiều kênh I/O.



**Hình 2.5. Hoạt động của Raid-5**

Để setup Raid 5 ta cần tối thiểu 3 ổ cứng. Theo như hình minh họa phương án lưu trữ của Raid 5 như sau. Giả sử có 1 file A thì khi lưu trữ sẽ tách ra 3 phần A1, A2, A3. Ba phần nãy sẽ tương ứng lưu trên ổ đĩa Disk 0, Disk 1, Disk 2, còn ổ đĩa Disk 3 sẽ giữ bản sao lưu backup của 3 phần này. Tương tự các file sau cũng vậy và tùy theo tiến trình thực hiện mà bản sao lưu có thể được lưu ở bất kì 1 trong những ổ trong cụm Raid.

**Chương 3. Chương trình quản lý bộ nhớ ngoài Disk Manager**

**3.1. Thuộc tính của đĩa:**

**Cấu trúc vật lý**

Một đĩa từ gồm một hay nhiều lá đĩa xếp chồng lên nhau

- Mỗi mặt của đĩa được chia thành nhiều rãnh tròn đồng tâm gọi là Track

- Mỗi Track được chia thành các cung tròn bằng nhau gọi là Sector ( Cluster)

- Các Track có cùng số hiệu được gọi là từ trụ (Cylinder)

- Mỗi mặt của đĩa có một đầu từ để đọc/ghi dữ liệu trên mặt đĩa đó, để điều khiển đầu đọc/ghi dữ liệu cần có một trình điều khiển đĩa.

- Thông tin trên đĩa từ bị tham chiếu bởi các thành phần:

+ ổ đĩa (Drive)

+ Mặt đĩa (Side)

+ Rãnh đĩa (Track)

+ Cung tròn trên track (Sector) - Cluster

OS xem đĩa như là một mảng 1 chiều mà thành phần là các khối đĩa (Disk Block). Mỗi khối đĩa ghi các thông tin về Cylinder, track, Sector mà OS có thể định vị trên đĩa đó.

**3.2. Thuộc tính của phân vùng hoặc đĩa cục bộ**

**Thư mục thiết bị**

| Master Boot | Boot Sector | FAT or None FAT | ROOT Directory | Database |
| --- | --- | --- | --- | --- |
| Gồm 2 thành phần là Master boot Record và Partition | Là Sector đầu tiên của ổ đĩa có nhiệm vụ khởi động | Bảng định vị và quản lý dữ liệu trên đĩa | Cấu trúc thư mục để tổ chức dữ liệu trên đĩa | Vùng lưu trữ CSDL của người dùng |

Trên mỗi đĩa thông thường có một thư mục thiết bị cho biết đĩa gồm những thông tin gì, độ dài, kiểu, người sở hữu, thời điểm khởi tạo, vị trí, vùng không gian nhớ được cấp phát như thế nào

Thư mục thiết bị được tạo ngay ở trên đĩa tại một vùng nhớ đặc biệt

**Chương 4. quản lý không gian nhớ tự do trong hệ điều hành**

Quản lý bộ nhớ là công việc của hệ điều hành với sự hỗ trợ của phần cứng nhằm phân phối, sắp xếp các process trong bộ nhớ sáo cho hiệu quả tối ưu

**4.1 Quản lý bộ nhớ bằng phương pháp liệt kê( free list)**

Trong phương pháp này hệ thống sử dụng một danh sách móc nối để liệt kê các Block tự do. Con trỏ đầu trong danh sách chỉ tới block đĩa tự do đầu tiên, mỗi block đĩa có một con trỏ để trỏ tới khối kế tiếp.

Ưu điểm : Tiết kiệm không gian đĩa

Nhược điểm : Tăng thời gian truy nhập trên đĩa

**Ví dụ : Giả sử Block 3, 7, 9 tự do có thể phân bổ được**



| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |

**4.2 Quản lý bộ nhớ bằng phương pháp lập nhóm(grouping)**

Nhóm các Block tự do liên tiếp thành một nhóm, block tự do đầu tiên trong nhóm ghi địa chỉ của các block còn lại trong nhóm, block tự do cuối cùng trong nhóm ghi địa chỉ của block tự do đầu tiên của nhóm tiếp theo.



***Hình 4.2 Mô tả không gian đĩa từ***

**Ví dụ: Theo hình 4.2, ta có bảng quản lý không gian nhớ tự do như sau:**

| **Nhóm** | **Block đầu** | **Block cuối** |
| --- | --- | --- |
| **1** | 2(2,3,4,5) | 5(8) |
| **2** | 8(8,9,10,11,12,13) | 13(17) |
| **3** | 17(17,18) | 18(25) |
| **4** | 25(25,26,27) | 27(...) |

**4.3 Quản lý bộ nhớ bằng phương pháp đếm (counting)**

Phương pháp đếm là sự biến đổi của phương pháp lập nhóm. Trong phương pháp này hệ thống lập danh sách quản lý địa chỉ của các block đĩa tự do đầu tiên và số lượng các block tự do liên tục kế tiếp các block đĩa đó.

