BÀI 5: BỘ NHỚ NGOÀI

Nội dung

- Đĩa từ.
- Hệ thống tệp (file).

Mục tiêu

- Nắm được khái niệm, cấu trúc bộ nhớ ngoài và cấu trúc đĩa từ.
- Biết cách quản lý bộ nhớ ngoài, chức năng hệ thống file.

Thời lượng học

- 8 tiết.

TÌNH HUỐNG DẪN NHẬP

Tình huống

Bộ nhớ trong (bộ nhớ thực) là bộ nhớ ngắn hạn, dung lượng hạn chế. Trong khi nhu cầu lưu trữ các cơ sở dữ liệu đòi hỏi khả năng lưu trữ lâu dài, dung lượng lớn. Phục vụ cho nhu cầu này là bộ nhớ ngoài. Có nhiều công nghệ, thiết bị lưu trữ khác nhau phục vụ cho mục đích này: đĩa từ (đĩa mềm, đĩa cứng), flash, CD-ROM, DVD, … Trong đó, thiết bị được sử dụng phổ biến nhất là đĩa từ.

Câu hỏi

Cấu tạo của đĩa từ? Tại sao truy cập đĩa từ chậm hơn nhiều so với truy cập bộ nhớ thực? Dữ liệu lưu trên đĩa từ được tổ chức thế nào? Làm thế nào để truy cập?

5.1. Đĩa từ

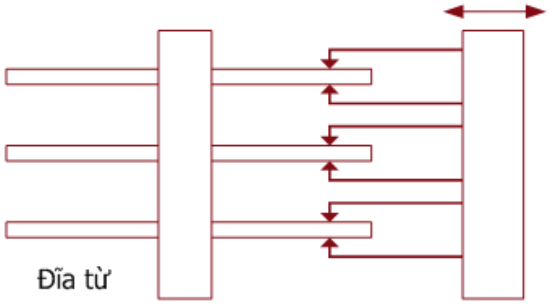
Sự chậm chạp của các hệ thống đa nhiệm thường do việc sử dụng không đúng các thiết bị lưu trữ ngoại vi như đĩa từ, trống từ,.... Trong chương này chúng ta sẽ xem xét một số phương pháp trong việc điều khiển các thiết bị đó.

Chúng ta sẽ phân tích sự làm việc của đĩa từ, xem xét các nguyên nhân dẫn đến làm việc không hiệu quả, phân tích các biện pháp nâng cao hiệu suất – so sánh sự giống nhau, khác nhau cũng như ưu, khuyết điểm của chúng – về phương diện tốc độ.

5.1.1. Cấu tạo đĩa từ

Trên hình 5.1.1a biểu diễn sơ đồ của ổ đĩa cứng với các đầu từ di động. Dữ liệu được ghi trên các mặt đĩa phủ từ, các đĩa này được gắn chặt vào một trục chung và quay với tốc độ rất cao (3600 vòng/ phút– 7200 vòng/ phút).

Việc truy cập dữ liệu (đọc hay ghi) được thực hiện với sự giúp đỡ của các đầu từ đọc/ghi, mỗi mặt đĩa có một đầu từ. Đầu từ chỉ có thể truy cập dữ liệu trên mặt đĩa nằm trực tiếp ngay dưới (trên) nó. Do đó để có thể truy cập đến dữ liệu, vùng đĩa chứa dữ liệu phải dịch chuyển trong quá trình quay sao cho nó nằm trực tiếp dưới đầu từ. Thời gian cần thiết để dịch chuyển (quay) vùng bề mặt đĩa đến dưới đầu từ gọi là 'thời gian trễ' (Latency).

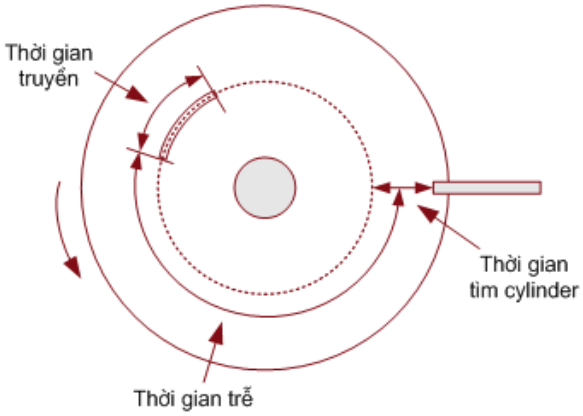


Hình 5.1.1a

Mỗi đầu từ, nếu như nó không dịch chuyển vẽ nên trên bề mặt đĩa (đang quay) đường tròn (Track) trên đó có thể lưu dữ liệu. Tất cả các đầu từ được gắn trên khối định vị. Khối định vị với các đầu từ có thể dịch chuyển theo bán kính các đĩa. Với việc dịch chuyển đầu từ đến vị trí mới, chúng ta có thể truy cập đến nhóm các rãnh (Track) khác nhau.

Nhóm các Track nằm dưới tất cả các đầu từ đọc/ghi trong một vị trí nào đó của khối tạo thành Cylinder. Quá trình dịch chuyển đầu từ đến Cylinder mới gọi là thao tác tìm Cylinder (Seek).

Như thế, để có thể truy cập đến dữ liệu trên đĩa với các đầu từ đọc/ghi nói chung cần thực hiện một vài thao tác (hình 5.1.1b). Trước tiên các đầu từ cần phải được định vị trên Cylinder cần thiết (Seek Cylinder). Sau đó cần phải chờ đến khi điểm bắt đầu của bản ghi đến đúng vị trí dưới đầu từ (tìm bản ghi–gắn với thời gian trễ), tiếp theo là bản thân bản ghi, về nguyên tắc có thể có kích thước tuỳ ý (đến toàn bộ rãng–Track), cần phải đi qua dưới đầu từ (gọi là thời gian truyền– Transmission Time). Bởi vì tất cả các thao tác trên đều gắn với chuyển động cơ học, thời gian tổng cộng cần để truy cập đến



Hình 5.1.1b. ***Các thành phần thời gian khi truy cập***

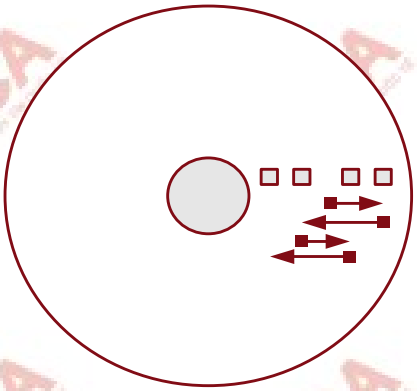
thông tin chiếm tới 0,01–0,1 s. Ngày nay các ổ đĩa cứng có thời gian truy cập ngẫu nhiên trung bình 8–12ms. ta thấy khoảng thời gian đó là rất lớn nếu so sánh với tốc độ của bộ xử lý.

