CHƯƠNG III

LƯU TRỮ VÀ CẤU TRÚC TẬP TIN
(Storage and File Structure)

MỤC ĐÍCH

Chương này trình bày các vấn đề liên quan đến vấn đề lưu trữ dữ liệu (trên lưu trữ ngoài, chủ yếu trên đĩa cứng). Việc lưu trữ dữ liệu phải được tổ chức sao cho có thể cất giữ một lượng lớn, có thể rất lớn dữ liệu nhưng quan trọng hơn cả là sự lưu trữ phải cho phép lấy lại dữ liệu cần thiết mau chóng. Các cấu trúc trợ giúp cho truy xuất nhanh dữ liệu được trình bày là: chỉ mục (indice), B^+ cây (B^+-tree), băm (hashing) ... Các thiết bị lưu trữ (đĩa) có thể bị hỏng hóc không lường trước, các kỹ thuật RAID cho ra một giải pháp hiệu quả cho vấn đề này.

YÊU CẦU

Hiểu rõ các đặc điểm của các thiết bị lưu trữ, cách tổ chức lưu trữ, truy xuất đĩa.

Hiểu rõ nguyên lý và kỹ thuật của tổ chức hệ thống đĩa RAID

Hiểu rõ các kỹ thuật tổ chức các mẩu tin trong file

Hiểu rõ các kỹ thuật tổ chức file

Hiểu và vận dụng các kỹ thuật hỗ trợ tìm lại nhanh thông tin: chỉ mục (được sắp, B^+-cây, băm)

KHÁI QUÁT VỀ PHƯƠNG TIỆN LƯU TRỮ VẬT LÝ

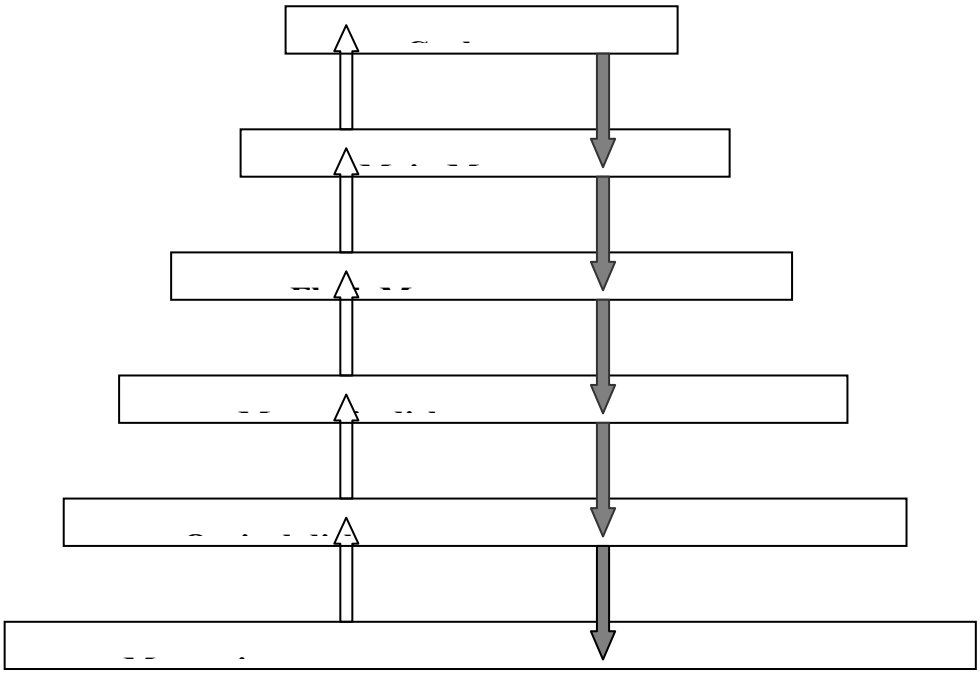
Có một số kiểu lưu trữ dữ liệu trong các hệ thống máy tính. Các phương tiện lưu trữ được phân lớp theo *tốc độ truy xuất, theo giá cả và theo độ tin cậy của phương tiện*. Các phương tiện hiện có là:

- **Cache**: là dạng lưu trữ nhanh nhất và cũng đắt nhất trong các phương tiện lưu trữ. Bộ nhớ cache nhỏ; sự sử dụng nó được quản trị bởi hệ điều hành
- **Bộ nhớ chính (main memory)**: Phương tiện lưu trữ dùng để lưu trữ dữ liệu sẵn sàng được thực hiện. Các chỉ thị máy mục đích chung (general-purpose) hoạt động trên bộ nhớ chính. Mặc dầu bộ nhớ chính có thể chứa nhiều megabytes dữ liệu, nó vẫn là quá nhỏ (và quá đắt giá) để lưu trữ toàn bộ một cơ sở dữ liệu. Nội dung trong bộ nhớ chính thường bị mất khi mất cấp nguồn
- **Bộ nhớ Flash:** Được biết như bộ nhớ chỉ đọc có thể lập trình, có thể xoá (EEPROM: Electrically Erasable Programmable Read-Only Memory), Bộ nhớ Flash khác bộ nhớ chính ở chỗ dữ liệu còn tồn tại trong bộ nhớ flash khi mất cấp nguồn. Đọc dữ liệu từ bộ nhớ flash mất ít hơn 100 ns , nhanh như đọc dữ liệu từ bộ nhớ chính. Tuy nhiên, viết dữ liệu vào bộ nhớ flash phức tạp hơn nhiều. Dữ liệu được viết (một lần mất khoảng 4 đến 10 μs) nhưng không thể viết đè trực tiếp. Để viết đè bộ nhớ đã được viết, ta phải xoá trắng toàn bộ bộ nhớ sau đó mới có thể viết lên nó.
- **Lưu trữ đĩa từ (magnetic-disk): (**ở đây, được hiểu là đĩa cứng) Phương tiện căn bản để lưu trữ dữ liệu trực tuyến, lâu dài. Thường toàn bộ cơ sở dữ liệu được lưu trữ trên đĩa từ. Dữ liệu phải được chuyển từ đĩa vào bộ nhớ chính trước khi được truy nhập. Khi dữ liệu trong bộ nhớ chính này bị sửa đổi, nó phải được viết lên đĩa. Lưu trữ đĩa được xem là truy xuất trực tiếp vì có thể đọc dữ liệu trên đĩa theo một thứ tự bất kỳ. Lưu trữ đĩa vẫn tồn tại khi mất cấp nguồn. Lưu trữ đĩa có thể bị hỏng hóc, tuy không thường xuyên.
- **Lưu trữ quang (Optical storage):** Dạng quen thuộc nhất của đĩa quang học là loại đĩa **CD-ROM** : Compact-Disk Read-Only Memory. Dữ liệu được lưu trữ trên các đĩa quang học được đọc bởi laser. Các đĩa quang học CD-ROM chỉ có thể đọc. Các phiên bản khác của chúng là loại đĩa quang học: viết một lần, đọc nhiều lần (write-once, read-many: **WORM**) cho phép viết dữ liệu lên đĩa một lần, không cho phép xoá và viết lại, và các đĩa có thể viết lại (rewritable) v..v
- **Lưu trữ băng từ (tape storage):** Lưu trữ băng từ thường dùng để backup dữ liệu. Băng từ rẻ hơn đĩa, truy xuất dữ liệu chậm hơn (vì phải truy xuất tuần tự). Băng từ thường có dung lượng rất lớn.

Các phương tiện lưu trữ có thể được tổ chức phân cấp theo tốc độ truy xuất và giá cả. Mức cao nhất là nhanh nhất nhưng cũng là đắt nhất, giảm dần xuống các mức thấp hơn.

Các phương tiện lưu trữ nhanh (*cache, bộ nhớ chính*) được xem như là lưu trữ sơ cấp (primary storage), các thiết bị lưu trữ ở mức thấp hơn như đĩa từ được xem như lưu trữ thứ cấp hay lưu trữ trực tuyến (on-line storage), còn các thiết bị lưu trữ ở mức thấp nhất và gần thấp nhất như đĩa quang học, băng từ kể cả các đĩa mềm được xếp vào lưu trữ tam cấp hay lưu trữ không trực tuyến (off-line).

Bên cạnh vấn đề tốc độ và giá cả, ta còn phải xét đến tính lâu bền của các phương tiện lưu trữ.



Phân cấp thiết bị lưu trữ

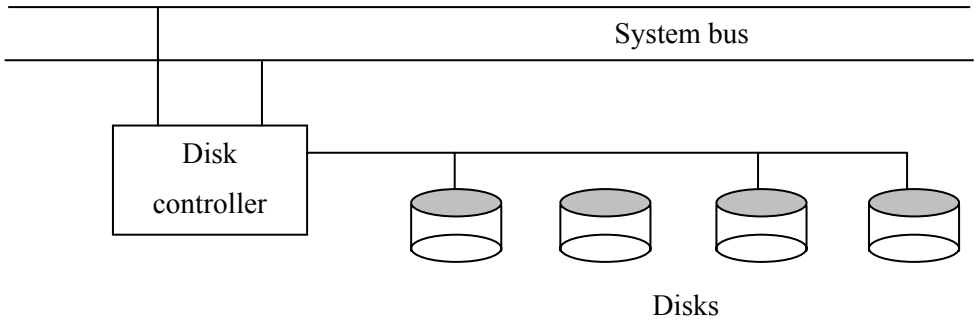
ĐĨA TỪ

ĐẶC TRƯNG VẬT LÝ CỦA ĐĨA

Mỗi tấm đĩa có dạng hình tròn, hai mặt của nó được phủ bởi vật liệu từ tính, thông tin được ghi trên bề mặt đĩa. Đĩa gồm nhiều tấm đĩa. Ta sẽ sử dụng thuật ngữ đĩa để chỉ các đĩa cứng.

Khi đĩa được sử dụng, một động cơ ổ đĩa làm quay nó ở một tốc độ không đổi. Một đầu đọc-viết được định vị trên bề mặt của tấm đĩa. Bề mặt tấm đĩa được chia logic thành các rãnh, mỗi rãnh lại được chia thành các sector, một sector là một đơn vị thông tin nhỏ có thể được đọc, viết lên đĩa. Tuỳ thuộc vào kiểu đĩa, sector thay đổi từ 32 bytes đến 4095 bytes, thông thường là 512 bytes. Có từ 4 đến 32 sectors trên một rãnh, từ 20 đến 1500 rãnh trên một bề mặt. Mỗi bề mặt của một tấm đĩa có một đầu đọc viết, nó có thể chạy dọc theo bán kính đĩa để truy cập đến các rãnh khác nhau. Một đĩa gồm nhiều tấm đĩa, các đầu đọc-viết của tất cả các rãnh được gắn vào một bộ được gọi là cánh tay đĩa, di chuyển cùng nhau. Các tấm đĩa được gắn vào một trục quay. Vì các đầu đọc-viết trên các tấm đĩa di chuyển cùng nhau, nên khi đầu đọc-viết trên một tấm đĩa đang ở rãnh thứ i thì các đầu đọc-viết của các tấm đĩa khác cũng ở rãnh thứ i , do vậy các rãnh thứ i của tất cả các tấm đĩa được gọi là trụ (cylinder) thứ i . Một bộ điều khiển đĩa -- giao diện giữa hệ thống máy tính và phần cứng hiện thời của ổ đĩa. Nó chấp nhận các lệnh mức cao để đọc và viết một sector, và khởi động các hành động như di chuyển cánh tay đĩa đến các rãnh đúng và đọc viết dữ liệu. bộ điều khiển đĩa cũng tham gia vào checksum mỗi sector được viết. Checksum được tính từ dữ liệu được viết lên sector. Khi sector được đọc lại, checksum được tính lại từ dữ liệu được lấy ra và so sánh với checksum đã lưu trữ. Nếu dữ liệu bị sai lạc, checksum được tính sẽ không khớp với checksum đã lưu trữ. Nếu lỗi như vậy xảy ra, bộ điều khiển sẽ lặp lại việc đọc vài lần, nếu lỗi vẫn xảy ra, bộ điều khiển sẽ thông báo việc đọc thất bại. Bộ điều khiển đĩa còn có