**Ví dụ: Theo hình trên ta có danh sách quản lý không gian nhớ tự do như sau:**

| **Danh sách** | **Số lượng** |
| --- | --- |
| **2** | **4** |
| **8** | **6** |
| **17** | **2** |
| **25** | **3** |

# Chương 5: Các phương pháp cấp phát không gian nhớ tự do

* + 1. **Cấp phát kề (Contiguous)**

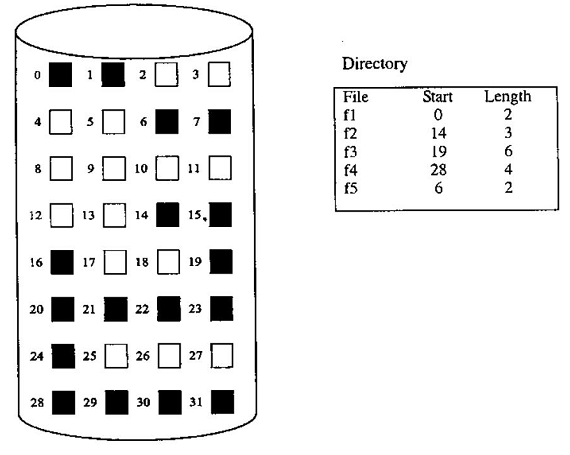
Để phân bố không gian nhớ cho một file, hệ thống chọn một đoạn liên tục các khối đĩa tự do để cấp phát cho filei đó. Với phương pháp này, để định vị file hệ thống chỉ cần biết địa chỉ của khối đĩa tự do đầu tiên và số lượng block đã dùng.

Ưu điểm của cấp phát liên tục là hỗ trợ phương pháp truy nhập tuần tự và truy nhập trực tiếp, nhưng tồn tại 3 nhược điểm chính:

- Phải chọn được thuật toán tối ưu để tìm các vùng không gian tự do cấp phát cho file (First – Fit, Best – Fit hoặc Worst – Fit)

- Có thể xảy ra trường hợp không đủ số khối đĩa tự do liên tiếp cần thiết để cấp phát cho file ( kích thước file lớn hơn vùng các khối đĩa liên tục lớn nhất).

- Trường hợp các khối đĩa tự do nằm tản mạn sẽ không được sử dụng, sẽ gây lãng phí không gian nhớ.

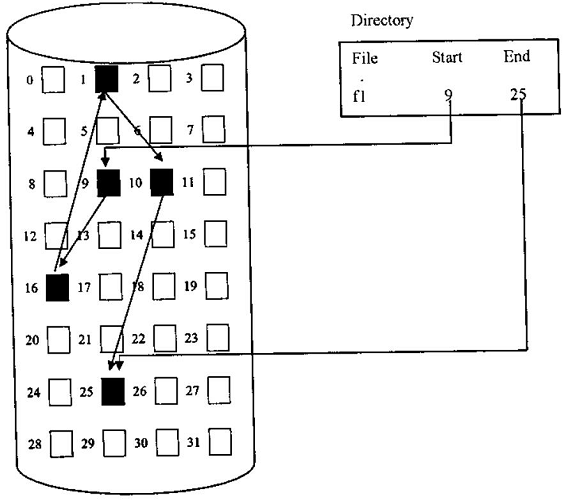


***Hình 5.1. không gian đĩa được cấp phát kề***

* 1. **Cấp phát liên kết (Linked)**

Trong phương pháp này, mỗi file được định vị trong thư mục thiết bị bằng 2 con trỏ, một cái trỏ tới khối đĩa đầu tiên, một cái trỏ tới khối đĩa cuối cùng đã cấp phát cho file. Trong mỗi khối đĩa đã cấp phát cũng có một con trỏ để trỏ tới khối đĩa kế tiếp.

Ví dụ: file f1 được cấp phát 5 khối đĩa có số hiệu 9, 16, 1, 11, 25; khối đầu là 9, khối cuối là 25.

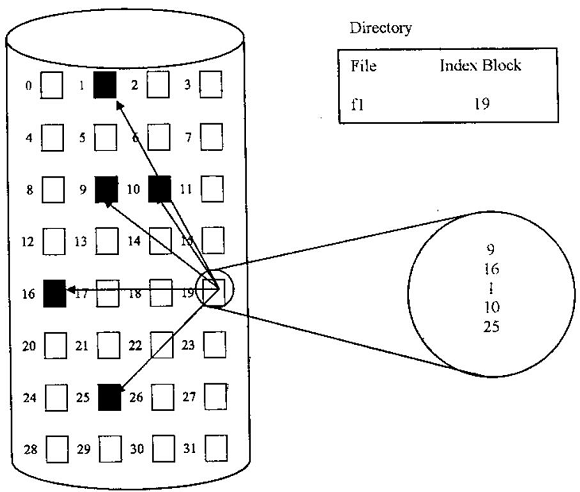


***Hình 5.2. cấp phát không gian đĩa liên kết***

Cấp phát liên kết có ưu điểm là sử dụng được các khối đĩa tự do nằm tản mạn nhưng chỉ hỗ trợ truy nhập tuần tự, không hỗ trợ truy nhập trực tiếp, độ tin cậy không đảm bảo nếu mất các con trỏ liên kết. Mặt khác phương pháp này tốn không gian nhớ để lưu trữ các con trỏ ( khoảng 0,38% không gian đĩa).

