

Hiệu suất là một lớp phần mềm ở giữa phần cứng và các chương trình ứng dụng/người dùng. Áo hóa các thành phần phần cứng nhằm giúp việc giao tiếp dễ dàng và an toàn, quản lý việc sử dụng chia sẻ tài nguyên máy tính của các chương trình/người dùng: công bằng và hiệu quả, cung cấp một tập hợp thư viện để đơn giản hóa việc phát triển ứng dụng.

Hệ thống xử lý theo lô

Máy tính mainframe: cung cấp khả năng tính toán

- Làm sao chia sẻ thiết bị đất tiền giữa nhiều người dùng?
 - Đăng ký giờ sử dụng
 - Đưa chương trình cho người sử dụng máy tính
 - Hoặc chạy chương trình và in kết quả trả lại cho bạn
- Thông thường, chương trình nằm trong các thẻ đọc (card) được phân cách bởi các thẻ điều khiển (control card). Đầu đọc thẻ (card reader) sẽ đọc dữ liệu trên các thẻ này
- Hệ điều hành (HĐH) được phát triển để thực hiện các công việc một cách tuân tự

HĐH luôn nằm trong bộ nhớ (Quản lý thường trú – Resident monitor)

Quản lý không gian đĩa trống

- Bit vector (Bit map)
 - Mỗi block được biểu diễn bằng 1 bit
 - Bit vector tồn không gian đĩa
- Danh sách liên kết
- Chi phí duyệt danh sách cao
- Không tồn 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.

Tổ chức hệ thống tập tin trong bộ nhớ chính

- 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 tới volume tương ứng
 - System-wide open-file Table – Danh sách các tập tin đang được mở trên hệ thống
 - Con trỏ tới 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 tới tập tin đang mở tương ứng trong system-wide open-file table

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 bẩn sau của Boot sector

Fragmentation

- Một bảng FAT gọi là bộ phân mảnh nếu xảy ra ít nhất một trong 2 điều kiện sau:
 - 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

Tiến trình

Tiến trình là một chương trình đang được thực thi
Một tiến trình cần sử dụng các tài nguyên: CPU, bộ nhớ, tập tin, thiết bị nhập xuất để hoàn tất công việc của nó

Tạo tiến trình

- Khởi động hệ thống
- Người dùng kích hoạt một chương trình
- Một tiến trình tạo một tiến trình khác
 - Unix/Linux: exec(), fork()
 - Windows: CreateProcess()
 - Cây tiến trình

Đa chương

- CPU sẽ rảnh mỗi khi chương trình thực thi cần giao tiếp với thiết bị ngoại vi
- Yếu tố: khi một chương trình tương tác với thiết bị ngoại vi thì CPU sẽ phục vụ chương trình khác
- Hệ thống đa chương (Multiprogrammed system) ra đời
- Nạp đồng thời nhiều chương trình vào đĩa (sau này là vào bộ nhớ RAM)
- Chuyển sang phục vụ chương trình khác nếu chương trình hiện thời đang tương tác với thiết bị ngoại vi
- Thiết bị ngoại vi thường chậm hơn đĩa (sau này là bộ nhớ RAM)
- Đặc điểm của hệ thống đa chương
- Đồng thời thực hiện nhập/xuất của chương trình này và tính toán cho chương trình khác
- Tiêu chí: bộ xử lý luôn trong tình trạng làm việc
- Phai biết khi nào công việc nhập/xuất xong: ngắt vs. polling

Chia sẻ thời gian

Nối nhiều thiết bị đầu cuối đều đến một máy tính
Điều phối sử dụng máy tính cho nhiều người dùng
Chuyển đổi phục vụ giữa các chương trình người dùng sao cho dù nhanh để người sử dụng có thể tương tác với chương trình trong khi chúng đang chạy (tạo cảm giác mỗi người dùng đang dùng máy riêng của mình)

Cấu trúc vật lý của đĩa từ

- Gồm nhiều lớp hình tròn, mỗi lớp phủ từ 1 hoặc cả 2 mặt (side)
- Mỗi mặt có đường kính 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)

Cơ chế đọc đĩa từ

Access time = Seek time + Rotational time + Read time

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 tới 1 cấu trúc chứa các thuộc tính.

