Chương 1:

1.1 Ba mục đích chính của hệ điều hành là gì?

Trả lời:

• Cung cấp môi trường cho người dùng máy tính thực thi các chương trình trên phần cứng máy tính một cách thuận tiện và hiệu quả.

• Phân bổ các tài nguyên riêng biệt của máy tính khi cần thiết để giải quyết vấn đề được đưa ra. Quá trình phân bổ phải công bằng và hiệu quả nhất có thể.

• Là một chương trình điều khiển, nó phục vụ hai chức năng chính: (1) giám sát việc thực thi các chương trình của người dùng để ngăn ngừa lỗi và sử dụng máy tính không đúng cách, và (2) quản lý hoạt động và điều khiển các thiết bị I / O

1.2 Sự khác biệt chính giữa hệ điều hành cho máy tính lớn và máy tính cá nhân là gì?

Trả lời:

Nói chung, hệ điều hành cho hệ thống lô (hàng loạt) có yêu cầu đơn giản hơn so với máy tính cá nhân. Hệ thống hàng loạt không phải quan tâm đến việc tương tác với người dùng nhiều như máy tính cá nhân. Do đó, hệ điều hành cho PC phải quan tâm đến thời gian phản hồi cho người dùng tương tác. Hệ thống hàng loạt không có các yêu cầu như vậy. Một hệ thống theo lô thuần túy cũng có thể không xử lý việc chia sẻ thời gian, trong khi một hệ điều hành phải chuyển đổi nhanh chóng giữa các công việc khác nhau.

1.3 Liệt kê bốn bước cần thiết để chạy một chương trình trên một máy hoàn toàn chuyên dụng.

Trả lời:

a. Dự trữ thời gian máy.

b. Tải chương trình vào bộ nhớ theo cách thủ công.

C. Tải địa chỉ bắt đầu và bắt đầu thực hiện.

d. Giám sát và điều khiển việc thực thi chương trình từ bảng điều khiển.

1.4 Chúng tôi đã nhấn mạnh sự cần thiết của một hệ điều hành để sử dụng hiệu quả phần cứng máy tính. Khi nào thì thích hợp để hệ điều hành từ bỏ

nguyên tắc này và "lãng phí" tài nguyên? Tại sao một hệ thống như vậy không thực sự lãng phí?

Trả lời:

Hệ thống một người dùng nên sử dụng tối đa hệ thống cho người dùng. GUI có thể "lãng phí" chu kỳ CPU, nhưng nó tối ưu hóa sự tương tác của người dùng với hệ thống.

1.5 Khó khăn chính mà một lập trình viên phải vượt qua khi viết một hệ điều hành cho môi trường thời gian thực là gì?

Trả lời:

Khó khăn chính là giữ hệ điều hành trong các giới hạn thời gian cố định của hệ thống thời gian thực. Nếu hệ thống không hoàn thành một nhiệm vụ trong một khung thời gian nhất định, nó có thể gây ra sự cố cho toàn bộ hệ thống mà nó đang chạy. Do đó, khi viết hệ điều hành cho hệ thống thời gian thực, người viết phải chắc chắn rằng các lược đồ lập lịch của mình không cho phép thời gian phản hồi vượt quá giới hạn thời gian.

1.6 Xem xét các định nghĩa khác nhau của hệ điều hành. Xem xét liệu hệ điều hành có nên bao gồm các ứng dụng như trình duyệt Web và chương trình thư hay không. Tranh luận cả điều nên và điều gì không nên, và hỗ trợ câu trả lời của bạn.

Trả lời:

Đồng tình. Các ứng dụng như trình duyệt web và công cụ email đang ngày càng đóng vai trò quan trọng trong các hệ thống máy tính để bàn hiện đại. Để thực hiện vai trò này, chúng nên được kết hợp như một phần của hệ điều hành. Bằng cách đó, chúng có thể cung cấp hiệu suất tốt hơn và tích hợp tốt hơn với phần còn lại của hệ thống. Ngoài ra, các ứng dụng quan trọng này có thể có giao diện giống như phần mềm hệ điều hành.