chức năng tái ánh xạ các sector xấu: ánh xạ các sector xấu đến một vị trí vật lý khác. Hình dưới bày tỏ các đĩa được nối với một hệ thống máy tính:



Các đĩa được nối với một hệ thống máy tính hoặc một bộ điều khiển đĩa qua một sự hợp nhất tốc độ cao. Hợp nhất hệ thống máy tính nhỏ (Small Computer-System Interconnect: SCSI) thường được sử dụng để nối kết các đĩa với các máy tính cá nhân và workstation. Mainframe và các hệ thống server thường có các bus nhanh hơn và đắt hơn để nối với các đĩa.

Các đầu đọc-viết được giữ sát với bề mặt đĩa như có thể để tăng độ dày đặc (density).

Đĩa đầu cố định (Fixed-head) có một đầu riêng biệt cho mỗi rãnh, sự sắp xếp này cho phép máy tính chuyển từ rãnh này sang rãnh khác mau chóng, không phải di chuyển đầu đọc-viết. Tuy nhiên, cần một số rất lớn đầu đọc-viết, điều này làm nâng giá của thiết bị.

ĐO LƯỜNG HIỆU NĂNG CỦA ĐĨA

Các tiêu chuẩn đo lường chất lượng chính của đĩa là ***dung lượng, thời gian truy xuất, tốc độ truyền dữ liệu và độ tin cậy***.

- ***Thời gian truy xuất (access time)***: là khoảng thời gian từ khi yêu cầu đọc/viết được phát đi đến khi bắt đầu truyền dữ liệu. Để truy xuất dữ liệu trên một sector đã cho của một đĩa, đầu tiên cánh tay đĩa phải di chuyển đến rãnh đúng, sau đó phải chờ sector xuất hiện dưới nó, thời gian để định vị cánh tay được gọi là thời gian tìm kiếm (seek time), nó tỷ lệ với khoảng cách mà cánh tay phải di chuyển, thời gian tìm kiếm nằm trong khoảng 2..30 ms tuỳ thuộc vào rãnh xa hay gần vị trí cánh tay hiện tại.
- ***Thời gian tìm kiếm trung bình (average seek time)***: Thời gian tìm kiếm trung bình là trung bình của thời gian tìm kiếm, được đo luờng trên một dãy các yêu cầu ngẫu nhiên (phân phối đều), và bằng khoảng 1/3 thời gian tìm kiếm trong trường hợp xấu nhất.
- ***Thời gian tiềm ẩn luân chuyển (rotational latency time)***: Thời gian chờ sector được truy xuất xuất hiện dưới đầu đọc/viết. Tốc độ quay của đĩa nằm trong khoảng 60..120 vòng quay trên giây, trung bình cần nửa vòng quay để sector cần thiết nằm dưới đầu đọc/viết. Như vậy, thời gian tiềm ẩn trung bình (average latency time) bằng nửa thời gian quay một vòng đĩa.

Thời gian truy xuất bằng tổng của thời gian tìm kiếm và thời gian tiềm ẩn và nằm trong khoảng 10..40 ms.

- ***Tốc độ truyền dữ liệu:*** là tốc độ dữ liệu có thể được lấy ra từ đĩa hoặc được lưu trữ vào đĩa. Hiện nay tốc này vào khoảng1..5 Mbps

- ***Thời gian trung bình không sự cố (mean time to failure):*** lượng thời gian trung bình hệ thống chạy liên tục không có bất kỳ sự cố nào. Các đĩa hiện nay có thời gian không sự cố trung bình khoảng 30000 .. 800000 giờ nghĩa là khoảng từ 3,4 đến 91 năm.

TỐI ƯU HÓA TRUY XUẤT KHỐI ĐĨA (disk-block)

Yêu cầu I/O đĩa được sinh ra cả bởi hệ thống file lẫn bộ quản trị bộ nhớ ảo trong hầu hết các hệ điều hành. Mỗi yêu cầu xác định địa chỉ trên đĩa được tham khảo, địa chỉ này ở dạng số khối. Một khối là một dãy các sector kề nhau trên một rãnh. Kích cỡ khối trong khoảng 512 bytes đến một vài Kbytes. Dữ liệu được truyền giữa đĩa và bộ nhớ chính theo đơn vị khối. Mức thấp hơn của bộ quản trị hệ thống file sẽ chuyển đổi địa chỉ khối sang số của trụ, của mặt và của sector ở mức phần cứng.