Tổ chức tập tin

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

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

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

- Chứa thông tin hỗ trợ cho việc xác định tổng số cluster trống & tìm kiếm cluster trống được hiệu quả
- Chứa một sector bẩn sau của Boot sector

Boot sector	File allocation table 1	File allocation table 2 (duplicate)	Root directory	Other directories and all files

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	Thuật ngữ trạng thái (0.0.A.D.V.S.I.R)
C	1	Danh riêng
D	3	Giờ tạo (miili giay:7; giay:6; phut:6; gio:5)
10	2	Ngày tạo (ngay: 5; tháng: 4; năm: 1980: 7)
12	2	Ngày truy cập gần nhất (lưu nhu trên)
14	2	Cluster bắt đầu - phần Word (2Byte) cao
16	2	Giờ sửa gần nhất (giay:2;5; phut:6; gio:5)
18	2	Ngày cập nhật gần nhất (lưu nhu trên)
1A	2	Cluster bắt đầu - phần Word thấp
1C	4	Kích thước của phần nội dung tập tin

Entry chính của RDET

Quá trình khởi động máy tính (booting)

CPU thực thi lệnh từ địa chỉ cố định biết trước (boot ROM)
Firmware nạp boot loader
Boot loader nạp HĐH

Truy xuất mức vật lý trên đĩa từ

- 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)**

Chuyển đổi sector vật lý ↔ sector logic

$$\begin{aligned} (\text{vật lý} \rightarrow \text{logic}) l &= t * st * hd + h * st + s - 1 \\ (\text{logic} \rightarrow \text{vật lý}) s &= (l \bmod st) + 1 \\ (\text{logic} \rightarrow \text{vật lý}) t &= l \div (st * hd) \\ (\text{logic} \rightarrow \text{vật lý}) h &= (l \bmod st) \bmod hd \end{aligned}$$

l: chỉ số sector logic
st: track/st, hd: số side
h: chỉ số header
st: st/track, s: chỉ số st

Master Boot Record (MBR)

- Đoạn chương trình để giúp khởi động hệ thống

– Bảng mô tả thông tin các phân vùng logic

– TYPE-ID = 0x07 : Windows

– TYPE-ID = 0x83 : Linux

– TYPE-ID = 0x00 : Không sử dụng.

Quá trình khởi động hệ thống từ đĩa

- POST (Power-On Self-Test)

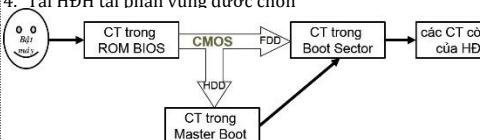
- Tải MBR để đọc thông tin bảng phân vùng.

Tim 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

- Chuyển quyền điều khiển về cho doan mã chương trình nằm trong Boot Sector của phân vùng được chọn

- Tải HĐH tại phân vùng được chọn



RDET

Nằm trên vùng hệ thống (FAT12 & FAT16) hoặc nằm trên vùng dữ liệu (FAT32)

Gồm một dãy các phần tử (goi 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 phu: chỉ chứa tên của tập tin

Entry phu

Offset 0 Sô byte 1 Ý nghĩa

1 A (10d) 5 ký tự Unicode – Bảng mã UTF16

B (11d) 1 Dấu hiệu nhận biết (luôn là 0Fh)

E (14d) C (12d) 6 ký tự kế tiếp

IC (28d) 4 2 ký tự kế tiếp

Bảng thư mục con

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

Mục tiêu điều phối

Mục tiêu chung

-Công bằng sử dụng CPU

-Cân bằng sử dụng các thành phần của hệ thống

Hệ thống theo lô

-Tối ưu throughput

-Giảm thiểu turnaround time: T_{quit} – T_{arrive}

Lời gọi hệ thống

Là tập các hàm cơ bản nhất của hệ điều hành để phục vụ các yêu cầu từ các chương trình người dùng

Ví dụ, chuỗi các lời gọi hệ thống được thực hiện để sao chép nội dung của một tập tin sang một tập tin khác

Các thành phần của hệ điều hành

Quản lý bộ xử lý – Quản lý tiến trình

Quản lý bộ nhớ

Quản lý nhập xuất

Quản lý lưu trữ – Hệ thống tập tin

Hệ thống bảo vệ và bảo mật

Tổ chức logic của đĩa từ

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 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)