* 1. **Cấp phát theo chỉ số (Index)**

Phương pháp này, để cấp phát không gian nhớ cho một file, hệ thống sử dụng một khối đĩa đặc biệt gọi là khối đĩa chỉ số (Index block) cho mỗi file. Trong khối đĩa chỉ số chứa địa chỉ các khối đĩa đã cấp phát cho file, trong thu mục thiết bị địa chỉ của các khối đĩa chỉ số. Khi một khối đĩa được cấp phát cho file thì hệ thống loại bỏ địa chỉ của khối đĩa này khỏi danh sách của cấc khối đĩa tự do và cập nhật vào khối chỉ số của file.



***Hình 5.3. cấp phát không gian đĩa theo chỉ số***

Phương pháp cấp phát theo chỉ số hỗ trợ truy nhập trực tiếp nhưng lãng phí không gian nhớ dành cho khối đĩa chỉ số.

Điểm này sinh ra câu hỏi: Khối chỉ số nên lớn bao nhiêu? Tuy nhiên, nếu khối chỉ số quá nhỏ nó không thể quản lý đủ các con trỏ cho một tập tin lớn, cần có một cơ chế giải quyết vấn đề này:

+ ***Cơ chế liên kết (linked scheme):*** một khối chỉ số thường là một đĩa. Do đó, nó có thể được đọc và viết trực tiếp bởi chính nó. Để cho phép đối với các tập tin lớn, chúng ta có thể liên kết nhiều khối chỉ số với nhau. Thí dụ, một khối chỉ số có thể chứa một header nhỏ cho tên tập tin và một tập hợp của các địa chỉ 100 khối đĩa đầu tiên. Địa chỉ tiếp theo (từ cuối cùng trong khối chỉ số ) là nil (đối với một tập tin nhỏ ) hay một con trỏ tới khối chỉ số khác (cho một tập tin lớn).

+ ***Chỉ số nhiều cấp (multilevel index):*** một biến dạng của biểu diễn liên kết là dùng khối chỉ số cấp 1 để chỉ tới khối chỉ số cấp 2. Khối chỉ số cấp 2 chỉ tới các khối tập tin. Để truy xuất một khối, hệ điều hành dùng chỉ số cấp 1 để tìm một khối chỉ số cấp 2 và khối đó tìm khối dữ liệu mong muốn. Tiếp cận này có thể được tiếp tục tới cấp 3 hay cấp 4, tùy thuộc vào kích thước tập tin lớn nhất được mong muốn. Với khối có kích thước 4,096 bytes, chúng ta có thể lưu 1,024 con trỏ 4 bytes trong một khối chỉ số. Chỉ số hai cấp cho phép 1,048,576 khối dữ liệu, cho phép tập tin có kích thước tới 4GB.

+ ***Cơ chế kết hợp (combined scheme):*** một biến dạng khác được dùng trong UFS là giữ 15 con trỏ đầu tiên của khối chỉ số trong inode của tập tin. 12 con trỏ đầu tiên của 15 con trỏ này chỉ tới khối trực tiếp (direct blocks), nghĩa là chúng chứa các địa chỉ của khối mà chứa dữ liệu của tập tin. Do đó, dữ liệu đối với các tập tin nhỏ (không lớn hơn 12 khối) không cần một khối chỉ số riêng. Nếu kích thước khối là 4KB, thì 48KB dữ liệu có thể truy xuất trực tiếp. 3 con trỏ tiếp theo chỉ tới các khối gián tiếp (indirect blocks). Con trỏ khối gián tiếp thứ nhất là đại chỉ của khối gián tiếp đơn (single indirect blocks). Khối gián tiếp đơn là một khối chỉ số không chứa dữ liệu nhưng chứa địa chỉ của các khối dữ liệu. Sau đó, có con trỏ khối gián tiếp đôi (double indirect block) chứa địa chỉ của một khối mà khối này chứa địa chỉ của các khối chứa con trỏ chỉ tới khối dữ liệu thật sự. Con trỏ cuối cùng chứa địa chỉ của khối gián tiếp ba (triple indirect block). Với phương pháp này, số khối có thể cấp phát tới một tập tin vượt quá hạn lượng không gian có thể đánh địa chỉ bởi các con trỏ tập tin 4 bytes hay 4 GB. Nhiều cài đặt UNIX gồm Solaris và AIX của IBM hỗ trợ tới 64 bit con trỏ tập tin. Các con trỏ có kích thước này cho phép các tập tin và hệ thống tập tin có kích thước tới terabytes. Một inode được hiển thị trong hình 5.4:

| **Mode** |
| --- |
| **Owners (2)** |
| **timestamps(3)** |
| **size block**  **count** |
| **Direct blocks**  **.**  **.**  **.** |
| **single indirect** |
| **double indirect** |
| **triple indirect** |



***Hình 5.4. Inode của UNIX***

Cơ chế cấp phát lập chỉ số gặp một số vấn đề khó khăn về năng lực như cấp phát liên kết. Đặc biệt các khối chỉ số có thể được lưu trữ (cache) trong bộ nhớ,nhưng các khối dữ liệu có thể trải rông khắp phân khu.

# Chương 6: Lập lịch cho đĩa từ trong hệ điều hành windows

* 1. **Khái niệm về lập lịch cho đĩa**

Thời gian truy nhập đĩa phụ thuộc vào ba yếu tố: thời gian di chuyển đầu từ đọc/ghi đến track hoặc cylinder cần thiết (seek-time), thời gian định vị đầu từ đọc/ghi tại khối đĩa cần truy nhập (latency-time) và thời gian truy nhập dữ liệu (transfer-time). Thời gian định vị đầu từ đọc/ghi và thời gian truy nhập dữ liệu thông thường cố định và phụ thuộc cấu trúc kỹ thuật của ổ đĩa. Do đó để tăng tốc độ truy nhập đĩa, các hệ điều hành thường quan tâm tới thời gian di chuyển đầu từ đọc/ghi.