Truy xuất dữ liệu trên đĩa chậm hơn nhiều so với truy xuất dữ liệu trong bộ nhớ chính, do vậy cần thiết một chiến lược nhằm nâng cao tốc độ truy xuất khối đĩa. Dưới đây ta sẽ thảo luận một vài kỹ thuật nhằm vào mục đích đó.

- ***Scheduling:*** Nếu một vài khối của một trụ cần được truyền từ đĩa vào bộ nhớ chính, ta có thể tiết kiệm thời gian truy xuất bởi yêu cầu các khối theo thứ tự mà nó chạy qua dưới đầu đọc/viết. Nếu các khối mong muốn ở trên các trụ khác nhau, ta yêu cầu các khối theo thứ tự sao cho làm tối thiểu sự di chuyển cánh tay đĩa. Các thuật toán scheduling cánh tay đĩa (Disk-arm-scheduling) nhằm lập thứ tự truy xuất các rãnh theo cách làm tăng số truy xuất có thể được xử lý. Một thuật toán thường dùng là thuật toán thang máy (elevator algorithm): Giả sử ban đầu cánh tay di chuyển từ rãnh trong nhất hướng ra phía ngoài đĩa, đối với mỗi rãnh có yêu cầu truy xuất, nó dừng lại, phục vụ yêu cầu đối với rãnh này, sau đó tiếp tục di chuyển ra phía ngoài đến tận khi không có yêu cầu nào chờ các rãnh xa hơn phía ngoài. Tại điểm này, cánh tay đổi hướng, di chuyển vào phía trong, lại dừng lại trên các rãnh được yêu cầu, và cứ như vậy đến tận khi không còn rãnh nào ở trong hơn được yêu cầu, rồi lại đổi hướng .. v .. v .. Bộ điều khiển đĩa thường làm nhiệm vụ sắp xếp lại các yêu cầu đọc để cải tiến hiệu năng.
- ***Tổ chức file:*** Để suy giảm thời gian truy xuất khối, ta có thể tổ chức các khối trên đĩa theo cách tương ứng gần nhất với cách mà dữ liệu được truy xuất. Ví dụ, Nếu ta muốn một file được truy xuất tuần tự, khi đó ta bố trí các khối của file một cách tuần tự trên các trụ kề nhau. Tuy nhiên việc phân bố các khối lưu trữ kề nhau này sẽ bị phá vỡ trong quá trình phát triển của file $\Rightarrow$ file không thể được phân bố trên các khối kề nhau được nữa, hiện tượng này dược gọi là sự phân mảnh (fragmentation). Nhiều hệ điều hành cung cấp tiện ích giúp suy giảm sự phân mảnh này (Defragmentation) nhằm làm tăng hiệu năng truy xuất file.
- ***Các buffers viết không hay thay đổi:*** Vì nội dung của bộ nhớ chính bị mất khi mất nguồn, các thông tin về cơ sở dữ liệu cập nhật phải được ghi lên đĩa nhằm đề phòng sự cố. Hiệu năng của các ứng dụng cập nhật cường độ cao phụ thuộc mạnh vào tốc độ viết đĩa. Ta có thể sử dụng bộ nhớ truy xuất ngẫu nhiên không hay thay đổi (nonvolatile RAM) để nâng tốc độ viết đĩa. Nội dung của nonvolatile RAM không bị mất khi mất nguồn. Một phương pháp chung để thực hiện nonvolatile RAM là sử dụng RAM pin dự phòng (battery-back-up RAM). Khi cơ sở dữ liệu yêu cầu viết một khối lên đĩa, bộ điều khiển dĩa viết khối này lên buffer nonvolatile RAM, và thông báo ngay cho hệ điều hành là việc viết đã thành công. Bộ điều khiển sẽ viết dữ liệu đến đích của nó trên đĩa, mỗi khi đĩa rảnh hoặc buffer nonvolatile RAM đầy. Khi hệ cơ sở dữ liệu yêu cầu một viết khối, nó chỉ chịu một khoảng lặng chờ đợi khi buffer nonvolatile RAM đầy.

- ***Đĩa log (log disk):*** Một cách tiếp cận khác để làm suy giảm tiềm năng viết là sử dụng log-disk: Một đĩa được tận hiến cho việc viết một log tuần tự. Tất cả các truy xuất đến log-disk là tuần tự, nhằm loại bỏ thời gian tìm kiếm, và một vài khối kề có thể được viết một lần, tạo cho viết vào log-disk nhanh hơn viết ngẫu nhiên vài lần. Cũng như trong trường hợp sử dụng nonvolatile RAM, dữ liệu phải được viết vào vị trí hiện thời của chúng trên đĩa, nhưng việc viết này có thể được tiến hành mà hệ cơ sở dữ liệu không cần thiết phải chờ nó hoàn tất. Log-disk có thể được sử dụng để khôi phục dữ liệu. Hệ thống file dựa trên log là một phiên bản của cách tiếp cận log-disk: Dữ liệu không được viết lại lên đích gốc của nó trên đĩa; thay vào đó, hệ thống file lưu vết nơi các khối được viết mới đây nhất trên log-disk, và hoàn lại chúng từ vị trí này. Log-disk được "cô đặc" lại (compacting) theo một định kỳ. Cách tiếp cận này cải tiến hiệu năng viết, song sinh ra sự phân mảnh đối với các file được cập nhật thường xuyên.