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

Boot sector của FAT12 và FAT16

Offs et	Sô byte	Ý nghĩa
0	3	Lệnh nhảy đến đầu đoạn mã Boot
3	8	Tên công ty/version của HĐH
B	2	Sô byte của sector, thường là 512
D	1	Sô sector của cluster (Sc)
E	2	Sô sector trước bảng FAT (S _F) (vùng bootsector)
10	1	Sô lượng bảng FAT (N _F), thường là 2
11	2	Sô entry của RDET (S _R), thường là 512 (FAT16) 32byte/entry
13	2	Sô sector của volume (S _v), băng 0 nếu S _v > 65535
15	1	Kí hiệu loại volume
16	2	Sô sector của FAT (S _F)
18	2	Sô sector của track
1A	2	Sô lượng đầu đọc (side)
1C	4	Khoảng cách từ nơi mở file vol đến đầu vol
20	4	Kích thước volume (nếu số 2 byte tại 13h là 0)
24	1	Ký hiệu vật lý của đĩa chứa vol(0: mềm, 80h: cứng)
25	1	Danh rieng
26	1	Ký hiệu nhân 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/ MasterBoot (AA55h)

Entry phu

Offset 0 Sô bytes 1 Ý nghĩa

1 A (10d) 5 ký tự Unicode – Bảng mã UTF16

B (11d) 1 Dấu hiệu nhận biết (luôn là 0Fh)

E (14d) C (12d) 6 ký tự kế tiếp

IC (28d) 4 2 ký tự kế tiếp

Truy xuất các phần tử trên bảng FAT

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

$$i = k * \text{kích thước phần tử FAT}$$

Lựa chọn tiến trình

Tiêu chí lựa chọn

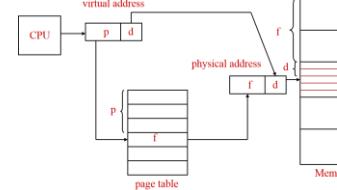
– Chọn tiến trình vào RQ trước

– Chọn tiến trình có độ ưu tiên cao hơn

Thời điểm lựa chọn

– Điều phối độc quyền (non-preemptive scheduling): một khi tiến trình đang ở trạng thái Running, nó sẽ tiếp tục sử dụng CPU cho đến khi kết thúc hoặc bị block vì I/O hay các dịch vụ của hệ thống (độc chiếm CPU)