Sự cần thiết phải Planning

Trong các hệ đa nhiệm, cùng lúc có thể có nhiều Process hoạt động và chúng có thể có yêu cầu truy cập đĩa. Bởi các Process thường sinh các yêu cầu nhanh hơn nhiều khả năng phục vụ của các thiết bị ngoại vi, do đó với mỗi thiết bị có một hàng đợi các yêu cầu. Trong một số các hệ thống các yêu cầu này được phục vụ theo nguyên tắc FCFS (First Come – First Served). Nguyên tắc FCFS là cách phục vụ đúng, nhưng khi số yêu cầu lớn thì nó có thể dẫn tới thời gian trễ lớn.

Phương pháp FCFS có đặc điểm là tìm kiếm ngẫu nhiên, trong đó các yêu cầu lần lượt có thể tạo ra các khoảng tìm kiếm Cylinder (Seek Cylinder) dài, từ các Track trong cùng đến các Track ngoài cùng (hình 5.1.1c). Để giảm tối thiểu thời gian tìm kiếm bản ghi, chúng ta cần sắp xếp các yêu cầu theo nguyên tắc nào đó khác với nguyên tắc FCFS. Quá trình đó gọi là Planning công việc với ổ đĩa.

Quá trình Planning cần sự phân tích cẩn thận các yêu cầu để xác định thứ tự phục vụ có hiệu quả nhất. Người ta phải phân tích các liên hệ vị trí của các yêu cầu, sau đó sắp xếp chúng sao cho đảm bảo phục vụ chúng với sự dịch chuyển cơ học ít nhất.



Hình 5.1.1c: ***Tìm kiếm cylinder ngẫu nhiên do nguyên tắc FCFS***

Có hai hướng Planning phổ biến, đó là tối ưu theo thời gian tìm kiếm Cylinder và tối ưu theo thời gian trễ (Latency). Vì thời gian tìm kiếm Cylinder lớn hơn thời gian trễ rất nhiều cho nên phần lớn các thuật toán Planning đạt mục đích giảm tối thiểu thời gian tìm kiếm Cylinder đối với một nhóm yêu cầu nào đó. Giảm thời gian chờ ghi – Thời gian trễ (Latency) thường không ảnh hưởng đáng kể đến đặc tính tốc độ của hệ thống, nếu không tính đến chế độ tải rất lớn.

Với các trường hợp tải nhỏ thì nguyên tắc FCFS có thể chấp nhận, còn với các hệ thống có tải trung bình đến lớn (về số yêu cầu truy cập đĩa) thì Planning có thể đảm bảo đặc tính tốc độ tốt hơn nhiều so với phương pháp FCFS đơn giản.

Các đặc tính đánh giá nguyên tắc Planning

Chúng ta đã thấy rằng nguyên tắc FCFS chấp nhận được trong một số trường hợp. Để đánh giá các nguyên tắc Planning tồn tại một số tiêu chuẩn:

1) Khả năng phục vụ (Throughput).

2) Thời gian trả lời trung bình (Mean Response Time).

3) Sự khác biệt thời gian trả lời (Variance In Response Time).

Rõ ràng rằng các nguyên tắc Planning phải đảm bảo tăng khả năng phục vụ tức là số yêu cầu phục vụ được trong một đơn vị thời gian. Vì các chiến lược Planning cho phép giảm thời gian tìm kiếm nên chúng hoàn toàn có thể nâng cao khả năng phục vụ so với trường hợp dùng phương pháp FCFS. Ngoài ra các chiến lược Planning phải cố gắng làm giảm tối thiểu thời gian trả lời trung bình. Vả lại Planning giảm thời gian tìm kiếm Cylinder cho nên chúng cũng làm rút ngắn thời gian trả lời trung bình so với FCFS.

Các tiêu chuẩn kể trên cố gắng theo hướng cải thiện các chỉ số tốc độ chung của cả hệ thống, và nói chung chúng thực sự làm bức tranh chung tốt hơn dù rằng có thể có một số yêu cầu sẽ bị phục vụ chậm đi đôi chút.

Một trong những chỉ số đánh giá quan trọng nhất là sự khác biệt (Variance) về thời gian trả lời. Nó đánh giá việc một giá trị (ở đây là thời gian phục vụ) cụ thể đối với một phần tử nào đó có thể sai lệch (khác biệt/dao động) bao nhiêu so với giá trị trung bình. Với ý nghĩa đó chúng ta dùng Variance như một chỉ số dự đoán trước– độ khác biệt càng nhỏ thì độ dự đoán trước càng lớn. Chúng ta cần các chiến lược Planning cho phép giảm tối thiểu độ khác biệt variance. Trong trường hợp ngược lại, có thể xảy ra tình huống rằng thời gian phục vụ một số yêu cầu nào đó không thể ước lượng trước. Điều đó không cho phép, ví dụ với hệ thống đăng ký chỗ máy bay khi mà việc trả lời nhanh, chậm ảnh hưởng đến việc bán vé. Nếu như các chiến lược Planning chỉ cố theo hướng tăng khả năng phục vụ (Throughput) mà không đồng thời làm giảm tối thiểu độ dao động (Variance In Response Time), thì nó có thể xử lý chỉ các yêu cầu dễ phục vụ và bỏ qua một số yêu cầu khác.

5.1.2. Tối ưu theo thời gian định vị Track

Chúng ta sẽ phân tích các chiến lược tối ưu thời gian tìm kiếm Cylinder phổ biến nhất:

1) FCFS – các yêu cầu được phục vụ theo thứ tự xuất hiện

2) SSTF (Shortes Seek Time First) – khi đó yêu cần nào gắn với sự dịch chuyển đầu từ ít nhất (từ vị trí hiện thời) được phục vụ trước

3) SCAN (Scan-quét) – đầu từ dịch chuyển đi, về trên bề mặt đĩa và phục vụ tất cả các yêu cầu gặp trên đường. Đầu từ chỉ đổi hướng trong trường hợp không còn yêu cầu nào nằm (ở phía trước) theo hướng hiện thời.

4) C-SCAN (Cycled Scan) – đầu từ chỉ phục vụ theo một hướng dịch chuyển từ ngoài vào trong, khi không còn yêu cầu nào ở phía trước thì nó nhảy trở lại phục vụ yêu cầu nào nằm ngoài cùng và tiếp tục đi vào trong.

5) N-step-SCAN – đầu từ dịch chuyển vào/ra như trong trường hợp SCAN, nhưng tất cả các yêu cầu xuất hiện trong quá trình đang phục vụ theo một hướng nào đó, được nhóm lại theo cách nào đó để chúng có thể được phục vụ hiệu quả nhất trong quá trình phục vụ theo hướng ngược lại.

5) Sơ đồ Eschenbach (Eschenbach Scheme) – đầu từ dịch chuyển lặp lại như trong trường hợp C-SCAN nhưng chiến lược này khác ở một số điểm quan trọng. Khi phục vụ mỗi Cylinder thì chỉ thực hiện truy cập đến một Track mà không để ý đến việc có thể có yêu cầu khác cũng thuộc Cylinder đó. Cũng có phân tích sắp xếp các yêu cầu trên cùng một Cylinder với tham số góc phân bố bản ghi, tuy nhiên nếu hai yêu cầu nằm trên vị trí cắt nhau (có chồng lên nhau) theo phương thẳng đứng thì chỉ có một yêu cầu được phục vụ.