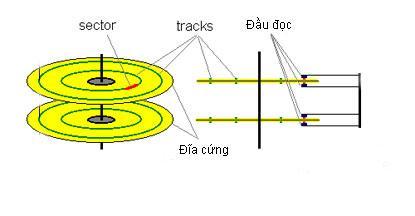
Lập lịch cho đĩa là xây dựng các thuật toán dịch chuyển đầu từ đọc ghi sao cho thời gian truy nhập đĩa là tối ưu nhất

Thời gian truy nhập đĩa

- Thời gian di chuyển đầu từ đọc ghi đến strack thích hợp(seek-time)

- Thời gian chờ cho khối cần thiết dưới đầu đọc(latency -time)

- Thời gian vận chuyển dữ liệu giữa đĩa và bộ nhớ chính(transfer-time)



Tất cả mọi công việc đều phụ thuộc vào việc nạp chương trình và nhập xuất tập tin, do đó điều quan trọng là dịch vụ đĩa phải càng nhanh càng tốt. Hệ điều hành có thể tổ chức dịch vụ truy xuất đĩa tốt hơn bằng cách lập lịch yêu cầu truy xuất đĩa.

Tốc độ đĩa bao gồm ba phần. Để truy xuất các khối trên đĩa, trước tiên phải di chuyển đầu đọc đến track hay cylinder thích hợp, thao tác này gọi là seek và thời gian để hoàn tất gọi là *seek time*. Một khi đã đến đúng track, còn phải chờ cho đến khi khối cần thiết đến dưới đầu đọc. Thời gian chờ này gọi là *latency time*. Cuối cùng là vận chuyển dữ liệu giữa đĩa và bộ nhớ chính gọi là *transfer time*. Tổng thời gian cho dịch vụ đĩa chính là tổng của ba khoảng thời gian trên. Trong đó *seek time* và *latency time* là mất nhiều thời gian nhất, do đó để giảm thiểu thời gian truy xuất hệ điều hành đưa ra các thuật toán lập lịch truy xuất.

**6.2.Các thuật toán lập lịch cho đĩa**

- First come first Served(FCFS)

- Shortest seek time first(SSTF)