• P_{cur} kết thúc

<ul style="list-style-type: none"> • Unix/ Linux: các tiến trình chia con có mối quan hệ chép ché • Windows: các tiến trình chia con độc lập với nhau <p>Dùng tiến trình</p> <ul style="list-style-type: none"> - Xử lý xong lệnh cuối cùng hay gọi lệnh kết thúc • Unix / Linux: exit() • Windows: ExitProcess() - Một tiến trình yêu cầu dùng một tiến trình khác • Unix / Linux: kill() • Windows: TerminateProcess() - Do lỗi chương trình 	<p>Bảng FAT</p> <ul style="list-style-type: none"> 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 Kích thước mỗi phần tử FAT phụ thuộc vào loại FAT <ul style="list-style-type: none"> -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 Phản ứng từ k trên bảng FAT (dánh số từ 0) cho biết trạng thái của cluster thứ k trên vùng dữ liệu (dánh số từ 2) → 2 phản ứng đầu của bảng FAT không dùng 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 	<p>Tận dụng CPU</p> <ul style="list-style-type: none"> Hệ thống tương tác Giảm thiểu thời gian chờ (Tối ưu thời gian hồi đáp): Tin ReadyQueue Cân đối mong muốn của người dùng Hệ thống thời gian thực Thời hạn hoàn thành công việc 	<ul style="list-style-type: none"> • P_{cur}: running -> blocked - Điều phối không độc quyền (preemptive scheduling): ngoài thời điểm lựa chọn như điều phối độc quyền, tiến trình đang sử dụng CPU có thể bị ngắt (chuyển sang trạng thái Ready) khi hết thời gian qui định hoặc có tiến trình có độ ưu tiên hơn vào ReadyQueue • Q: blocked / new -> ready 																																																																																															
<p>Round Robin</p> <p>Mỗi tiến trình chỉ sử dụng một lượng q cho mỗi lần sử dụng CPU</p> <p>Tiêu chí lựa chọn tiến trình</p> <ul style="list-style-type: none"> - Thứ tự vào hàng đợi Ready Queue <p>Thời điểm lựa chọn tiến trình</p> <ul style="list-style-type: none"> - Không độc quyền (không có độ ưu tiên) <p>Loại bỏ hiện tượng độc chiếm CPU</p> <p>Phù hợp với hệ thống tương tác người dùng</p> <p>Hiệu quả ? Phụ thuộc vào việc lựa chọn quantum q</p> <ul style="list-style-type: none"> - q quá lớn => FCFS (giảm tính tương tác) - q quá nhỏ => chủ yếu thực hiện chuyển đổi ngữ cảnh (context switching) <p>- Thời gian $q = 10-100$ milliseconds</p>	<p>PCB - Process Control Block</p> <table border="1"> <thead> <tr> <th>pid</th> <th>State (State, details)</th> <th>Context (IP, Mem, Files...)</th> <th>Relatives (Dad, children)</th> <th>Scheduling statistic</th> </tr> </thead> </table> <p>Định danh (Process ID)</p> <p>Trạng thái tiến trình</p> <ul style="list-style-type: none"> • Trạng thái CPU • Bộ xử lý (cho máy nhiều CPU) • Bộ nhớ chính • Tài nguyên sử dụng / tạo lập • Thông tin giao tiếp • Tiến trình cha, tiến trình con • Độ ưu tiên <p>Thông tin thống kê</p>	pid	State (State, details)	Context (IP, Mem, Files...)	Relatives (Dad, children)	Scheduling statistic	<p>Boot Sector của FAT32</p> <table border="1"> <thead> <tr> <th>Offset</th> <th>Số byte</th> <th>Nội dung</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>3</td> <td>Jump Code: lệnh nhảy qua vùng thông số (như FAT)</td> </tr> <tr> <td>3</td> <td>8</td> <td>OEM_ID: nơi sản xuất - version, thường là "MSWIN4.1"</td> </tr> <tr> <td>B</td> <td>2</td> <td>Số byte trên Sector, thường là 512 (như FAT)</td> </tr> <tr> <td>D</td> <td>1</td> <td>S: số sector trên cluster (như FAT)</td> </tr> <tr> <td>E</td> <td>2</td> <td>S: số sector thuộc vùng Boot Sector (như FAT)</td> </tr> <tr> <td>10</td> <td>1</td> <td>Nr: số bảng FAT, thường là 2 (như FAT)</td> </tr> <tr> <td>11</td> <td>2</td> <td>Không dùng thường là 0 (số entry của RDET - với FAT)</td> </tr> <tr> <td>13</td> <td>2</td> <td>Không dùng thường là 0 (số sector của vol - với FAT)</td> </tr> <tr> <td>15</td> <td>1</td> <td>Loại thiết bị (F8h nếu là