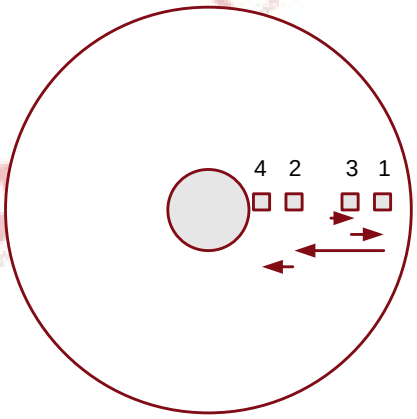
Tối ưu theo FCFS (First Come– First Served)

Theo chiến lược này yêu cầu nào đến trước sẽ được phục vụ trước. Nó đúng ở chỗ, sau khi xuất hiện yêu cầu nào đó – nó sẽ được có chỗ cố định trong hàng. Nó sẽ được phục vụ (không bị loại ra do có các yêu cầu khác được ưu tiên hơn). Nếu các yêu cầu phân bố đều theo bề mặt đĩa thì chiến lược FCFS dẫn tới tìm kiếm ngẫu nhiên. Trong đó bỏ qua các liên hệ vị trí của các yêu cầu đang chờ được phục vụ, và không có bất cứ sự tối ưu nào trong tìm kiếm.

Chiến lược FCFS chấp nhận được nếu hệ thống làm việc với tải nhỏ. Nhưng khi tải tăng lên thì thời gian phục vụ nhanh chóng trở nên quá lâu. Chiến lược FCFS đảm bảo variance không lớn.

Chiến lược SSTF

Khi Planning theo chiến lược SSTF, đầu tiên sẽ phục vụ yêu cầu có khoảng cách nhỏ nhất (do đó có thời gian tìm Cylinder ít nhất) dù yêu cầu đó không phải xuất hiện đầu tiên.



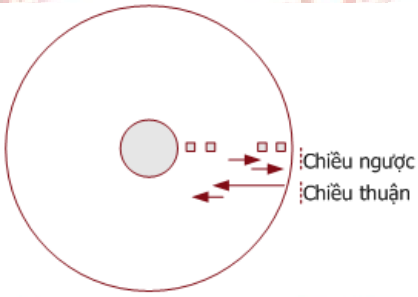
Hình 5.1.2a: ***Tìm kiếm cylinder ngẫu nhiên do nguyên tắc FCFS***

Chiến lược SSTF có đặc điểm variance nhỏ đối với các yêu cầu xác định. Việc truy cập đĩa xuất hiện xu hướng tập trung, kết quả là yêu cầu truy cập các Track trong cùng và ngoài cùng có thể được phục vụ kém hơn nhiều so với các yêu cầu truy cập Track ở giữa.

Chiến lược SSTF đảm bảo khả năng phục vụ lớn hơn FCFS và thời gian trả lời trung bình tốt hơn với tải lớn. Một trong những khuyết điểm của nó là sự tăng độ dao động thời gian trả lời (Variance In Response Time) với các Track trong cùng và ngoài cùng. Nhược điểm của nó có thể bỏ qua trong trường hợp yêu cầu quan trọng nhất là khả năng phục vụ và giảm thời gian trả lời trung bình, ví dụ trong các hệ thống xử lý theo gói.

Chiến lược SCAN

Để giảm variance đối với các Track biên, Denning đã xây dựng chiến lược Scan. Chiến lược này nói chung cũng tương tự như SSTF nếu không tính đến một vấn đề là nó phục vụ yêu cầu có khoảng cách tìm kiếm nhỏ nhất theo một xu hướng xác định (hình 5.1.2b)



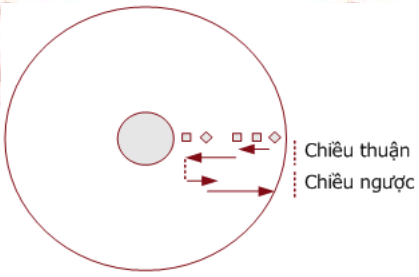
Hình 5.1.2b

Nếu như tại thời điểm hiện tại hướng quét là từ trong ra thì chiến lược SCAN sẽ chọn yêu cầu với khoảng cách nhỏ nhất theo hướng ra ngoài. Trong chiến lược SCAN, đầu từ không đổi hướng chuyển động cho đến khi nó đạt đến Cylinder ngoài cùng hay khi không còn yêu cầu nào chờ theo hướng đó. Nguyên tắc SCAN là cơ bản trong phần lớn các hệ thống có Planning công việc với đĩa từ. Chiến lược SCAN rất giống với SSTF từ quan điểm tăng khả năng phục vụ và giảm thời gian trung bình, nhưng nó giảm đáng kể độ chênh lệch đối với các yêu cầu đến Track biên như của SSTF và đảm bảo variance nhỏ hơn nhiều. Trong chiến lược scan đầu từ, quét từ trong ra ngoài và ngược lại nên nó quét (nằm trên) các Track biên ít hơn (thưa hơn) so với các Track ở giữa, nhưng đó chỉ là nhược điểm nhỏ so với variance trong trường hợp SSTF.

Nguyên lý N–step SCAN

Trên nguyên tắc phương pháp SCAN ở trên có một biến thể gọi là N-step-SCAN. Trong đó đầu từ cũng dịch chuyển đi/về như trong phương pháp SCAN, nhưng trên mỗi chiều dịch chuyển chỉ phục vụ các yêu cầu đã xuất hiện đến lúc bắt đầu dịch chuyển. Các yêu cầu xuất hiện trong thời gian dịch chuyển được nhóm lại và sắp xếp thế nào để chúng có thể được phục vụ tốt nhất trong lần dịch chuyển ngược lại (hình 5.1.2c)

Chiến lược N-step-SCAN đảm bảo chỉ số cao cả về khả năng phục vụ cũng như thời gian trung bình. Nhưng điểm quan trọng nhất của nó là độ chênh lệch (Variance) nhỏ so với khi sử dụng chiến lược SSTF hay SCAN thuần tuý. Chiến lược N-step SCAN loại trừ khả năng yêu cầu bị chờ quá lâu, tình huống thường xuất hiện khi có số lượng lớn yêu cầu đến Cylinder hiện thời. Chiến lược này sẽ lưu các yêu cầu đó để phục vụ vào lúc chuyển động ngược lại.

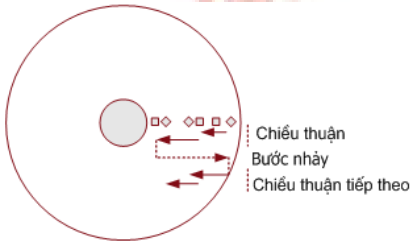


Hình 5.1.2c

Chiến lược C-SCAN

Còn một biến thể của chiến lược scan gọi là C-SCAN. Chiến lược này loại trừ tính chất tăng variance đối với các Track biên.