- Scan

- C-Scan

- Look

- C-Look

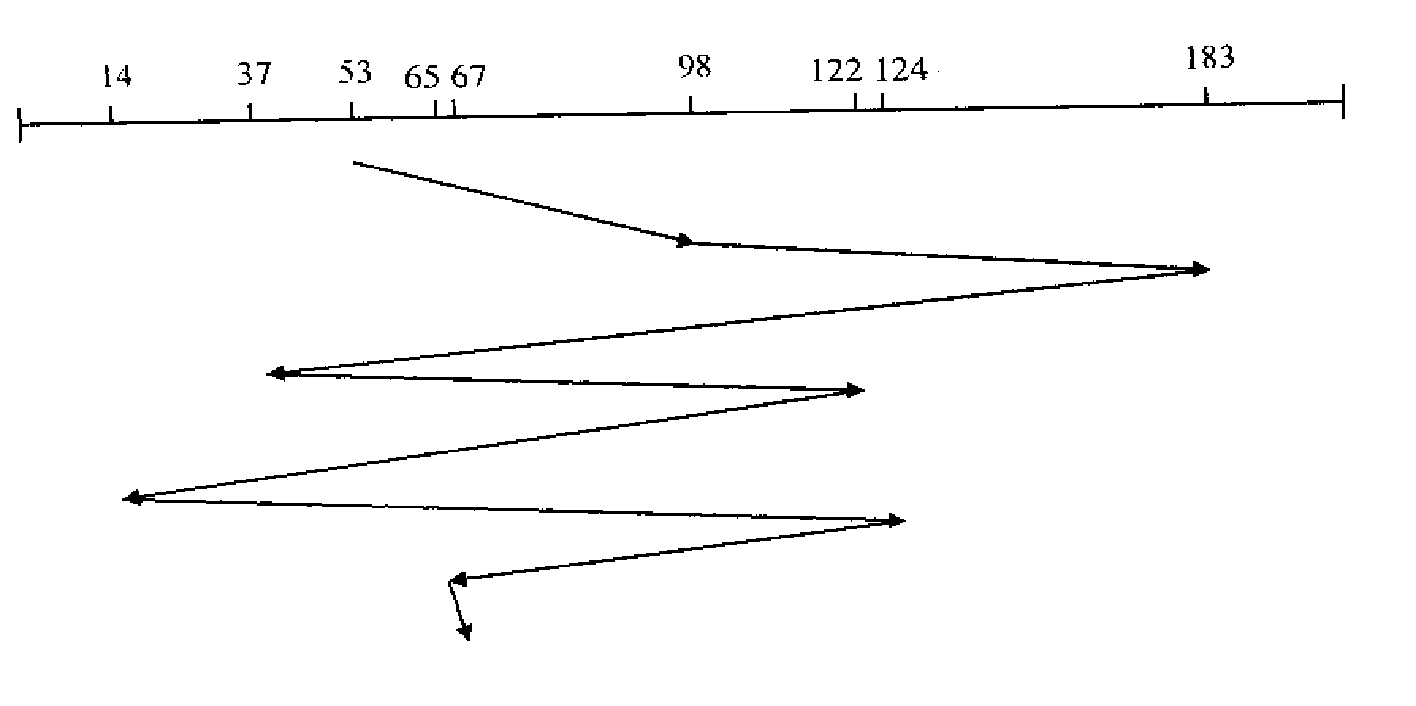
**6.2.1.First come first Served(FCFS)**

-Để truy nhập tới 1 file, hệ thống sẽ tổ chức một hàng đợi các yêu cầu phục vụ của các track(lưu trữ dữ liệu của file cần truy nhập)

-Nội dung: track nào có yêu cầu phục vụ trước thì đầu đọc ghi sẽ dịch chuyển tới đó trước

-Ví dụ:File F1 được phân bổ lần lượt các track số thứ tự :98,183,37,122,14,124,65,67  
giả sử đầu đọc đang ở track 53

-Sơ đồ dịch chuyển theo FCFS(640 bước dịch chuyển)



Ưu,nhược điểm

-Ưu điểm:

+Dễ lập trình

+Các track cần truy xuất là liên tục

-Nhược điểm

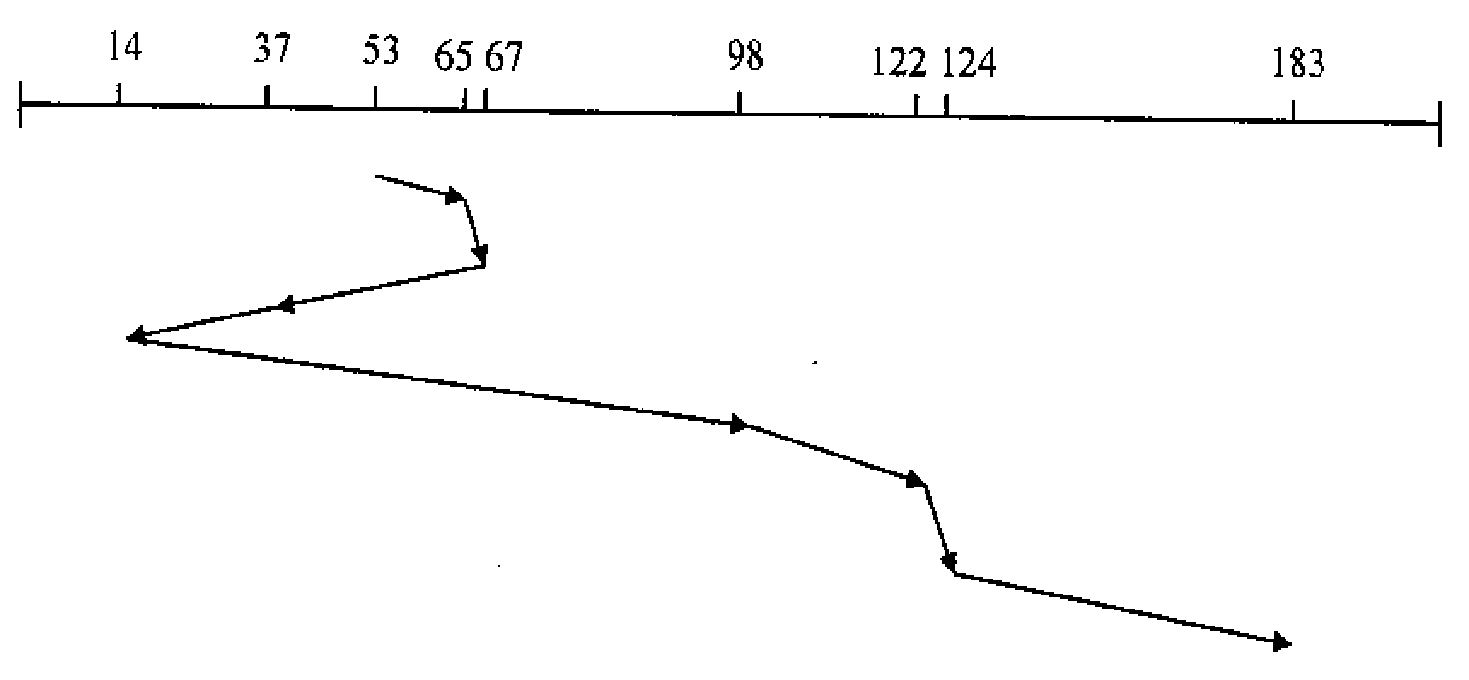
+Số track mà đầu đọc phải di chuyển là nhiều

+Hiệu quả của thuật toán phụ thuộc vào thứ tự của các track trong hàng đợi

**6.2.2.Shortest Seek time First**

-Nội dung: track nào có thời gian di chuyển đầu từ đọc ghi ngắn nhất thì phục vụ trước

-Ví dụ:File F1 được phân bổ lần lượt các track số thứ tự :98,183,37,122,14,124,65,67  
giả sử đầu đọc đang ở track 53

-Sơ đồ dịch chuyển theo SSTF(Tổng quãng đường dịch chuyển là 360)