RAID

Trong một hệ thống có nhiều đĩa, ta có thể cải tiến tốc độ đọc viết dữ liệu nếu cho chúng hoạt động song song. Mặt khác, hệ thống nhiều đĩa còn giúp tăng độ tin cậy lưu trữ bằng cách lưu trữ dư thừa thông tin trên các đĩa khác nhau, nếu một đĩa có sự cố dữ liệu cũng không bị mất. Một sự đa dạng các kỹ thuật tổ chức đĩa, được gọi là **RAID (Redundant Arrays of Inexpensive Disks)**, được đề nghị nhằm vào vấn đề tăng cường hiệu năng và độ tin cậy.

CẢI TIẾN ĐỘ TIN CẬY THÔNG QUA SỰ DƯ THỪA

Giải pháp cho vấn đề độ tin cậy là đưa vào sự dư thừa: lưu trữ thông tin phụ, bình thường không cần thiết, nhưng nó có thể được sử dụng để tái tạo thông tin bị mất khi gặp sự cố hỏng hóc đĩa, như vậy thời gian trung bình không sự cố tăng lên (xét tổng thể trên hệ thống đĩa).

Đơn giản nhất, là làm bản sao cho mỗi đĩa. Kỹ thuật này được gọi là mirroring hay shadowing. Một đĩa logic khi đó bao gồm hai đĩa vật lý, và mỗi việc viết được thực hiện trên cả hai đĩa. Nếu một đĩa bị hư, dữ liệu có thể được đọc từ đĩa kia. Thời gian trung bình không sự cố của đĩa mirror phụ thuộc vào thời gian trung bình không sự cố của mỗi đĩa và phụ thuộc vào thời gian trung bình được sửa chữa (mean time to repair): thời gian trung bình để một đĩa bị hư được thay thế và phục hồi dữ liệu trên nó.

CẢI TIẾN HIỆU NĂNG THÔNG QUA SONG SONG

Với đĩa mirror, tốc độ đọc có thể tăng lên gấp đôi vì yêu cầu đọc có thể được gửi đến cả hai đĩa. Với nhiều đĩa, ta có thể cải tiến tốc độ truyền bởi phân nhỏ (striping data) dữ liệu qua nhiều đĩa. Dạng đơn giản nhất là tách các bít của một byte qua nhiều đĩa, sự phân nhỏ này được gọi là sự phân nhỏ mức bit (bit-level striping). Ví dụ, ta có một dàn 8 đĩa, ta viết bít thứ i của một byte lên đĩa thứ i . dàn 8 đĩa này có thể được xử lý như một đĩa với các sector 8 lần lớn hơn kích cỡ thông thường, quan trọng hơn là tốc dộ truy xuất tăng lên tám lần. Trong một tổ chức như vậy, mỗi đĩa tham gia vào mỗi truy xuất (đọc/viết), như vậy, số các truy xuất có thể được xử lý trong một giây là tương tự như trên một đĩa, nhưng mỗi truy xuất có thể đọc/viết nhiều dữ liệu hơn tám lần.

Phân nhỏ mức bit có thể được tổng quát cho số đĩa là bội hoặc ước của 8, Ví dụ, ta có một dàn 4 đĩa, ta sẽ phân phối bít thứ i và bít thứ 4+i vào đĩa thứ i. Hơn nữa, sự phân nhỏ không nhất thiết phải ở mức bit của một byte. Ví dụ, trong sự phân nhỏ mức khối, các khối của một file được phân nhỏ qua nhiều đĩa, với n đĩa, khối thứ i có thể được phân phối qua đĩa (i mod n) + 1. Ta cũng

có thể phân nhỏ ở mức byte, sector hoặc các sector của một khối. Hai đích song song trong một hệ thống đĩa là:

1. Nạp nhiều truy xuất nhỏ cân bằng (truy xuất trang) sao cho lượng dữ liệu được nạp trong một đơn vị thời gian của truy xuất như vậy tăng lên.
2. Song song hoá các truy xuất lớn sao cho thời gian trả lời các truy xuất lớn giảm.

CÁC MỨC RAID

Mirroring cung cấp độ tin cậy cao, nhưng đắt giá. Phân nhỏ cung cấp tốc độ truyền dữ liệu cao, nhưng không cải tiến được độ tin cậy. Nhiều sơ đồ cung cấp sự dư thừa với giá thấp bằng cách phối hợp ý tưởng của phân nhỏ với "parity" bit. Các sơ đồ này có sự thoả hiệp *giá-hiệu năng* khác nhau và được phân lớp thành các mức được gọi là các mức RAID.

• **Mức RAID 0 :** Liên quan đến các dàn đĩa với sự phân nhỏ mức khối, nhưng không có một sự dư thừa nào.

• **Mức RAID 1 :** Liên quan đến mirror đĩa

• **Mức RAID 2 :** Cũng được biết dưới cái tên mã sửa lỗi kiểu bộ nhớ (memory-style error-correcting-code : ECC). Hệ thống bộ nhớ thực hiện phát hiện lỗi bằng bit parity. Mỗi byte trong hệ thống bộ nhớ có thể có một bit parity kết hợp với nó. Sơ đồ sửa lỗi lưu hai hoặc nhiều hơn các bit phụ, và có thể dựng lại dữ liệu nếu một bit bị lỗi. ý tưởng của mã sửa lỗi có thể được sử dụng trực tiếp trong dàn đĩa thông qua phân nhỏ byte qua các đĩa. Ví dụ, bít đầu tiên của mỗi byte có thể được lưu trên đĩa 1, bit thứ hai trên đĩa 2, và cứ như vậy, bit thứ 8 trên đĩa 8, các bit sửa lỗi được lưu trên các đĩa thêm vào. Nếu một trong các đĩa bị hư, các bít còn lại của byte và các bit sửa lỗi kết hợp được đọc từ các đĩa khác có thể giúp tái tạo bít bị mất trên đĩa hư, như vậy ta có thể dựng lại dữ liệu. Với một dàn 4 đĩa dữ liệu, RAID mức 2 chỉ cần thêm 3 đĩa để lưu các bit sửa lỗi (các đĩa thêm vào này được gọi là các đĩa overhead), so sánh với RAID mức 1, cần 4 đĩa overhead.