Không đồng tình. Vai trò cơ bản của hệ điều hành là quản lý tài nguyên hệ thống như CPU, bộ nhớ, thiết bị I / O, v.v. Ngoài ra, vai trò của nó là chạy các ứng dụng phần mềm như trình duyệt web và ứng dụng email. Bằng cách kết hợp các ứng dụng như vậy vào hệ điều hành, chúng tôi tạo gánh nặng cho hệ điều hành với chức năng bổ sung. Gánh nặng như vậy có thể dẫn đến việc hệ điều hành thực hiện

công việc quản lý tài nguyên hệ thống kém khả quan. Ngoài ra, chúng tôi tăng kích thước của hệ điều hành do đó làm tăng khả năng hệ thống bị treo và vi phạm bảo mật.

1.7 Sự phân biệt giữa chế độ hạt nhân và chế độ người dùng hoạt động như một dạng hệ thống bảo vệ (bảo mật) thô sơ như thế nào?

Trả lời:

Sự khác biệt giữa chế độ hạt nhân và chế độ người dùng cung cấp một hình thức bảo vệ thô sơ theo cách sau. Một số lệnh chỉ có thể được thực thi khi CPU ở chế độ hạt nhân. Tương tự, các thiết bị phần cứng chỉ có thể được truy cập khi chương trình đang thực thi ở chế độ hạt nhân. Việc kiểm soát khi nào ngắt có thể bị ngắt hoặc bị vô hiệu hóa cũng chỉ có thể thực hiện được khi CPU ở chế độ hạt nhân.

Do đó, CPU có khả năng rất hạn chế khi thực thi ở chế độ người dùng, do đó thực thi bảo vệ các tài nguyên quan trọng.

1.8 Hướng dẫn nào sau đây nên được ưu tiên?

a. Đặt giá trị của bộ hẹn giờ. (bộ định thời)

b. Đọc đồng hồ.

C. Dọn dẹp bộ nhớ.

d. Đưa ra lệnh bẫy.

e. Tắt ngắt.

f. Sửa đổi các mục nhập trong bảng trạng thái thiết bị.

g. Chuyển từ chế độ người dùng sang chế độ hạt nhân.

h. Truy cập thiết bị I / O.

Trả lời: Các thao tác sau cần được đặc quyền: Đặt giá trị của bộ hẹn giờ, xóa bộ nhớ, tắt ngắt, sửa đổi các mục trong bảng trạng thái thiết bị, truy cập thiết bị I / O. Phần còn lại có thể được thực hiện ở chế độ người dùng.

1.9 Một số máy tính đời đầu đã bảo vệ hệ điều hành bằng cách đặt nó vào một phân vùng bộ nhớ mà công việc của người dùng hoặc chính hệ điều hành không thể sửa đổi được. Mô tả hai khó khăn mà bạn nghĩ có thể phát sinh với một kế hoạch như vậy.

Trả lời:

Dữ liệu được yêu cầu bởi hệ điều hành (mật khẩu, kiểm soát truy cập, thông tin kế toán, v.v.) sẽ phải được lưu trữ trong hoặc chuyển qua bộ nhớ không được bảo vệ và do đó người dùng trái phép có thể truy cập được.

1.10 Một số CPU cung cấp nhiều hơn hai chế độ hoạt động. Hai cách sử dụng có thể có của nhiều chế độ này là gì?

Trả lời:

Mặc dù hầu hết các hệ thống chỉ phân biệt giữa chế độ người dùng và hạt nhân, một số CPU đã hỗ trợ nhiều chế độ. Nhiều chế độ có thể được sử dụng để cung cấp chính sách bảo mật chi tiết hơn. Ví dụ, thay vì phân biệt giữa chế độ chỉ người dùng và hạt nhân, bạn có thể phân biệt giữa các loại chế độ người dùng khác nhau. Có lẽ những người dùng thuộc cùng một nhóm có thể thực thi mã của nhau. Máy sẽ chuyển sang chế độ được chỉ định khi một trong những người dùng này đang chạy mã. Khi máy ở chế độ này, một thành viên của

nhóm có thể chạy mã của bất kỳ ai khác trong nhóm.