Ưu, nhược điểm SSTF

-Ưu điểm:

+Số track mà đầu đọc phải đi chuyển giảm

-Nhược điểm

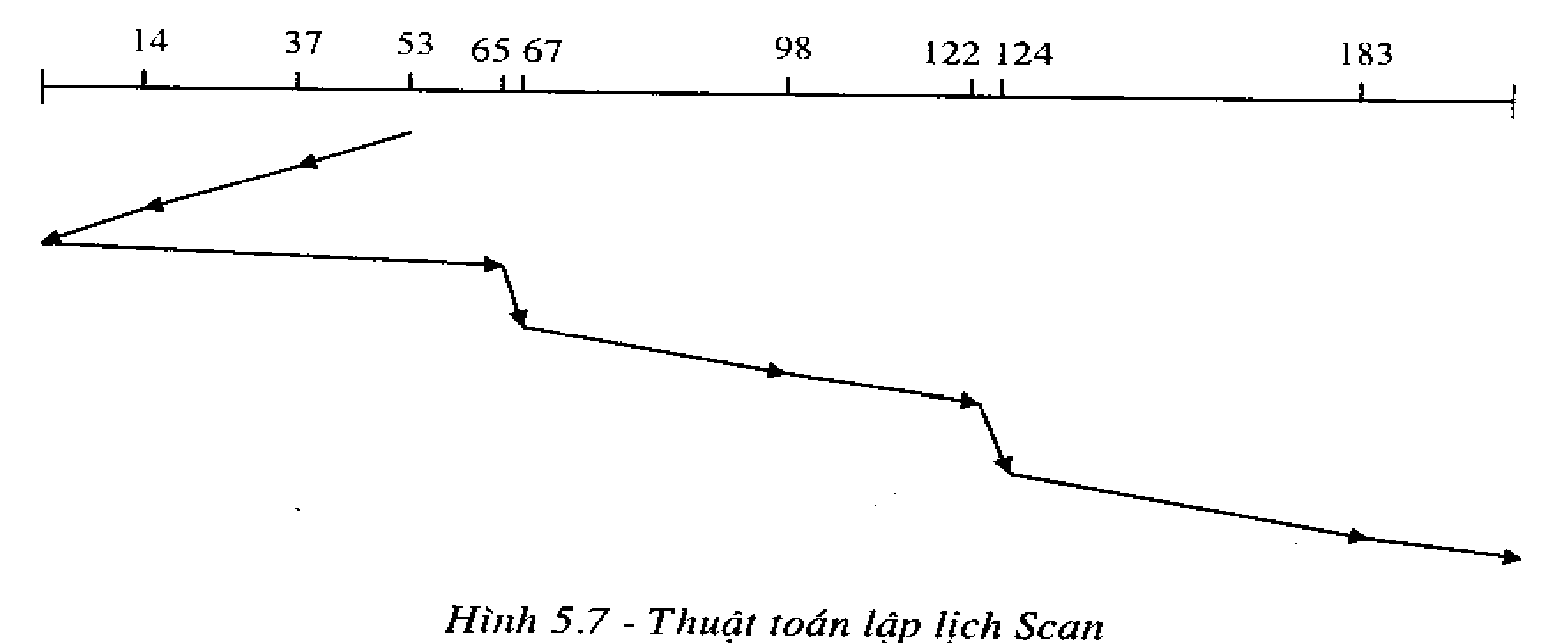
+Có thể gây ra 1 số yêu cầu không bao giờ được phục vụ

**6.2.3.Scan**

-Nội dung: Đầu đọc của đĩa di chuyển từ một phía (ví dụ bên ngoài hoặc bên trong đĩa) sang phía kia để phục vụ các yêu cầu đọc, sau đó di chuyển ngược lại... quá trình này lặp đi lặp lại

-Ví dụ:File F1 được phân bổ lần lượt các track số thứ tự :98,183,37,122,14,124,65,67  
giả sử đầu đọc đang ở track 53

-Tổng quãng đường dịch chuyển là 252



Đặc điểm:

+ Phương thức h/đ như thang máy

+ Số bước đầu đọc phải di chuyển giảm

**6.2.4.C-Scan**

-Nội dung: Đầu đọc chuyển từ một phía (trong/ngoài)sang phía kia và phục vụ các yêu cầu. Khi sang đến phía kia, đầu đọc quay trở lại nhưng trong khi quay trở lại không phục vụ yêu cầu nào.(quét 1 chiều)

-Ví dụ: File F1 được phân bổ lần lượt các track số thứ tự :98,183,37,122,14,124,65,67  
giả sử đầu đọc đang ở track 53