• **Mức RAID 3 :** Còn được gọi là tổ chức parity chen bit (bit-interleaved parity). Bộ điều khiển đĩa có thể phát hiện một sector được đọc đúng hay sai, như vậy có thể sử dụng chỉ một bit parity để sửa lỗi: Nếu một trong các sector bị hư, ta biết chính xác đó là sector nào, Với mỗi bit trong sector này ta có thể hình dung nó là bít 1 hay bit 0 bằng cách tính parity của các bit tương ứng từ các sector trên các đĩa khác. Nếu parity của các bit còn lại bằng với parity được lưu, bit mất sẽ là 0, ngoài ra bit mất là 1. RAID mức 3 tốt như mức 2 nhưng it tốn kém hơn (chỉ cần một đĩa overhead).

• **Mức RAID 4 :** Còn được gọi là tổ chức parity chen khối (Block-interleaved parity), lưu trữ các khối đúng như trong các đĩa chính quy, không phân nhỏ chúng qua các đĩa nhưng lấy một khối parity trên một đĩa riêng biệt đối với các khối tương ứng từ N đĩa khác. Nếu một trong các đĩa bị hư, khối parity có thể được dùng với các khối tương ứng từ các đĩa khác để khôi phục khối của đĩa bị hư.

Một đọc khối chỉ truy xuất một đĩa, cho phép các yêu cầu khác được xử lý bởi các đĩa khác. Như vậy, tốc độ truyền dữ liệu đối với mỗi truy xuất chậm, nhưng nhiều truy xuất đọc có thể được xử lý song song, dẫn đến một tốc độ I/O tổng thể cao hơn. Tốc độ truyền đối với các đọc dữ liệu lớn (nhiều khối) cao do tất cả các đĩa có thể được đọc song song; các viết dữ liệu lớn (nhiều khối) cũng có tốc độ truyền cao vì dữ liệu và parity có thể được viết song song. Tuy nhiên, viết một khối đơn phải truy xuất đĩa trên đó khối được lưu trữ, và đĩa parity (do khối parity cũng phải được cập nhật). Như vậy, viết một khối đơn yêu cầu 4 truy xuất: hai để đọc hai khối cũ, và hai để viết lại hai khối.

- **Mức RAID 5 :** Còn gọi là parity phân bố chen khối (Block-interleaved Distributed Parity), cải tiến của mức 4 bởi phân hoạch dữ liệu và parity giữa toàn bộ N+1 đĩa, thay vì lưu trữ dữ liệu trên N đĩa và parity trên một đĩa riêng biệt như trong RAID 4. Trong RAID 5, tất cả các đĩa có thể tham gia làm thoả mãn các yêu cầu đọc, như vậy sẽ làm tăng tổng số yêu cầu có thể được đặt ra trong một đơn vị thời gian. Đối với mỗi khối, một đĩa lưu trữ parity, các đĩa khác lưu trữ dữ liệu. Ví dụ, với một dàn năm đĩa, parity đối với khối thứ n được lưu trên đĩa (n mod 5)+1. Các khối thứ n của 4 đĩa khác lưu trữ dữ liệu hiện hành của khối đó.

- **Mức RAID 6 :** Còn được gọi là sơ đồ dư thừa P+Q (P+Q redundancy scheme), nó rất giống RAID 5 nhưng lưu trữ thông tin dư thừa phụ để canh chừng nhiều đĩa bị hư. Thay vì sử dụng parity, người ta sử dụng các mã sửa lỗi.

CHỌN MỨC RAID ĐÚNG

Nếu đĩa bị hư, Thời gian tái tạo dữ liệu của nó là đáng kể và thay đổi theo mức RAID được dùng. Sự tái tạo dễ dàng nhất đối với mức RAID 1. Đối với các mức khác, ta phải truy xuất tất cả các đĩa khác trong dàn đĩa để tái tạo dữ liệu trên đĩa bị hư. Hiệu năng tái tạo của một một hệ thống RAID có thể là một nhân tố quan trọng nếu việc cung cấp dữ liệu liên tục được yêu cầu (thường xảy ra trong các hệ CSDL hiệu năng cao hoặc trao đổi). Hơn nữa, hiệu năng tái tạo ảnh hưởng đến thời gian trung bình không sự cố.

Vì RAID mức 2 và 4 được gộp lại bởi RAID mức 3 và 5, Việc lựa chọn mức RAID thu hẹp lại trên các mức RAID còn lại. Mức RAID 0 được dùng trong các ứng dụng hiệu năng cao ở đó việc mất dữ liệu không có gì là trầm trọng cả. RAID mức 1 là thông dụng cho các ứng dụng lưu trữ các log-file trong hệ CSDL. Do mức 1 có overhead cao, mức 3 và 5 thường được ưa thích hơn đối với việc lưu trữ khối lượng dữ liệu lớn. Sự khác nhau giữa mức 3 và mức 5 là tốc độ truyền dữ liệu đối lại với tốc độ I/O tổng thể. Mức 3 được ưa thích hơn nếu truyền dữ liệu cao được yêu cầu, mức 5 được ưa thích hơn nếu việc đọc ngẫu nhiên là quan trọng. Mức 6, tuy hiện nay ít được áp dụng, nhưng nó có độ tin cậy cao hơn mức 5.