Một khả năng khác là cung cấp các phân biệt khác nhau trong mã nhân. Ví dụ: một chế độ cụ thể có thể cho phép trình điều khiển thiết bị USB chạy. Điều này có nghĩa là các thiết bị USB có thể được bảo dưỡng mà không cần phải chuyển sang chế độ hạt nhân, do đó về cơ bản cho phép trình điều khiển thiết bị USB chạy ở chế độ bán người dùng / hạt nhân.

1.11 Bộ hẹn giờ có thể được sử dụng để tính thời gian hiện tại. Cung cấp một mô tả ngắn về cách có thể thực hiện điều này.

Trả lời:

Một chương trình có thể sử dụng cách tiếp cận sau để tính thời gian hiện tại bằng cách sử dụng ngắt bộ định thời. Chương trình có thể đặt bộ đếm thời gian trong

tương lai và chuyển sang chế độ ngủ. Khi nó được đánh thức bởi ngắt, nó có thể cập nhật trạng thái cục bộ của nó, trạng thái mà nó đang sử dụng để theo dõi số lượng ngắt, nó đã nhận được cho đến nay. Sau đó, nó có thể lặp lại quá trình này liên tục thiết lập các ngắt hẹn giờ và cập nhật trạng thái cục bộ của nó khi các ngắt thực sự được nâng lên.

1.12 Internet là mạng LAN hay mạng WAN?

Trả lời: Internet là một mạng WAN vì các máy tính khác nhau được đặt ở các vị trí địa lý khác nhau và được kết nối bằng các liên kết mạng đường dài

Chương 2:

2.1 Mục đích của lệnh gọi hệ thống là gì?

Trả lời:

Lệnh gọi hệ thống cho phép các tiến trình cấp người dùng yêu cầu các dịch vụ của hệ điều hành.

2.2 Năm hoạt động chính của một hệ điều hành liên quan đến quản lý quá trình là gì?

Trả lời:

a. Việc tạo và xóa cả quá trình của người dùng và hệ thống

b. Việc tạm ngừng và tiếp tục các tiến trình

C. Việc cung cấp các cơ chế để đồng bộ hóa tiến trình

d. Cung cấp các cơ chế để giao tiếp tiến trình

e. Cung cấp các cơ chế để xử lý bế tắc

2.3 Ba hoạt động chính của hệ điều hành liên quan đến quản lý bộ nhớ là gì?

Trả lời:

a. Theo dõi những phần nào của bộ nhớ hiện đang được sử dụng và bởi ai.

b. Quyết định quy trình nào sẽ được tải vào bộ nhớ khi có dung lượng bộ nhớ.

c. Phân bổ và phân bổ không gian bộ nhớ khi cần thiết.

2.4 Ba hoạt động chính của hệ điều hành liên quan đến quản lý bộ nhớ thứ cấp là gì?

Trả lời:

• Quản lý không gian trống.

• Phân bổ lưu trữ.

• Lập lịch đĩa.

2.5 Mục đích của trình thông dịch lệnh là gì? Tại sao nó thường tách biệt với hạt nhân?

Trả lời: Nó đọc các lệnh từ người dùng hoặc từ một tệp lệnh và thực thi chúng, thường bằng cách chuyển chúng thành một hoặc nhiều lệnh gọi hệ thống. Nó thường không phải là một phần của hạt nhân vì trình thông dịch lệnh có thể thay đổi.

2.6 Lệnh gọi hệ thống nào phải được thực thi bởi trình thông dịch lệnh hoặc trình bao để bắt đầu một tiến trình mới?

Trả lời:

Trong hệ thống Unix, một lệnh gọi hệ thống rẽ nhánh theo sau một lệnh gọi hệ thống thực thi cần được thực hiện để bắt đầu một quy trình mới. Cuộc gọi fork sao chép quy trình hiện đang thực thi, trong khi lệnh gọi thực thi phủ lên một quy trình mới dựa trên một quy trình thực thi khác trên quy trình gọi.