Theo chiến lược C-SCAN, đầu từ dịch chuyển từ các Cylinder phía ngoài vào trong, ngoài ra phục vụ các yêu cầu theo nguyên tắc thời gian tìm kiếm Cylinder nhỏ nhất. Khi đầu từ hoàn thành chuyển dịch theo chiều thuận, nó sẽ nhảy trở về phục vụ yêu cầu gần Cylinder ngoài cùng nhất và sau đó lại tiếp tục dần vào trong. Chiến lược C-SCAN có thể thực hiện để các yêu cầu xuất hiện trong thời gian đang phục vụ sẽ được phục vụ vào lần sau (hình 5.1.2d). Nhờ đó chiến lược C-SCAN loại bỏ được sự tăng variance với các yêu cầu truy cập Cylinder biên.



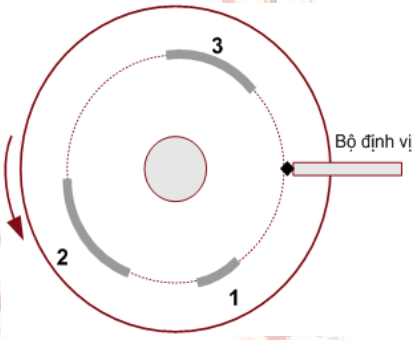
Hình 5.1.2d

Các nghiên cứu cho thấy rằng chiến lược Planning tốt nhất có thể có hai chế độ. Trong chế độ tải thấp, phương pháp tốt nhất là chiến lược SCAN, còn khi tải trung bình và lớn thì kết quả tốt nhất có được khi dùng C–SCAN. Chiến lược này kết hợp với tối ưu theo thời gian trễ (tìm bản ghi) đảm bảo kết quả tốt trong các điều kiện tải rất lớn.

5.1.3. Tối ưu theo thời gian trễ

Trong điều kiện tải lớn, xác suất có lớn hơn một yêu cầu đến cùng Cylinder nào đó tăng lên, do đó việc tối ưu theo thời gian trễ trở nên cần thiết. Tối ưu theo thời gian trễ đã được áp dụng nhiều năm trong các công việc với thiết bị có đầu từ cố định như trống từ.

Tương tự chiến lược SSTF theo hướng tối ưu thời gian tìm Cylinder, trong hướng tối ưu theo thời gian trễ có chiến lược SLTF. Khi bộ định vị (với các đầu từ) nằm trên một Cylinder nào đó với nhiều yêu cầu truy cập các Track khác của Cylinder, chiến lược SLTF phân tích tất cả các yêu cầu và phục vụ yêu cầu với thời gian trễ nhỏ nhất trước tiên (hình 5.1.3a) không phụ thuộc thứ tự yêu cầu nào có trước.



Hình 5.1.3a

Các nghiên cứu cho thấy chiến lược này hoàn toàn gần với kết quả tối ưu theo lý thuyết, ngoài ra việc thực hiện nó không phải là phức tạp.

Các đánh giá hệ thống

Ổ đĩa-tài nguyên quan trọng:

Khi xem xét thấy rằng đĩa cứng là tài nguyên tới hạn (chỗ yếu) của hệ thống, một số kỹ sư khuyên nên tăng dung lượng ổ cứng. Việc này không phải bao giờ cũng giải

quyết vấn đề bởi vì tình trạng critical có thể sinh ra do tần số yêu cầu truy cập đến vùng đĩa nhỏ quá lớn. Nếu như phân tích thấy rằng tình trạng tới hạn (critical) là do vấn đề trên thì có thể áp dụng các chiến lược tối ưu để nâng cao chỉ số tốc độ và loại trừ chỗ yếu đó.

Mức đa nhiệm:

Tải trên đĩa và yêu cầu ngẫu nhiên, thường tăng lên với sự tăng mức độ đa nhiệm. Việc sử dụng các biện pháp tối ưu (Planning) công việc với đĩa có thể không hiệu quả trong các hệ thống với mức độ đa nhiệm thấp. Nhưng với các hệ thống có mức độ đa nhiệm trung bình thì Planning có hiệu quả và nó đạt hiệu quả rõ rệt với các hệ với mức độ đa nhiệm cao (có thể phải xử lý hàng nghìn yêu cầu).

Multdisk subsystem:

Từ cách nhìn kinh tế và module, các ổ đĩa thường được xây dựng sao cho một vài ổ đĩa vật lý làm việc dưới sự điều khiển của một Disk Controller.

Đến lượt mình, Disk Controller lại được nối vào kênh vào/ra đảm bảo sự trang đổi thông tin giữa các ổ đĩa và bộ xử lý. Một kênh có thể phục vụ một vài Disk Controller và đến lượt mình một Disk Controller có thể phục vụ một số ổ đĩa.

Các kênh vào/ra không nối trực tiếp với ổ đĩa. Điều này làm chúng ta phải phân tích cẩn thận chỗ yếu trước khi áp dụng các biện pháp giải quyết. Điểm yếu có thể là do controller không đủ mạnh hay giải thông của kênh không đủ. Việc xác định điểm yếu có thể dễ dàng hơn nhờ các chương trình và thiết bị Diagnostic đặc biệt, kiểm tra hoạt động (các thông số khác nhau) của kênh (Channel) cũng chư Controller. Nếu như điểm yếu là Controller thì chúng ta có thể theo hướng đặt lại cấu hình hệ thống, giảm số lượng ổ đĩa nối vào Controller. Nếu như kênh không đủ giải thông chúng ta có thể đổi một số Controller sang kênh khác hay thêm kênh vào/ra. Như thế, để khắc phục các điểm yếu chúng ta có thể phải thay đổi cấu hình hệ thống.

Để giảm xác suất quá tải kênh vào/ra, trong nhiều hệ thống sử dụng các phương tiện chuyên dụng theo dõi vị trí góc quay của đĩa (RPS-Rotational Position Sensing). Các phương tiện này cho phép giảm thời gian kênh bị bận khi tìm bản ghi trên đĩa. Khi có yêu cầu truy cập đến bản ghi nào đó trên đĩa, RPS giải phóng kênh để kênh thực hiện các thao tác khác cho đến khi bản ghi cần thiết nằm đúng vị trí dưới đầu từ. RPS cho phép kênh có thể phục vụ đồng thời một số yêu cầu, do đó tăng hệ số sử dụng thiết bị.

Phân bố các yêu cầu không đều:

Các nghiên cứu lý thuyết liên quan đến hoạt động của ổ đĩa thường dựa trên một giả thiết là các yêu cầu truy cập đĩa phân bố đồng đều. Kết luận của các nghiên cứu đó có thể không chính xác đối với nhiều hệ thống có đặc điểm các yêu cầu truy cập đĩa phân bố không đều theo bề mặt đĩa. Việc phân bố không đều đó trong một số trường hợp hoàn toàn là bình thường.