đĩa cứng - như FAT)</td> </tr> <tr> <td>16</td> <td>2</td> <td>Không dùng, thường là 0 (số sector của bảng FAT - với FAT)</td> </tr> <tr> <td>18</td> <td>2</td> <td>Số sector của track (như FAT)</td> </tr> <tr> <td>1A</td> <td>2</td> <td>Số lượng đầu đọc (như FAT)</td> </tr> <tr> <td>1C</td> <td>4</td> <td>Khoảng cách từ nơi mở tâ đến đầu vol (như FAT)</td> </tr> <tr> <td>20</td> <td>4</td> <td>Sv: kích thước volume (như FAT)</td> </tr> <tr> <td>24</td> <td>4</td> <td>Sf: kích thước mỗi bảng FAT</td> </tr> <tr> <td>28</td> <td>2</td> <td>Bit 8 bắt: chỉ ghi vào bảng FAT active (có chỉ số là 4 bit đầu)</td> </tr> <tr> <td>2A</td> <td>2</td> <td>Version của FAT32 trên vol này</td> </tr> <tr> <td>2C</td> <td>4</td> <td>Cluster bắt đầu của RDET</td> </tr> <tr> <td>30</td> <td>2</td> <td>Sector chứa thông tin phu (về cluster trống) thường là 1</td> </tr> <tr> <td>32</td> <td>2</td> <td>Sector chứa bản lề của Boot Sector</td> </tr> <tr> <td>34</td> <td>C</td> <td>Danh rieng (cho các phiên bản sau)</td> </tr> <tr> <td>40</td> <td>1</td> <td>Ký hiệu vật lý cả đĩa chứa vol (0: mềm, 80h: cứng)</td> </tr> <tr> <td>41</td> <td>1</td> <td>Danh rieng</td> </tr> <tr> <td>42</td> <td>1</td> <td>Ký hiệu nhận diện HDH</td> </tr> <tr> <td>43</td> <td>4</td> <td>SerialNumber của volume</td> </tr> <tr> <td>47</td> <td>B</td> <td>Volume Label</td> </tr> <tr> <td>52</td> <td>8</td> <td>Loại FAT, là chuỗi "FAT32"</td> </tr> <tr> <td>5A</td> <td>1A4</td> <td>Đoạn chương trình khởi tạo & nạp HDH khi khởi động máy</td> </tr> <tr> <td>1FE</td> <td>2</td> <td>Đầu hiệu kết thúc BootSector/MasterBoot (luôn là AA55h)</td> </tr> </tbody> </table>	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: số sector trên cluster (như FAT)	E	2	S: số sector thuộc vùng Boot Sector (như FAT)	10	1	Nr: 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 mở tâ đến đầu vol (như FAT)	20	4	Sv: kích thước volume (như FAT)	24	4	Sf: kích thước mỗi bảng FAT	28	2	Bit 8 bắt: 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 phu (về cluster trống) thường là 1	32	2	Sector chứa bản lề của Boot Sector	34	C	Danh rieng (cho các phiên bản sau)	40	1	Ký hiệu vật lý cả đĩa chứa vol (0: mềm, 80h: cứng)	41	1	Danh rieng	42	1	Ký hiệu nhận diện HDH	43	4	SerialNumber của volume	47	B	Volume Label	52	8	Loại FAT, là chuỗi "FAT32"	5A	1A4	Đoạn chương trình khởi tạo & nạp HDH khi khởi động máy	1FE	2	Đầu hiệu kết thúc BootSector/MasterBoot (luôn là AA55h)	<p>FCFS</p> <p>Tiêu chí lựa chọn tiến trình</p> <ul style="list-style-type: none"> - Thứ tự vào hàng đợi Ready Queue <p>Thời điểm lựa chọn tiến trình</p> <ul style="list-style-type: none"> - Độc quyền <p>Đơn giản, dễ cài đặt</p> <p>Chịu đựng hiện tượng tích lũy thời gian chờ</p> <p>Tiến trình có thời gian xử lý ngắn đợi tiến trình có thời gian xử lý dài</p> <p>Ưu tiên tiến trình cpu-bounded</p> <p>Có thể xảy ra tình trạng độc chiếm CPU</p> <p>Không phù hợp với hệ thống tương tác người dùng</p>
pid	State (State, details)	Context (IP, Mem, Files...)	Relatives (Dad, children)	Scheduling statistic																																																																																														
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: số sector trên cluster (như FAT)																																																																																																
E	2	S: số sector thuộc vùng Boot Sector (như FAT)																																																																																																
10	1	Nr: 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 mở tâ đến đầu vol (như FAT)																																																																																																
20	4	Sv: kích thước volume (như FAT)																																																																																																