2.7 Mục đích của các chương trình hệ thống là gì?

Trả lời: Các chương trình hệ thống có thể được coi là một gói các lệnh gọi hệ thống hữu ích. Chúng cung cấp chức năng cơ bản cho người dùng để người dùng

không cần phải viết chương trình của riêng mình để giải quyết các vấn đề thông thường.

2.8 Ưu điểm chính của cách tiếp cận phân lớp đối với thiết kế hệ thống là gì? Nhược điểm của việc sử dụng phương pháp phân lớp là gì?

Trả lời:

Như trong tất cả các trường hợp thiết kế mô-đun, thiết kế hệ điều hành theo cách mô-đun có một số ưu điểm. Hệ thống dễ gỡ lỗi và sửa đổi hơn vì các thay đổi chỉ ảnh hưởng đến các phần giới hạn của hệ thống chứ không phải chạm vào tất cả các phần của hệ điều hành. Thông tin chỉ được lưu giữ ở những nơi cần thiết và chỉ có thể truy cập được trong một khu vực được xác định và hạn chế, vì vậy bất kỳ lỗi nào ảnh hưởng đến dữ liệu đó phải được giới hạn trong một mô-đun hoặc lớp cụ thể.

2.9 Liệt kê năm dịch vụ được cung cấp bởi một hệ điều hành. Giải thích cách mỗi thứ cung cấp sự tiện lợi cho người dùng. Cũng giải thích những trường hợp nào thì các chương trình cấp người dùng sẽ không thể cung cấp các dịch vụ này.

Trả lời:

a. Thực hiện chương trình. Hệ điều hành tải nội dung (hoặc các phần) của tệp vào bộ nhớ và bắt đầu thực thi. Không thể tin cậy một chương trình cấp người dùng để phân bổ thời gian CPU đúng cách.

b. Hoạt động I / O. Đĩa, băng, đường truyền nối tiếp và các thiết bị khác phải được giao tiếp ở mức rất thấp. Người dùng chỉ cần chỉ định thiết bị và thao tác thực hiện trên thiết bị đó, trong khi hệ thống chuyển đổi yêu cầu đó thành các lệnh dành riêng cho thiết bị hoặc bộ điều khiển. Không thể tin cậy các chương trình cấp người dùng để chỉ truy cập vào các thiết bị mà họ phải có quyền truy cập và chỉ truy cập chúng khi chúng không được sử dụng.

c. Thao tác với hệ thống tệp. Có nhiều chi tiết trong việc tạo, xóa, cấp phát và đặt tên tệp mà người dùng không cần phải thực hiện. Các khối không gian đĩa được sử dụng bởi các tệp và phải được theo dõi. Xóa tệp yêu cầu xóa thông tin tệp tin và giải phóng các khối được cấp phát. Các biện pháp bảo vệ cũng phải được

kiểm tra để đảm bảo quyền truy cập tệp thích hợp. Các chương trình người dùng không thể đảm bảo tuân thủ các phương pháp bảo vệ cũng như không được tin cậy để chỉ cấp phát các khối miễn phí và phân bổ khối khi xóa tệp.

d. Thông tin liên lạc. Việc truyền thông điệp giữa các hệ thống yêu cầu các thông điệp phải được chuyển thành các gói thông tin, được gửi đến bộ điều khiển mạng, được truyền qua một phương tiện truyền thông và được hệ thống đích tập hợp lại. Đặt hàng gói và sửa dữ liệu phải diễn ra. Một lần nữa, các chương trình người dùng có thể không điều phối quyền truy cập vào thiết bị mạng hoặc chúng có thể nhận các gói dành cho các quá trình khác.