MỞ RỘNG

Các quan niệm của RAID được khái quát hoá cho các thiết bị lưu trữ khác, bao hàm các dàn băng, thậm chí đối với quảng bá dữ liệu trên các hệ thống không dây. Khi áp dụng RAID cho dàn băng, cấu trúc RAID cho khả năng khôi phục dữ liệu cả khi một trong các băng bị hư hại. Khi áp dụng đối với quảng bá dữ liệu, một khối dữ liệu được phân thành các đơn vị nhỏ và được quảng bá cùng với một đơn vị parity; nếu một trong các đơn vị này không nhận được, nó có thể được dựng lại từ các đơn vị còn lại.

LƯU TRỮ TAM CẤP (tertiary storage)

ĐĨA QUANG HỌC

CR-ROM có ưu điểm là có khả năng lưu trữ lớn, dễ di chuyển (có thể đưa vào và lấy ra khỏi ổ đĩa như đĩa mềm), hơn nữa giá lại rẻ. Tuy nhiên, so với ổ đĩa cứng, thời gian tìm kiếm của ổ CD-ROM chậm hơn nhiều (khoảng 250ms), tốc độ quay chậm hơn (khoảng 400rpm), từ đó dẫn đến độ trễ cao hơn; tốc độ truyền dữ liệu cũng chậm hơn (khoảng 150Kbytes/s). Gần đây, một định dạng mới của đĩa quang học - Digital video disk (DVD) - được chuẩn hoá, các đĩa này có dung lượng trong khoảng 4,7GBytes đến 17 GBytes. Các đĩa WORM, REWRITABLE cũng trở thành phổ biến. Các WORM jukeboxes là các thiết bị có thể lưu trữ một số lớn các đĩa WORM và có thể nạp tự động các đĩa theo yêu cầu đến một hoặc một vài ổ WORM.

BĂNG TỪ

Băng từ có thể lưu một lượng lớn dữ liệu, tuy nhiên, chậm hơn so với đĩa từ và đĩa quang học. Truy xuất băng buộc phải là truy xuất tuần tự, như vậy nó không thích hợp cho hầu hết các đòi hỏi của lưu trữ thứ cấp. Băng từ được sử dụng chính cho việc backup, cho lưu trữ các thông tin không được sử dụng thường xuyên và như một phương tiện ngoại vi (off-line medium) để truyền thông tin từ một hệ thống đến một hệ thống khác. Thời gian để định vị đoạn băng lưu dữ liệu cần thiết có thể kéo dài đến hàng phút. Jukeboxes băng chứa một lượng lớn băng, với một vài ổ băng và có thể lưu trữ được nhiều TeraBytes (10^{12} Bytes)

TRUY XUẤT LƯU TRỮ

Một cơ sở dữ liệu được ánh xạ vào một số các file khác nhau được duy trì bởi hệ điều hành nền. Các file này lưu trú thường trực trên các đĩa với backup trên băng. Mỗi file được phân hoạch thành các đơn vị lưu trữ độ dài cố định được gọi là khối - đơn vị cho cả cấp phát lưu trữ và truyền dữ liệu.

Một khối có thể chứa một vài hạng mục dữ liệu (data item). Ta giả thiết không một hạng mục dữ liệu nào trải ra trên hai khối. Mục tiêu nổi trội của hệ CSDL là tối thiểu hoá số khối truyền giữa đĩa và bộ nhớ. Một cách để giảm số truy xuất đĩa là giữ nhiều khối như có thể trong bộ nhớ chính. Mục đích là để khi một khối được truy xuất, nó đã nằm sẵn trong bộ nhớ chính và như vậy không cần một truy xuất đĩa nào cả.

Do không thể lưu tất cả các khối trong bộ nhớ chính, ta cần quản trị cấp phát không gian sẵn có trong bộ nhớ chính để lưu trữ các khối. Bộ đệm (Buffer) là một phần của bộ nhớ chính sãn có để lưu trữ bản sao khối đĩa. Luôn có một bản sao trên đĩa cho mỗi khối, song các bản sao trên đĩa của các khối là các phiên bản cũ hơn so với phiên bản trong buffer. Hệ thống con đảm trách cấp phát không gian buffer được gọi là bộ quản trị buffer.

BỘ QUẢN TRỊ BUFFER

Các chương trình trong một hệ CSDL đưa ra các yêu cầu cho bộ quản trị buffer khi chúng cần một khối đĩa. Nếu khối này đã sẵn sàng trong buffer, địa chỉ khối trong bộ nhớ chính được chuyển cho người yêu cầu. Nếu khối chưa có trong buffer, bộ quản trị buffer đầu tiên cấp phát không gian trong buffer cho khối, rút ra một số khối khác, nếu cần thiết, để lấy không gian cho khối mới. Khối được rút ra chỉ được viết lại trên đĩa khi nó có bị sửa đổi kể từ lần được viết lên đĩa gần nhất. Sau đó bộ quản trị buffer đọc khối từ đĩa vào buffer, và chuyển địa chỉ của khối trong bộ nhớ chính cho người yêu cầu. Bộ quản trị buffer không khác gì nhiều so với bộ quản trị bộ nhớ ảo, một điểm khác biệt là kích cỡ của một CSDL có thể rất lớn không đủ chứa toàn bộ trong bộ nhớ chính do vậy bộ quản trị buffer phải sử dụng các kỹ thuật tinh vi hơn các sơ đồ quản trị bộ nhớ ảo kiểu mẫu.