24	4	Sf: kích thước mỗi bảng FAT																																																																																																
28	2	Bit 8 bắt: 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 phu (về cluster trống) thường là 1																																																																																																
32	2	Sector chứa bản lề của Boot Sector																																																																																																
34	C	Danh rieng (cho các phiên bản sau)																																																																																																
40	1	Ký hiệu vật lý cả đĩa chứa vol (0: mềm, 80h: cứng)																																																																																																
41	1	Danh rieng																																																																																																
42	1	Ký hiệu nhận diện HDH																																																																																																
43	4	SerialNumber của volume																																																																																																
47	B	Volume Label																																																																																																
52	8	Loại FAT, là chuỗi "FAT32"																																																																																																
5A	1A4	Đoạn chương trình khởi tạo & nạp HDH khi khởi động máy																																																																																																
1FE	2	Đầu hiệu kết thúc BootSector/MasterBoot (luôn là AA55h)																																																																																																
<p>SJF</p> <p>Là một dạng độ ưu tiên đặt biệt với độ ưu tiên $p_i = \text{time_remain}(P_i)$</p> <p>Tối ưu thời gian chờ</p> <p>Ưu lượng - sử dụng thời gian sử dụng CPU ngay trước, dùng qui luật trung bình giảm theo hàm mũ.</p> $r_{n+1} = at_n + (1-\alpha)r_n$	<p>Tranh đoạt điều khiển</p> <p>Kết quả thực hiện tiến trình phụ thuộc vào kết quả điều phối</p> <p>Cùng input, không chắc cùng output</p> <p>Không debug lỗi sai trong xử lý đồng hành</p> <p>Xử lý</p> <p>Lý do xảy ra Race condition ? Bad interleavings : một tiến trình "xen vào" quá trình truy xuất tài nguyên của một tiến trình khác</p> <p>Race Condition → Nhu cầu "độc quyền truy xuất" (Mutual Exclusion)</p> <p>Các tiến trình phối hợp hoạt động → Nhu cầu "hò hẹn" (Rendez-vous)</p> <p>Thực hiện đồng bộ hóa :</p> <ul style="list-style-type: none"> - Lập trình viên đề xuất chiến lược: các tiến trình liên quan trong bài toán phải tôn trọng các luật đồng bộ - Giải pháp sử dụng các cơ chế đồng bộ: do lập trình viên / phần cứng / HDH / NNLT cung cấp. 	<p>Giải pháp đồng bộ hóa</p> <p>Một phương pháp giải quyết tốt bài toán đồng bộ hóa cần thỏa mãn 4 điều kiện sau:</p> <ul style="list-style-type: none"> • Mutual Exclusion : Không có hai tiến trình cùng ở trong miền găng cùng lúc. • Progress : Một tiến trình tam dừng bên ngoài miền găng không được ngăn cản các tiến trình khác vào miền găng • Bounded Waiting : Không có tiến trình nào phải chờ vô hạn để được vào miền găng. • Không có giả thiết nào đặt ra cho sự liên hệ về tốc độ của các tiến trình, cũng như về số lượng bộ xử lý trong hệ thống <p>Monitor (Sleep & Wakeup)</p> <p>Hỗ trợ cung các chức năng như semaphore, dễ sử dụng và kiểm soát hơn, tự động đảm bảo mutual exclusion, hỗ trợ synchronization với các condition variables.</p> <p>Message (Sleep & Wakeup)</p> <p>Được hỗ trợ bởi HDH, đồng bộ hóa trên môi trường phân tán, 2 primitive Send&Receive (cài đặt theo module blocking)</p>	<p>Miền găng (CriticalSection)</p> <p>Miền găng (CS) là đoạn chương trình có khả năng gây ra hiện tượng race condition</p> <p>Hỗ trợ Atomicity : Cần bảo đảm tính "độc quyền truy xuất" (Mutual Exclusion) cho miền găng (CS)</p> <p>Giải pháp kiểm tra luân phiên (Busy-waiting phần mềm)</p> <p>Chi dành cho 2 tiến trình, bảo đảm độc quyền truy xuất, không đảm bảo progress.