e. Phát hiện lỗi. Phát hiện lỗi xảy ra ở cả cấp độ phần cứng và phần mềm. Ở cấp độ phần cứng, tất cả các quá trình truyền dữ liệu phải được kiểm tra để đảm bảo rằng dữ liệu không bị hỏng trong quá trình truyền. Tất cả dữ liệu trên phương tiện phải được kiểm tra để chắc chắn rằng chúng không thay đổi kể từ khi chúng được ghi vào phương tiện. Ở cấp độ phần mềm, phương tiện phải được kiểm tra tính nhất quán của dữ liệu; chẳng hạn, liệu số lượng khối lưu trữ được phân bổ và chưa được phân bổ có khớp với tổng số trên thiết bị hay không. Ở đó, các lỗi thường không phụ thuộc vào quy trình (ví dụ, dữ liệu bị hỏng trên đĩa), vì vậy phải có một chương trình toàn cầu (hệ điều hành) xử lý tất cả các loại lỗi

2.10 Mục đích của lệnh gọi hệ thống là gì?

Trả lời:

Các cuộc gọi hệ thống cho phép các quy trình cấp người dùng yêu cầu các dịch vụ của hệ điều hành.

2.11 Ưu điểm chính của phương pháp microkernel đối với thiết kế hệ thống là gì?

Trả lời:

Các lợi ích thường bao gồm những lợi ích sau (a) thêm một dịch vụ mới không yêu cầu sửa đổi hạt nhân, (b) nó an toàn hơn vì nhiều hoạt động được thực hiện ở chế độ người dùng hơn ở chế độ hạt nhân và (c) thiết kế và chức năng hạt nhân đơn giản hơn thường dẫn đến một hệ điều hành đáng tin cậy hơn.

2.12 Tại sao một số hệ thống lưu trữ hệ điều hành trong phần sụn và những hệ thống khác trên đĩa?

Trả lời:

Đối với một số thiết bị nhất định, chẳng hạn như PDA cầm tay và điện thoại di động, đĩa có hệ thống tệp có thể không khả dụng cho thiết bị. Trong tình huống này, hệ điều hành phải được lưu trữ trong phần sụn.

2.13 Làm thế nào một hệ thống có thể được thiết kế để cho phép lựa chọn hệ điều hành để khởi động từ đó? Chương trình bootstrap cần làm gì?

Trả lời:

Hãy xem xét một hệ thống muốn chạy cả Windows XP và ba bản phân phối khác nhau của Linux (ví dụ: RedHat, Debian và Mandrake). Mỗi hệ điều hành sẽ được lưu trữ trên đĩa. Trong quá trình khởi động hệ thống, một chương trình đặc biệt (chúng ta sẽ gọi là trình quản lý khởi động) sẽ xác định hệ điều hành nào cần khởi động. Điều này có nghĩa là thay vì khởi động ban đầu vào hệ điều hành, trình quản lý khởi động sẽ chạy trước tiên trong quá trình khởi động hệ thống. Chính trình quản lý khởi động này chịu trách nhiệm xác định hệ thống sẽ khởi động vào. Thông thường, các trình quản lý khởi động phải được lưu trữ tại một số vị trí nhất định của đĩa cứng để được nhận dạng trong quá trình khởi động hệ thống. Trình quản lý khởi động thường cung cấp cho người dùng lựa chọn hệ thống để khởi động vào; trình quản lý khởi động cũng thường được thiết kế để khởi động vào hệ điều hành mặc định nếu người dùng không có lựa chọn nào được chọn.

Chương 3

3.1 Palm OS không cung cấp phương tiện xử lý đồng thời. Thảo luận về ba phức tạp chính mà xử lý đồng thời thêm vào hệ điều hành.

Trả lời:

a. Một phương pháp chia sẻ thời gian phải được thực hiện để cho phép từng quy trình trong số một số quy trình có quyền truy cập vào hệ thống. Phương pháp này liên quan đến việc ưu tiên các quá trình không tự nguyện từ bỏ CPU (ví dụ: bằng cách sử dụng lệnh gọi hệ thống) và hạt nhân được đưa vào lại (vì vậy nhiều quá trình có thể đang thực thi mã hạt nhân đồng thời).

b. Các quy trình và tài nguyên hệ thống phải có các biện pháp bảo vệ và phải được bảo vệ lẫn nhau. Bất kỳ quy trình nhất định nào cũng phải bị giới hạn về dung lượng bộ nhớ mà nó có thể sử dụng và các hoạt động mà nó có thể thực hiện trên các thiết bị như đĩa.

c. Cần phải cẩn thận trong nhân để ngăn chặn bế tắc giữa các quy trình, vì vậy các quy trình không chờ đợi các tài nguyên được phân bổ của nhau.