Một trong những nguyên nhân phổ biến dẫn tới sự phân bố các yêu cầu không đều là các file lớn liên tục. Khi hệ điều hành chọn chỗ trống để ghi chúng, nó thường ghi dữ liệu lên cùng một Track và khi Track đã đầy thì hệ điều hành chuyển sang ghi lên các Track khác trên cùng Cylinder, và khi Cylinder đầy thì chuyển sang các Cylinder bên cạnh. Như thế khi làm việc với các file liên tục hoàn toàn bình thường khi xuất hiện tình huống các yêu cầu truy cập liên tiếp nói chung sẽ không dẫn tới thao tác tìm kiếm theo Cylinder. Và ngay cả khi phải tìm theo Cylinder thì nó cũng rất ngắn vì đơn giản

sẽ chuyển sang Cylinder ngay cạnh. Trong các tình huống này thì việc tối ưu - Planning nói chung không đem lại lợi ích gì. Ngoài ra các chi phí cho Planning hoàn toàn có thể dẫn đến giảm tốc độ của hệ thống.

Một số hệ thống có kiểm soát tình trạng của bề mặt đĩa và có thể chuyển dữ liệu trên các Track hỏng sang Track tốt khác. Các Track có thể nằm ở các vị trí rất khác nhau và có thể gây ra các dịch chuyển đầu từ thêm (tìm kiếm) vào các thời điểm không ngờ.

Các phương pháp tổ chức file:

Các phương pháp tổ chức file, với tổ chức phức tạp ví dụ dãy chỉ số có thể tạo ra hiện tượng số lượng lớn yêu cầu với thời gian tìm kiếm lâu. Việc phục vụ các yêu cầu trong phương pháp truy cập chỉ số nối tiếp (Index Sequential Access Method–ISAM) có thể gắn với việc thực hiện nhiều lần truy cập đĩa.

Trong một số trường hợp, truy cập bản ghi có thể đòi hỏi truy cập đến index chính, truy cập index của Cylinder và sau đó mới xác định được vị trí bản ghi. Quá trình này có thể dẫn tới nhiều lần tìm kiếm theo Cylinder bởi vì index chính và index của Cylinder thường nằm trên đĩa, nên thời gian trễ trong tìm kiếm có thể khá lớn. Tuy nhiên tổ chức truy cập ISAM thuận tiện cho những người viết phần mềm ứng dụng.

5.2. Hệ thống file

File – là tập hợp dữ liệu, thường được lưu trữ trong các thiết bị lưu trữ ngoại vi như đĩa từ, băng từ,... Với file, ta có thể thực hiện các thao tác như với một đơn vị:

- Open – chuẩn bị file cho truy cập.
- Close – kết thúc truy cập file.
- Create – tạo file mới.
- Copy – tạo bản sao.
- Destroy (Delete) – xoá file.
- Rename – đổi tên file.

Đối với các đơn vị thông tin trong file, thường sử dụng các lệnh:

- Read – đọc dữ liệu từ file.
- Write – ghi các thông tin vào file.
- Insert – chèn thêm thông tin.
- Delete – xoá các đơn vị thông tin.

File system (FS) là một cấu thành của hệ điều hành, có nhiệm vụ điều khiển thao tác với các file trên bộ nhớ ngoài. Nó còn đảm bảo khả năng chia sẻ và an toàn thông tin giữa nhiều người dùng.

Chức năng của hệ thống file:

Hệ thống file phải đảm bảo nhiều chức năng khác nhau liên quan đến điều khiển truy cập, trong đó có:

- Người dùng phải có khả năng tạo, thay đổi, xoá file.
- Cung cấp khả năng chia sẻ file dưới sự điều khiển chặt chẽ.
- Cơ chế chia sẻ file phải xem xét hình thức truy cập cần kiểm soát, ví dụ thao tác đọc, ghi, thay đổi,. ..
- Người dùng phải có khả năng thao tác dễ dàng với cấu trúc file, độc lập với phần cứng.

- Đảm bảo sự trao đổi thông tin giữa các file.
- Cần có các công cụ khắc phục, khôi phục lại thông tin khi có sự cố.
- Với các thông tin quan trọng, cần có các cơ chế kiểm soát truy cập chặt chẽ, tránh các truy cập bất hợp pháp, khả năng mã hoá dữ liệu.
- Hệ thống cần cung cấp giao diện thân thiện với người dùng, cho phép người dùng làm việc với các cấu trúc dữ liệu logic của mình, không cần quan tâm đến các chi tiết vật lý, cụ thể.

5.2.1. Cấu trúc dữ liệu

Tất cả các dữ liệu máy tính có thể xử lý đều tạo thành từ các bit có giá trị 0 hoặc 1. Khi kết hợp các bit thành tổ hợp, ta có thể lưu trữ, thể hiện mọi thông tin.

Mức cao hơn, ta có đơn vị byte là tổ hợp 8 bit. Như thế có thể có 256 giá trị khác nhau của 1 byte, trong số đó có các ký tự (a–z, A–Z), chữ số (0–9), các ký tự đặc biệt,... Phân bố tổ hợp các bit theo ký tự được gọi là bảng mã. Hiện nay có một số bảng mã phổ biến: EBCDIC (Extend Bit Code Decimal For Interchange) thường được sử dụng để biểu diễn thông tin bên trong máy, hệ ASCII (Americal Standard Code For Interchange Information) thông dụng trong các hệ thống truyền thông, ngày nay trong xu thế toàn cầu hoá, mã unicode đang được chấp nhận ngày càng rộng rãi với ưu thế có thể lưu trữ và thể hiện hầu hết các ngôn ngữ trên thế giới.

Ở mức cao hơn, nhóm các ký tự liên quan đến nhau gọi là trường – field, ví dụ trường họ tên sinh viên,... các trường liên quan tạo thành bản ghi, ví dụ bản ghi thông tin sinh viên có các trường họ tên, lớp…

Nhóm các bản ghi tạo thành file và tập hợp thông tin cao nhất được gọi là cơ sở dữ liệu.

Block và Record

Bản ghi vật lý hay khối là đơn vị thông tin thực sự được trao đổi với thiết bị lưu trữ. Còn bản ghi logic là tập hợp thông tin được coi là đơn vị toàn vẹn nhìn từ phía người dùng, ví dụ bản ghi sinh viên.

Với hai khái niệm trên, ta có các quan hệ sau về độ dài tương đối giữa chúng:

- Một khối = một bản ghi.
- Một khối = nhiều bản ghi.
- Một bản ghi = nhiều khối.

Tổ chức file

Tổ chức file là cách phân bố bản ghi của file trong bộ nhớ ngoài. Có thể chia thành các dạng tổ chức sau:

- Nối tiếp: các bản ghi phân bố theo thứ tự vật lý, bản ghi logic tiếp theo nằm nối tiếp về vật lý. Tổ chức file theo kiểu nối tiếp thường áp dụng trong băng từ.
- Dãy chỉ số: các bản ghi nối tiếp về logic không nhất thiết liên tiếp về vật lý. Trong hệ thống sử dụng các chỉ mục riêng trỏ đến vị trí vật lý của bản ghi. Tổ chức này thường áp dụng trong đĩa từ.
- Truy cập trực tiếp: truy cập bản ghi trực tiếp theo địa chỉ vật lý. Tổ chức hệ thống đơn giản nhưng gánh nặng chuyển sang người lập trình, họ phải biết rõ cấu trúc vật lý của thiết bị. Vì thế hình thức này ít áp dụng.
- Thư mục (Directory): là kiến trúc cao hơn file, bao gồm tập hợp các file. Về thực chất thì thư mục cũng là file, chỉ có điều dữ liệu trong đó là thông tin về các file nằm trong thư mục.