- **Chiến lược thay thế.** Khi không có chỗ trong buffer, một khối phải được xoá khỏi buffer trước khi một khối mới được đọc vào. Thông thường, hệ điều hành sử dụng sơ đồ LRU (Least Recently Used) để viết lên đĩa khối ít được dùng gần đây nhất, xoá bỏ nó khỏi buffer. Cách tiếp cận này có thể được cải tiến đối với ứng dụng CSDL.

- **Khối chốt (pinned blocks).** Để hệ CSDL có thể khôi phục sau sự cố, cần thiết phải hạn chế thời gian khi viết lại lên đĩa một khối. Một khối không cho phép viết lại lên đĩa được gọi là khối chốt.

- **Xuất ra bắt buộc các khối (Forced output of blocks).** Có những tình huống trong đó cần phải viết lại một khối lên đĩa, cho dù không gian buffer mà nó chiếm là không cần đến. Việc

viết này được gọi là *sự xuất ra bắt buộc của một khối*. Lý do ngắn gọn của yêu cầu xuất ra bắt buộc khối là nội dung của bộ nhớ chính bị mất khi có sự cố, ngược lại dữ liệu trên dĩa còn tồn tại sau sự cố.

CÁC ĐỐI SÁCH THAY THẾ BUFFER (Buffer-Replacement Policies).

Mục đích của chiến lược thay thế khối trong buffer là tối thiểu hoá các truy xuất đĩa. Các hệ điều hành thường sử dụng chiến lược LRU để thay thế khối. Tuy nhiên, một hệ CSDL có thể dự đoán mẫu tham khảo tương lai. Yêu cầu của một người sử dụng đối với hệ CSDL bao gồm một số bước. Hệ CSDL có thể xác định trước những khối nào sẽ là cần thiết bằng cách xem xét mỗi một trong các bước được yêu cầu để thực hiện hoạt động được yêu cầu bởi người sử dụng. Như vậy, khác với hệ điều hành, hệ CSDL có thể có thông tin liên quan đến tương lai, chí ít là tương lai gần. Trong nhiều trường hợp, chiến lược thay thế khối tối ưu cho hệ CSDL lại là MRU (Most Recently Used): Khối bị thay thế sẽ là khối mới được dùng gần đây nhất!

Bộ quản trị buffer có thể sử dụng thông tin thống kê liên quan đến xác suất mà một yêu cầu sẽ tham khảo một quan hệ riêng biệt nào đó. Tự điển dữ liệu là một trong những phần được truy xuất thường xuyên nhất của CSDL. Như vậy, bộ quản trị buffer sẽ không nên xoá các khối tự điển dữ liệu khỏi bộ nhớ chính trừ phi các nhân tố khác bức chế làm điều đó. Một chỉ mục (Index) đối với một file được truy xuất thường xuyên hơn chính bản thân file, vậy thì bộ quản trị buffer cũng không nên xoá khối chỉ mục khỏi bộ nhớ chính nếu có sự lựa chọn.

Chiến luợc thay thế khối CSDL lý tưởng cần hiểu biết về các hoạt động CSDL đang được thực hiện. Không một chiến lược đơn lẻ nào được biết nắm bắt được toàn bộ các viễn cảnh có thể. Tuy vậy, một điều đáng ngạc nhiên là phần lớn các hệ CSDL sử dụng LRU bất chấp các khuyết điểm của chiến lược đó.

Chiến lược được sử dụng bởi bộ quản trị buffer để thay thế khối bị ảnh hưởng bởi các nhân tố khác hơn là nhân tố thời gian tại đó khối được tham khảo trở lại. Nếu hệ thống đang xử lý các yêu cầu của một vài người sử dụng cạnh tranh, hệ thống (con) điều khiển cạnh tranh (concurrency-control subsystem) có thể phải làm trễ một số yêu cầu để đảm bảo tính nhất quán của CSDL. Nếu bộ quản trị buffer được cho các thông tin từ hệ thống điều khiển cạnh tranh mà nó nêu rõ những yêu cầu nào đang bị làm trễ, nó có thể sử dụng các thông tin này để thay đổi chiến lược thay thế khối của nó. Đặc biệt, các khối cần thiết bởi các yêu cầu tích cực (active requests) có thể được giữ lại trong buffer, toàn bộ các bất lợi đổ dồn lên các khối cần thiết bởi các yêu cầu bị làm trễ.

Hệ thống (con) khôi phục (crash-recovery subsystem) áp đặt các ràng buộc nghiêm nhặt lên việc thay thế khối. Nếu một khối bị sửa đổi, bộ quản trị buffer không được phép viết lại phiên bản mới của khối trong buffer lên đĩa, vì điều này phá huỷ phiên bản cũ. Thay vào đó, bộ quản trị khối phải tìm kiếm quyền từ hệ thống khôi phục trước khi viết khối. Hệ thống khôi phục có thể đòi hỏi một số khối nhất định khác là xuất bắt buộc (forced output) trước khi cấp quyền cho bộ quản trị buffer để xuất ra khối được yêu cầu.

TỔ CHỨC FILE

Một file được tổ chức logic như một dãy các mẩu tin (record). Các mẩu tin này được ánh xạ lên các khối đĩa. File được cung cấp như một xây dựng cơ sở trong hệ điều hành, như vậy ta sẽ giả thiết sự tồn tại của hệ thống file nền. Ta cần phải xét những phương pháp biểu diễn các mô hình dữ liệu logic trong thuật ngữ file.