3.2 Bộ xử lý Sun UltraSPARC có nhiều bộ thanh ghi. Mô tả các hành động của một chuyển đổi ngữ cảnh nếu ngữ cảnh mới đã được tải vào một trong các tập đăng ký. Điều gì khác phải xảy ra nếu ngữ cảnh mới nằm trong bộ nhớ chứ không phải trong một tập thanh ghi và tất cả các tập thanh ghi đang được sử dụng?

Trả lời:

Con trỏ bộ đăng ký hiện tại của CPU được thay đổi để trỏ đến bộ chứa ngữ cảnh mới, điều này mất rất ít thời gian. Nếu ngữ cảnh nằm trong bộ nhớ, một trong các ngữ cảnh trong tập thanh ghi phải được chọn và được chuyển vào bộ nhớ, và ngữ cảnh mới phải được tải từ bộ nhớ vào tập hợp. Quá trình này mất nhiều thời gian hơn một chút so với các hệ thống có một bộ thanh ghi, tùy thuộc vào cách chọn nạn nhân thay thế.

3.3 Khi một tiến trình tạo một tiến trình mới bằng thao tác fork (), trạng thái nào sau đây được chia sẻ giữa tiến trình mẹ và tiến trình con?

a. Cây rơm

b. Đống

C. Phân đoạn bộ nhớ được chia sẻ

Trả lời:

Chỉ các phân đoạn bộ nhớ được chia sẻ được chia sẻ giữa quy trình mẹ và quy trình con mới được phân nhánh. Các bản sao của ngăn xếp và đóng được tạo cho quá trình mới được tạo.

3.4 Một lần nữa xem xét cơ chế RPC, hãy xem xét ngữ nghĩa "chính xác một lần". Thuật toán triển khai ngữ nghĩa này có thực thi chính xác ngay cả khi thông báo "ACK" gửi lại máy khách bị mất do mạng không

vấn đề? Mô tả chuỗi tin nhắn và liệu "chính xác một lần" có còn được lưu giữ hay không.

Trả lời:

Ngữ nghĩa "chính xác một lần" đảm bảo rằng một thủ tục từ xa sẽ được thực thi chính xác một lần và chỉ một lần. Thuật toán chung để đảm bảo điều này kết hợp lược đồ xác nhận (ACK) kết hợp với dấu thời gian (hoặc một số bộ đếm gia tăng khác cho phép máy chủ phân biệt giữa các thông báo trùng lặp).

Chiến lược chung là máy khách gửi RPC đến máy chủ cùng với dấu thời gian. Máy khách cũng sẽ bắt đầu đồng hồ thời gian chờ. Sau đó, máy khách sẽ đợi một trong hai lần xuất hiện: (1) nó sẽ nhận được ACK từ máy chủ cho biết rằng thủ tục từ xa đã được thực hiện hoặc (2) nó sẽ hết thời gian. Nếu máy khách hết thời gian chờ, nó giả sử máy chủ không thể thực hiện thủ tục từ xa, do đó máy khách gọi RPC lần thứ hai, gửi một dấu thời gian sau đó. Máy khách có thể không nhận được ACK vì một trong hai lý do: (1) RPC ban đầu không bao giờ được máy chủ nhận, hoặc (2) RPC đã được nhận chính xác — và được thực hiện — bởi máy chủ nhưng ACK đã bị mất. Trong tình huống (1), việc sử dụng ACK cho phép máy chủ cuối cùng nhận và thực hiện RPC. Trong tình huống (2), máy chủ sẽ nhận được một RPC trùng lặp và nó sẽ sử dụng dấu thời gian để xác định nó là một bản sao để không thực hiện RPC lần thứ hai. Điều quan trọng cần lưu ý là máy chủ phải gửi lại ACK thứ hai cho máy khách để thông báo cho máy khách rằng RPC đã được thực hiện.