Hệ thống file (File System)

Ta có thể đưa ra các đặc điểm sau của file:

- Tần số thay đổi: Static và Dynamic.
- Kích thước file.

File System là cấu thành quan trọng của hệ điều hành, hệ thống file thường cung cấp các công cụ:

- Các phương pháp truy cập: xác định, tổ chức truy cập dữ liệu.
- Các cơ chế điều khiển file: điều khiển truy cập, chia sẻ file cho nhiều người dùng, bảo vệ dữ liệu.
- Công cụ điều khiển bộ nhớ ngoài: quản lý việc phân bố, cấp phát bộ nhớ ngoài.
- Công cụ đảm bảo tính toàn vẹn dữ liệu.
- Chức năng quan trọng của hệ thống file là cấp phát bộ nhớ ngoài, điều khiển truy cập. Trong các hệ đa người dùng, đa nhiệm, cùng một lúc có thể có nhiều yêu cầu truy cập đồng thời. Hệ thống file phải đảm bảo phục vụ các yêu cầu nhanh nhất, vẫn đảm bảo an toàn dữ liệu.

Cấp phát và giải phóng không gian đĩa

Vấn đề cấp phát và giải phóng không gian đĩa, ở mức độ nào đó có nhiều điểm chung với phân bố bộ nhớ trong các hệ đa nhiệm. Nếu như ban đầu file được ghi trong một vùng nhớ liên tục thì theo thời gian, các file liên tục thay đổi và không gian đĩa trở nên bị chia nhỏ (Fragmentation).

Một trong những biện pháp giải quyết tình trạng đó là theo chu kỳ thực hiện việc dồn đĩa. Các file được tổ chức lại để chúng chiếm vùng đĩa liên tục. Việc này thường được thực hiện vào thời gian rỗi của hệ thống. Một số hệ thống còn có thể dọn dẹp đĩa ngay cả khi đang phục vụ người dùng.

Trong một số tình huống, khi mà số yêu cầu đọc/ghi dữ liệu rất nhiều thì việc dồn đĩa có thể không mang lại lợi ích. Ví dụ các yêu cầu có thể yêu cầu dữ liệu của các file trên vùng đĩa cách xa nhau ngay cả khi các file nằm trên vùng đĩa liên tục.

Khái niệm locality trong bộ nhớ ảo cũng có mặt trong hệ thống file, theo đó thường khi truy cập thông tin thường scan các khối dữ liệu. Do đó việc sắp xếp để các file chiếm vùng đĩa liên tục thực sự có hiệu quả.

Việc xác định thói quen của người dùng: các ứng dụng, các dữ liệu thường xuyên được dùng cũng có thể cần để ý trong thiết kế hệ thống file.

Trong các hệ thống dùng tổ chức bộ nhớ theo trang, khối trao đổi bộ nhớ với bộ nhớ ngoài là bội của trang, khi đó việc tổ chức bộ nhớ ngoài theo các khối có kích thước tương đương cũng có ý nghĩa. Với tính locality, việc truy xuất các trang bộ nhớ liền nhau dẫn đến việc cần phân bố dữ liệu trong bộ nhớ ngoài cũng cần liền nhau.

5.2.2. Phân bố liên tục, không liên tục

Có hai hình thức phân bố: liên tục và không liên tục.

Phân bố liên tục

Trong hình thức này, mỗi file được cấp phát một vùng liên tục. Nếu không có được vùng trống liên tục đủ lưu file thì file sẽ không thể được lưu.

Một trong các ưu điểm của phân bố liên tục là các bản ghi liên tiếp về logic cũng được lưu liên tục về mặt vật lý, điều đó cho phép cải thiện tốc độ truy cập.

Việc tổ chức lưu trữ cũng tương đối đơn giản.

Tuy nhiên phân bố liên tục có những hạn chế, khi các file bị xoá, thay đổi, tạo ra tình trạng phân mảnh. Hơn nữa, khi cần ghi thêm dữ liệu vào file không phải luôn có vùng trống ngay cuối file để lưu.

Để tránh tình trạng phân mảnh có thể thực hiện dồn file, tuy nhiên việc đó không phải luôn mang lại hiệu quả.

Phân bố không liên tục

Bởi vì thường xuyên các file bị thay đổi, do đó hình thức phân bố liên tục đã nhanh chóng bị thay thế bởi hình thức phân bố không liên tục, mặc dù tổ chức phức tạp hơn nhưng nó cho phép xây dựng hệ thống mềm dẻo hơn. Có một số hình thức phân bố không liên tục.

1) Sử dụng danh sách Sector

Trong sơ đồ này, đĩa được xem như tập hợp các Sector riêng rẽ. File có thể bao gồm các Sector nằm trên các vị trí rải rác. Các Sector thuộc cùng một file có các con trỏ đến nhau, tạo thành chuỗi/danh sách các Sector. Không gian trống được chỉ ra trong một danh sách các Sector trống.

Khi cần cấp phát thêm bộ nhớ, chỉ cần cấp Sector từ danh sách Sector trống. Còn khi file giảm kích thước thì Sector được đánh dấu trống và đưa vào danh sách. Việc dồn đĩa không phải đặt ra.

Tuy nhiên nó cũng có những nhược điểm. Bởi vì các Sector có thể nằm rải rác trên đĩa, việc truy cập dữ liệu liên tiếp trong file có thể kéo theo thời gian tìm kiếm Sector khá lâu. Việc đọc dữ liệu kéo theo phải duyệt qua chuỗi các Sector, xử lý con trỏ kết nối.

2) Phân bố theo khối

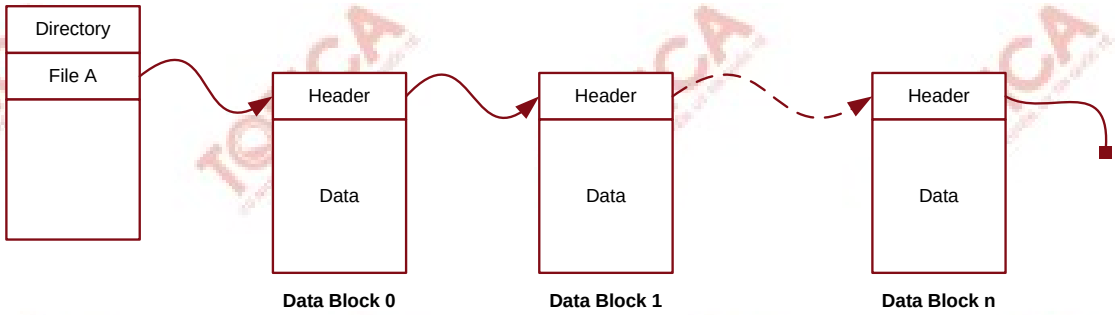
Hình thức phân bố theo khối là kết hợp giữa hình thức phân bố liên tục và không liên tục, trong đó bộ nhớ ngoài được chia thành các khối/block gồm nhiều Sector liên tục. Khi cấp phát thêm, hệ thống file cố gắng cấp phát khối trống gần nhất. Khi truy cập dữ liệu, hệ thống file xác định số blokc và sau đó số Sector trong khối. Có 3 hình thức phân bố theo khối: chuỗi block/block chaining, chuỗi block chỉ số/index block và bảng ánh xạ block.