-Số bước dịch chuyển của đầu từ 252

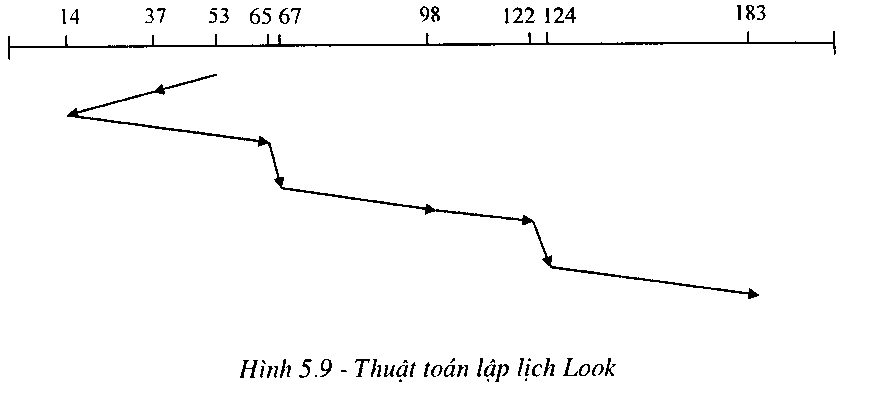


**6.2.5.Look**

-Nội dung: tương tự như Scan nhưng trong thuật toán này đầu đọc ghi chỉ quét trong phạm vi các track có nhu cầu phục vụ không quét tới track đầu tiên hoặc cuối cùng

-Ví dụ: File F1 được phân bổ lần lượt các track số thứ tự :98,183,37,122,14,124,65,67  
giả sử đầu đọc đang ở track 53

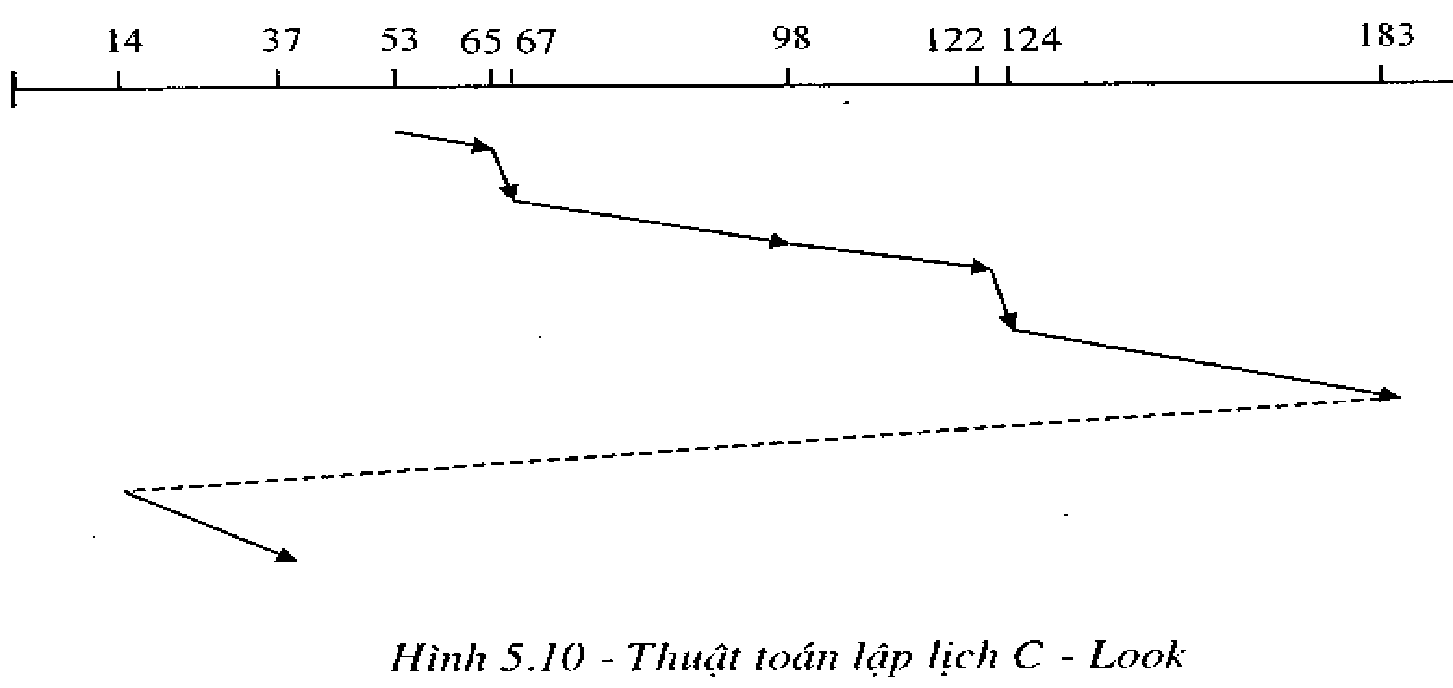
-Số bước dịch chuyển của đầu từ 208



**6.2.6.C-Look**

-Nội dung: Tương tự như Look nhưng đầu đọc ghi không phục vụ đường về

-Ví dụ: File F1 được phân bổ lần lượt các track số thứ tự :98,183,37,122,14,124,65,67  
giả sử đầu đọc đang ở track 53



Lựa chọn một giải thuật lập lịch đĩa:

-FCFS là thuật toán phù hợp khi các track cần truy xuất là liên tục

-SSTF phổ biến và có hiệu quả tốt.

-SCAN và LOOK thích hợp cho những hệ thống phải truy xuất dữ liệu lớn

**6.3.RAM Disks**

RAM disk là một phần bộ nhớ RAM được chuyển đổi để sử dụng như ổ lưu trữ. Vì tốc độ của RAM cực kỳ nhanh, thậm chí nhanh hơn so với SSD truyền thống, nên các ứng dụng/tác vụ sẽ đạt được hiệu suất lớn hơn khi chạy từ RAM disk.