</p> <p>Bộ nhớ ảo (Virtual Memory)</p> <p>Để giả lập như chúng ta có bộ nhớ lớn hơn để thực thi chương trình mà yêu cầu bộ nhớ lớn hơn bộ nhớ ta đang có sẵn. Một trang là một đơn vị của bộ nhớ ảo. HDH ánh xạ giữa các trang của VM và bộ nhớ vật lý. ($\text{frame(vật lý)} \rightarrow \text{page(logic)}$).</p>																																																																																															
<p>Nhận xét chung giải pháp trong Busy-waiting</p> <p>Sử dụng CPU không hiệu quả: liên tục kiểm tra điều kiện chờ vào CS. Phần mềm: không cần hỗ trợ của hệ thống, dễ sai, khó mở rộng. Phần cứng (cầm ngắt và TSL): cần được hỗ trợ của cơ chế phần cứng, dễ mở rộng cho N tiến trình.</p>	<p>Semaphore (Sleep & Wakeup)</p> <p>Có 1 g.tri Semaphore s chỉ được thao tác bởi 2 primitives: up&down</p> <p>Semaphore được xem như là 1 resource: các tiến trình yêu cầu semaphore: gọi down, nếu không hoàn tất được down thì chưa cấp resource và đưa vào s.L</p> <p>Down(s){ s.value--; if(s.value<0){add(P,s.L);Sleep();}}</p> <p>Up(s){s.value++;if(s.value<=0){remove(P,s.L);WakeUp(P,);}}</p>	<p>Giải pháp phần mềm</p> <p>Đáp ứng được 3 điều kiện, không thể mở rộng cho n tiến trình.</p> <p>Quản lý bộ nhớ của HDH</p> <p>Bên trong: các bộ nhớ vật lý trông "giống nhau"</p> <p>C.p.h.t: t.trình có thể được nạp lên tại bất kì địa chỉ vật lý nào</p> <p>Bảo vệ: 1 tiến trình không thể truy cập vùng nhớ của tiến trình khác</p> <p>Chia sẻ: cho phép chia sẻ bộ nhớ vật lý (phải cài đặt điều khiển)</p> <p>Phản mãnh</p> <p>Ngoại vi (External Fragmentation) - tổng bộ nhớ trống thỏa yêu cầu, nhưng không liên tục</p> <p>Nội vi (Internal Fragmentation) - mỗi block được cấp phát lớn hơn yêu cầu bộ nhớ một ít</p> <p>Giải pháp phản mãnh ngoại vi: kết hợp</p> <p>Chuyển các vùng trống thành một khối bộ nhớ liên tục</p> <p>Chỉ thực hiện được nếu HDH hỗ trợ biến dịch địa chỉ trong thời gian thực thi</p>	<p>Giải pháp biến cờ hiệu (Busy-Waiting phần mềm)</p> <p>Có thể mở rộng cho N tiến trình.</p> <p>Không đảm bảo truy xuất độc quyền vì bản thân code kiểm tra và đánh quyền cũng là CS.</p> <p>Peterson (Busy-waiting phần mềm)</p> <p>Đáp ứng được 3 điều kiện, không thể mở rộng cho n tiến trình.</p> <p>Quản lý bộ nhớ của HDH</p> <p>Bên trong: các bộ nhớ vật lý trông "giống nhau"</p> <p>C.p.h.t: t.trình có thể được nạp lên tại bất kì địa chỉ vật lý nào</p> <p>Bảo vệ: 1 tiến trình không thể truy cập vùng nhớ của tiến trình khác</p> <p>Chia sẻ: cho phép chia sẻ bộ nhớ vật lý (phải cài đặt điều khiển)</p> <p>Phản mãnh</p> <p>Ngoại vi (External Fragmentation) - tổng bộ nhớ trống thỏa yêu cầu, nhưng không liên tục</p> <p>Nội vi (Internal Fragmentation) - mỗi block được cấp phát lớn hơn yêu cầu bộ nhớ một ít</p> <p>Giải pháp phản mãnh ngoại vi: kết hợp</p> <p>Chuyển các vùng trống thành một khối bộ nhớ liên tục</p> <p>Chỉ thực hiện được nếu HDH hỗ trợ biến dịch địa chỉ trong thời gian thực thi</p> <p>Lỗi trang (Page Fault)</p> <p>Trap lỗi trang</p> <p>Kiểm tra có phải truy xuất hợp lệ (có trang vật lý đúng)</p> <p>Tim 1 frame bộ nhớ trống</p> <p>Đọc trang cần thiết từ bộ nhớ phụ (đĩa)</p> <p>Đổi valid bit của trang thành v (hợp lệ)</p> <p>Bắt đầu lại lệnh bị ngắt bởi trap.</p> <p>Thay trang</p> <p>Chọn frame để thay thế (nan nhản) (FIFO, LRU, Clock, Nth)</p> <p>Lưu trang nạn nhân vào ổ đĩa</p> <p>Cập nhật lại bảng trang (trang nạn nhân thành invalid)</p> <p>Đọc trang cần thiết vào frame vừa chọn</p> <p>Cập nhật lại bảng trang (trang vừa thay thế là valid)</p> <p>Bắt đầu lại lệnh đã gây ra lỗi trang</p> <p>Working Set</p> <p>Là một tập các trang được sử dụng trong khoảng thời gian gần nhất. Kích thước WS có thể thay đổi trong suốt quá trình thực thi, chỉ điều phối cho tiến trình khi mà bộ nhớ dù để nạp WS. Working-set window = số trang được gọi.</p>																																																																																															