5.2.3. Các sơ đồ tổ chức hệ thống file theo khối

Chuỗi các block dữ liệu

Trong sơ đồ này, bản ghi trong directory trỏ đến khối đầu tiên của file. Mỗi khối (có độ dài cố định) gồm hai phần: phần header chứa con trỏ đến khối tiếp theo (khối cuối cùng - con trỏ đặc biệt để báo hiệu) và phần dữ liệu. Đơn vị cấp phát nhỏ nhất là khối. Để đọc bản ghi, đầu tiên cần tìm đến khối sau đó đến Sector trong khối chứa bản ghi.

Quá trình tìm kiếm phải bắt đầu duyệt từ khối đầu tiên, theo các con trỏ đến khối cần tìm. Như thế nếu khối phân bố rải rác thì thời gian truy cập lâu hơn đáng kể. Việc cấp phát thêm khối hay xoá bớt khá dễ dàng thông qua định hướng lại con trỏ. Trong một số hệ thống, tổ chức chuỗi khối liên kết 2 chiều, mỗi khối có 2 con trỏ, 1 trỏ tới khối tiếp theo và 1 trỏ tới khối trước.



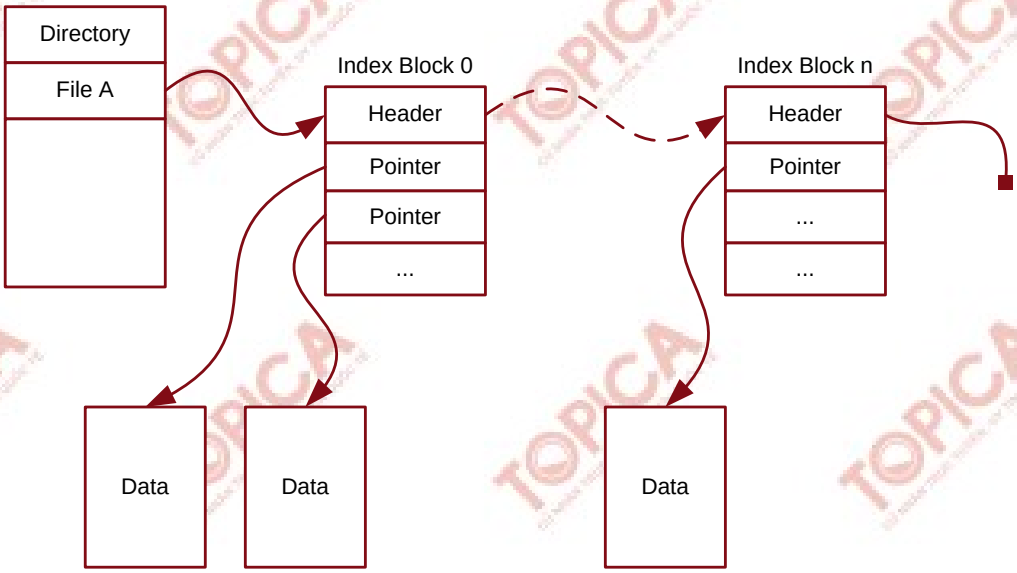
Hình 5.2.3-1: ***Chuỗi các block***

Sơ đồ chuỗi block chỉ mục

Trong sơ đồ này, các chỉ mục (Index) được chứa trong các khối chỉ số. Mỗi khối chỉ số chứa số lượng cố định các phần tử, mỗi phần tử chứa con trỏ đến khối dữ liệu. Nếu một index block không đủ, hệ thống dùng chuỗi các chỉ số block để chứa thông tin của file, mỗi index block có con trỏ đến index block tiếp theo trong chuỗi.

So với sơ đồ dùng chuỗi các block, sơ đồ này có một số ưu điểm. Khi tìm kiếm khối dữ liệu nào đó, chỉ cần tìm (duyệt) trong số lượng ít các index block thay vì duyệt qua số lượng lớn các khối dữ liệu. Để tăng tốc độ, có thể bố trí các chỉ số block nằm trên vùng liên tục trong bộ nhớ ngoài. Khi tìm thấy vị trí khối dữ liệu thì chỉ cần nạp khối dữ liệu đó vào bộ nhớ.

Sơ đồ này có nhược điểm chủ yếu là khi có thay đổi dữ liệu, ví dụ khi cần thêm các bản ghi dữ liệu mới có thể dẫn đến phải cấu trúc lại nhiều index block. Trong một số hệ thống, để khắc phục người ta đánh sẵn các vùng trống dự trữ cho các index block có thể có trong tương lai. Tuy nhiên, khi vùng trống đó đầy thì cũng dẫn đến phải cấu trúc lại chỉ số.



Hình 5.2.3-2: ***Sơ đồ chuỗi index block***

Sơ đồ bảng ánh xạ block

Trong sơ đồ này để xác định vị trí block, không cần phải có con trỏ mà dùng số của block, bởi vì từ số khối có thể dễ dàng tính toán được vị trí khối trên bộ nhớ ngoài. Sơ đồ này sử dụng bảng ánh xạ khối, mỗi dòng chứa thông tin ánh xạ cho 1 khối. Dòng trong bảng thư mục sẽ trỏ đến một dòng trong bảng ánh xạ – tương ứng với khối dữ

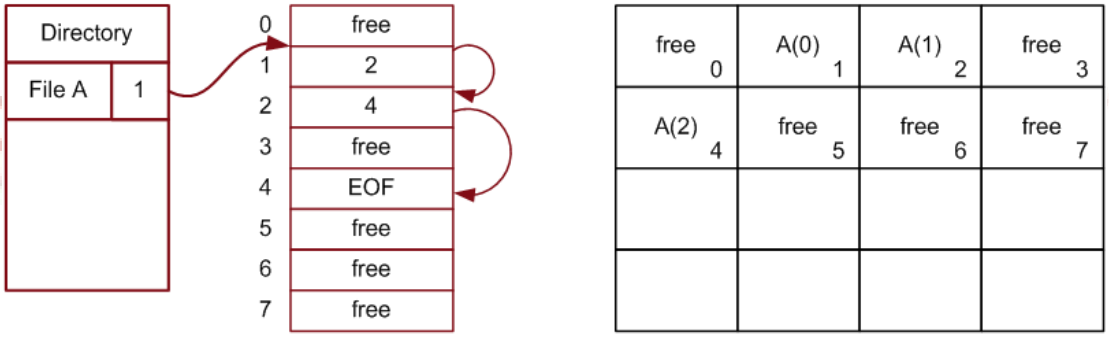
liệu đầu tiên của file. Mỗi dòng của bảng ánh xạ chứa số của khối tiếp theo của file. Như thế từ bảng ánh xạ có thể đọc được tất cả thông tin vị trí các khối dữ liệu của file.