-Ưu điểm của RAM disk:

Tốc độ đọc/ghi ổ đĩa RAM nhanh hơn tốc độ của những ổ cứng truyền thống, nên giúp các ứng dụng, thao tác nhanh, mượt mà hơn.

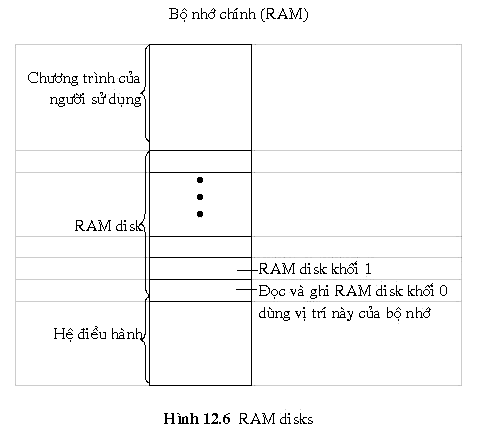
-Nhược điểm của RAM disk:

RAM đắt, dung lượng nhỏ. RAM thông thường chỉ từ 4GB-16GB, trên một số máy cao cấp hơn thì có thể có đến 64GB RAM. Nếu RAM của bạn dưới 8GB thì chỉ nên thử cho biết thôi.

Việc tạo một ổ RAM đòi hỏi phần mềm chuyên dụng.

Phần RAM bị biến thành RAM disk không thể phục vụ cho các tác vụ ghi nhớ thông thường nữa.

Tính không ổn định cao do đặc tính cố hữu của bộ nhớ truy cập tạm thời, chúng sẽ mất dữ liệu mỗi lần PC sập nguồn/mất điện/khởi động lại, nên không lưu các file quan trọng trên ổ này.



***Hình 6.5. Mô tả ý tưởng của RAM disk***

**6.4.Quản lý lỗi**

Đĩa là đối tượng mà khi truy xuất có thể gây nhiều lỗi. Một trong số các lỗi thường gặp là :

*Lỗi lập trình* : yêu cầu đọc các sector không tồn tại.

Lỗi lập trình xảy ra khi yêu cầu bộ điều khiển tìm kiếm cylinder không tồn tại, đọc sector không tồn tại, dùng đầu đọc không tồn tại, hoặc vận chuyển vào và ra bộ nhớ không tồn tại. Hầu hết các bộ điều khiển kiểm tra các tham số và sẽ báo lỗi nếu không thích hợp.

*Lỗi checksum tạm thời* : gây ra bởi bụi trên đầu đọc.

Bụi tồn tại giữa đầu đọc và bề mặt đĩa sẽ gây ra lỗi đọc. Nếu lỗi tồn tại, khối có thể bị đánh dấu hỏng bởi phần mềm.

*Lỗi checksum thường trực* : đĩa bị hư vật lý trên các khối.

*Lỗi tìm kiếm* : ví dụ đầu đọc đến cylinder 7 trong khi đó phải đọc 6.

*Lỗi điều khiển* : bộ điều khiển từ chối thi hành lệnh.

**6.5. Interleave**

Bộ điều khiển đọc ghi đĩa phải thực hiện hai chức năng là đọc/ghi dữ liệu và chuyển dữ liệu vào hệ thống. Để thực hiện được đồng bộ hai chức năng này, bộ điều khiển đọc đĩa cung cấp chức năng interleave. Trên đĩa các sector số hiệu liên tiếp nhau không nằm kế bên nhau mà có một khoảng cách nhất định, khoảng cách này được xác định bởi quá trình format đĩa. Ví dụ : giả sử hệ thống chỉ có 17 sector, và interleave được chọn là 4 thì các sector được bố trí theo thứ tự như sau :

1, 14, 10, 6, 2, 15, 11, 7, 3, 16, 12, 8, 4, 17, 13, 9, 5

Cách đọc lần lượt như sau :

 Lần 1:

***1***, 14, 10, 6, ***2***, 15, 11, 7, ***3***, 16, 12, 8, ***4***, 17, 13, 9, ***5***

Lần 2:

1, 14, 10, ***6***, 2, 15, 11, ***7***, 3, 16, 12, ***8***, 4, 17, 13, ***9***, 5

Lần 3:

1, 14, ***10***, 6, 2, 15, ***11***, 7, 3, 16, ***12***, 8, 4, 17, ***13***, 9, 5

Lần 4:

1, ***14***, 10, 6, 2, ***15***, 11, 7, 3, ***16***, 12, 8, 4, ***17***, 13, 9, 5

Như vậy sau bốn lần thứ tự các sector đọc được vẫn là từ 1 đến 17

# Kết Luận

Trên đây là những nghiên cứu của nhóm 5 về quản lý bộ nhớ ngoài trong Window. Qua bài tập lớn này đã phần nào chỉ ra được cấu tạo của bộ nhớ ngoài cũng các phương thức lập lịch cho bộ nhớ ngoài. Hi vọng là từ đó chúng ta có thể lựa chọn cách lập lịch cũng như quản lý bộ nhớ ngoài sao cho tối ưu nhất. Chúng em xin chân thành cảm ơn thầy Nguyễn Tuấn Tú đã giúp đỡ chúng em trong quá trình làm bài tập lớn này. Chúng em rất mong nhận được những nhận xét và được chỉ ra những thiếu xót, những chỗ mà chúng em chưa làm tốt. Nhóm em xin cảm ơn!