<p>Địa mèm 1.44MB (2880sector), 2h/d, 80t/h, 18s/t</p> <p>Thay các giá trị trên và đăng thức $S_B + N_r * S_F + S_R = S_V \Leftrightarrow 2S_F + S_D = 2877$ (sector) (*)</p> <p>$S_D < 2877$ (sector) = 719.25 (cluster) (vi $S_C = 4$ sector).</p> <p>Loại FAT tối ưu nhất là FAT12, vì $S_D < 4079$ (cluster)</p> <p>Giá trị $S_F = 1$ (sector): (*) $\Rightarrow S_D = 2875$ (sector) = 718.75 (cluster)</p> <p>Vùng dữ liệu có 719 cluster, bảng FAT có 721 phản ứng, do đó $S_F = (721*1.5)/512 = 2.1x$ (sector)</p> <p>Bảng FAT phải chiếm 3 sector - mâu thuẫn với giả thiết $S_F = 1$. Vậy kích thước bảng FAT của vol này không thể là 1 sector</p> <p>Giá trị $S_F = 2$ (sector): tương tự, ta vẫn thấy mâu thuẫn, tức kích thước bảng FAT phải lớn hơn 2 sector.</p> <p>Giá trị $S_F = 3$ (sector): (*) $\Rightarrow S_D = 2871$ (sector) = 717.75 (cluster).</p> <p>Vùng dữ liệu có 718 cluster, bảng FAT có 720 phản ứng, do đó $S_F = (720*1.5)/512 = 2.1x$ (sector)</p> <p>Bảng FAT phải chiếm 3 sector - phù hợp với giả thiết $S_F = 3$.</p> <p>Vậy kích thước bảng FAT của vol này là 3 sector.</p>	<p>Kiến trúc phân đoạn</p> <p>Địa chỉ logic bao gồm 2 tham số: $<\text{segment-number}, \text{offset}>$</p> <p>Segment table - ánh xạ địa chỉ vật lý; mỗi mục tin trên bảng lưu: base - lưu địa chỉ vật lý bắt đầu trên bộ nhớ.</p> <p>limit - xác định chiều dài của đoạn.</p> <p>Segment-table base register (STBR) lưu vị trí của segment table trên bộ nhớ.</p> <p>Segment-table length register (STLR) lưu số segment được sử dụng bởi người dùng; segment s là hợp lệ nếu $s < \text{STLR}$.</p> <p>Chia lại vùng nhớ: động, thông qua segment table.</p> <p>Chia sẻ: chia sẻ các đoạn, cùng số segment.</p> <p>Cấp phát: first fit/best fit, phân mảnh ngoài.</p> <p>Bảo vệ: mỗi mục tin trong segment table có thêm valid bit, quyền(r,w,exec).</p> <p>Bộ nhớ chính 320MB, dc logic 20bit, pg size 8K. Dc offset 8K = 2^{13} → 13bit. Số frame 320M/8K=40960frames. Page logic $2^{20}/8K=2^{12}$ page. Kích thước không gian bộ nhớ là 2^{20}byte.</p> <p>Địa chỉ 20030 sang $<p, d>$, p = 20030 div 8K, d = 20030 mod 8K. Quá trình truy xuất bộ nhớ lần lượt: P1.200, P1.399, P2.400, P1.200. Hệ thống có 4 khung trang, page size = 200. Chuỗi truy xuất trang: P1.1, P1.2, P1.3, P1.0 (chia địa chỉ cho 200). Cấp phát theo chiến lược tối ưu có 4 page fault.</p>	<p>Phân đoạn</p> <p>Các đoạn có kích thước khác nhau</p> <p>Biên dịch chỉ dựa vào các thanh ghi (base, size, state) - bảng p.đoạn</p> <p>State: valid/invalid, access permission, reference bit, modified bit</p> <p>Phân trang</p> <p>Các trang có kích thước cố định</p> <p>Bộ nhớ vật lý tương ứng với trang gọi là page frame</p> <p>Chuyển đổi chỉ thông qua bảng trang, đc đánh chỉ mục bảng page number</p> <p>Mỗi mục tin trong bảng trang lưu một con số đại diện page frame mà trang đó ánh xạ tới và trạng thái của trang trong bộ nhớ</p> <p>State: valid/invalid, access permission, reference bit, modified bit, caching</p>  <p>TLB (Translation Lookaside Buffer)</p> <p>Cache cho các mẩu tin trong bảng trang, mỗi mẩu tin của TLB chứa 1 page number & một mẩu tin của bảng trang tương ứng.</p> <p>Nếu TLB Miss → đòng trên page table.</p>	<p>Working Set</p> <p>Là một tập các trang được sử dụng trong khoảng thời gian gần nhất. Kích thước WS có thể thay đổi trong suốt quá trình thực thi, chỉ điều phối cho tiến trình khi mà bộ nhớ dù để nạp WS. Working-set window = số trang được gọi.</p>																																																																																															