Với dòng ánh xạ tương ứng với block cuối của file, thường chứa giá trị đặc biệt (ví dụ null) thể hiện rằng không còn khối nào tiếp theo – nó là khối cuối cùng.

Các dòng ánh xạ của khối trống chứa giá trị đặc biệt khác thể hiện rằng đó là khối trống, có thể cấp phát. Hệ thống có thể tra cứu bảng ánh xạ để tìm các khối trống, hoặc lưu thông tin khối trống trong một danh sách riêng.

Bảng ánh xạ có thể được thay đổi để lưu cả các thông tin khác, phục vụ cho mục đích tìm kiếm.

Một trong các ưu điểm lớn nhất của sơ đồ này là chỉ cần tra cứu thông tin trong bảng ánh xạ file là có thể xác định các khối trống và vị trí tương đối của chúng so với các khối khác, từ đó khi cần cấp phát có thể chọn các khối gần với khối thuộc cùng 1 file một cách dễ dàng, nâng cao tốc độ truy cập.



Hình 5.2.3-3: ***Bảng ánh xạ khối***

5.2.4. Điều khiển truy cập

Tại sao cần quản lý quyền truy cập đến file. Trong hệ thống nhiều người dùng, mỗi người đều lưu thông tin riêng của mình trên các file và các người dùng khác nói chung không được truy cập đến chúng, do đó hệ thống cần có các thông tin về quyền truy cập để điều khiển truy cập.

Có nhiều cơ chế khác nhau để quản lý quyền truy cập, một trong các sơ đồ đơn giản nhất là sử dụng ma trận điều khiển quyền truy cập (hình 5.2.4). Trong đó một chiều là các ID người dùng (hàng Ri), một chiều là ID file (cột Ci). Vị trí ô RiCj chứa giá trị quyền truy cập của người dùng Ui đối với file Fj. Nếu kích thước ô là 1 bit, ta có thể xác định 2 quyền, ví dụ giá trị 0 là không được đọc, 1 là có quyền đọc,... Tất nhiên để quản lý nhiều quyền hơn thì ta cần lưu được tổ hợp nhiều quyền hơn.

file / user	f1	f2	...
u1	R=0	R=1	
u2	R=1	R=1	
...			

Hình 5.2.4

Mô tả file – File Descriptor

File Descriptor là bản ghi lưu đầy đủ các thông tin về file mà hệ điều hành cần quan tâm, quản lý. Các thông tin lưu trong File Descriptor sẽ phụ thuộc vào từng hệ thống, nhưng nói chung chúng đều có các thông tin sau:

- Tên file.
- Kích thước file.
- Kiểu file.
- Ngày tháng tạo, cập nhật, ...
- Thông tin thuộc tính.

File Descriptor nằm trên bộ nhớ ngoài và chúng được nạp vào bộ nhớ khi mở (Open) file. Nói chung thì File Descriptor được xử lý bởi File System, người dùng không có quyền truy cập trực tiếp.

TÓM LƯỢC CUỐI BÀI

Các bạn đã được học về Bộ nhớ ngoài.

Chúng ta cần ghi nhớ các vấn đề sau:

- Hiểu được về đĩa từ:
  - Cấu tạo của đĩa từ.
  - Tối ưu theo thời gian định vị Track.
  - Tối ưu theo thời gian trễ.
- Hiểu được về hệ thống file:
  - Phân bố liên tục và không liên tục.
  - Các sơ đồ tổ chức hệ thống file.
  - Điều khiển truy cập.
- Làm được các bài tập về hệ thống file.

CÂU HỎI TỰ LUẬN

Câu 1. Giả sử một hệ thống có 3 loại thiết bị là Máy in, Ổ Đĩa cứng và Ổ CD–ROM. Có 1 yêu cầu in tập tin DanhSach.doc, 1 yêu cầu đọc F1.txt từ đĩa cứng, 1 yêu cầu ghi ra F2.txt trên đĩa cứng. Hãy thể hiện bằng hình vẽ Bảng trạng thái thiết bị với 3 yêu cầu Cập/Xuất kể trên.

Câu 2. Trong hai loại bộ nhớ là Bộ nhớ chính và Đĩa từ, loại nào là bộ nhớ sơ cấp, loại nào là bộ nhớ thứ cấp? Phân loại như vậy để làm gì?

Câu 3. Trình bày thuật giải điều phối FCFS.

Câu 4. Giả sử một Partition trên đĩa cứng được cài hệ tập tin FAT. Hãy thể hiện cấu trúc của Partition đó bằng hình vẽ.

Câu 5. Trong một hệ tập tin FAT16, tập tin Vanban.doc có nội dung trải trên các liên cung (Cluster) 5, 3, 2. Hãy minh hoạ cấu trúc Đầu mục (Directory Entry) của tập tin này cùng nội dung bảng FAT.

BÀI TẬP TRẮC NGHIỆM

1. Máy tính PC sử dụng hệ thống cơ số nào để lưu trữ dữ liệu trên đĩa cứng?

a) Hệ thập lục phân.
b) Hệ thập phân.
c) Hệ bát phân.
d) Hệ nhị phân.

2. Hệ điều hành thường được lưu trữ trong:

a) ROM.
b) RAM.
c) Bộ nhớ ngoài.
d) Bộ xử lí trung tâm (CPU).

3. Khi khởi động máy tính, hệ điều hành được nạp vào:

a) Bộ nhớ RAM.
b) Bộ nhớ ROM.
c) Bộ nhớ ngoài.
d) Bộ xử lý trung tâm.

4. Hệ điều hành đảm nhiệm việc nào trong những việc dưới đây?

a) Soạn thảo văn bản.
b) Giao tiếp với ổ đĩa cứng, quản lí bộ nhớ trong.
c) Chơi trò chơi điện tử.
d) Giải các bài toán trên máy tính.

5. Tệp tin thường được lưu trữ tại:

a) RAM.
b) ROM.
c) Bộ xử lí trung tâm.
d) Bộ nhớ ngoài.

6. Trong các câu sau câu nào không phải nhiệm vụ của hệ quản lý tệp?

a) Tổ chức thông tin trên bộ nhớ ngoài.

b) Cung cấp các dịch vụ để đọc/ ghi thông tin trên bộ nhớ ngoài dễ dàng.

c) Đảm bảo cho các chương trình đang hoạt động trong hệ thống có thể đồng thời truy cập tới các tệp.

d) Quản lí các thiết bị vật lý kết nối đến máy tính.

7. Các tài nguyên điển hình thuộc phần cứng bao gồm:

a) Thiết bị xử lý trung tâm (CPU).

b) Bộ nhớ trong, hệ thống vào/ra (kênh, thiết bị điều khiển thiết bị vào/ra và thiết bị vào/ra).

c) Bộ nhớ ngoài.

d) Cả 3 đáp án trên đều đúng.