3.5 Giả sử rằng một hệ thống phân tán dễ bị lỗi máy chủ. Những cơ chế nào sẽ được yêu cầu để đảm bảo ngữ nghĩa "chính xác một lần" để thực thi RPC?

Trả lời:

Máy chủ phải theo dõi thông tin lưu trữ ổn định (chẳng hạn như nhật ký đĩa) về những gì các hoạt động RPC đã nhận được, liệu chúng có được thực hiện thành công hay không và kết quả liên quan đến các hoạt động. Khi xảy ra sự cố máy chủ và nhận được thông báo RPC, máy chủ có thể kiểm tra xem RPC đã được thực hiện trước đó hay chưa và do đó đảm bảo ngữ nghĩa "chính xác một lần" cho thực thi các RPC.

Chương 4

4.1 Cung cấp hai ví dụ lập trình trong đó đa luồng cung cấp hiệu suất tốt hơn giải pháp đơn luồng.

Trả lời:

(1) một máy chủ web phục vụ từng yêu cầu trong một chuỗi riêng biệt.

2) (Một ứng dụng song song như phép nhân ma trận trong đó (các phần khác nhau của ma trận có thể được thực hiện song song.

(3) Một (chương trình GUI tương tác chẳng hạn như trình gỡ lỗi trong đó một luồng được sử dụng (để giám sát đầu vào của người dùng, một luồng khác đại diện cho việc chạy (ứng dụng và luồng thứ ba giám sát hiệu suất).

4.2 Sự khác biệt giữa luồng cấp người dùng và luồng cấp nhân là gì? Trong những trường hợp nào thì loại này tốt hơn loại kia?

Trả lời:

(1) Hạt nhân không xác định được các luồng cấp người dùng, trong khi hạt nhân nhận biết các luồng nhân. (2) Trên các hệ thống sử dụng ánh xạ M: 1 hoặc M: N, các luồng người dùng được lập lịch bởi thư viện luồng và hạt nhân lập lịch cho các luồng nhân. (3) Các luồng nhân không cần phải được liên kết với một quy trình trong khi mọi luồng người dùng thuộc về một quy trình. Các luồng nhân thường đắt hơn để duy trì các luồng người dùng vì chúng phải được biểu diễn bằng cấu trúc dữ liệu nhân.

4.3 Mô tả các hành động được thực hiện bởi một nhân để chuyển đổi ngữ cảnh giữa các luồng cấp nhân.

Trả lời:

Việc chuyển đổi ngữ cảnh giữa các luồng nhân thường yêu cầu lưu giá trị của các thanh ghi CPU từ luồng đang được chuyển ra ngoài và khôi phục các thanh ghi CPU của luồng mới đang được lên lịch.

4.4 Những tài nguyên nào được sử dụng khi một luồng được tạo? Chúng khác với những gì được sử dụng khi một quy trình được tạo ra?

Trả lời:

Bởi vì một luồng nhỏ hơn một quy trình, việc tạo luồng thường sử dụng ít tài nguyên hơn so với tạo quy trình. Tạo một quy trình yêu cầu phân bổ một khối điều khiển quy trình (PCB), một cấu trúc dữ liệu khá lớn. PCB bao gồm một bản đồ bộ nhớ, danh sách các tệp đang mở và các biến môi trường. Phân bổ và quản lý bản đồ bộ nhớ thường là

hoạt động tốn nhiều thời gian nhất. Việc tạo luồng người dùng hoặc nhân bao gồm việc phân bổ cấu trúc dữ liệu nhỏ để chứa tập thanh ghi, ngăn xếp và mức độ ưu tiên.

4.5 Giả sử một hệ điều hành ánh xạ các luồng cấp người dùng tới hạt nhân bằng cách sử dụng mô hình nhiều-nhiều và việc ánh xạ được thực hiện thông qua các LWP. Hơn nữa, hệ thống cho phép các nhà phát triển tạo các luồng thời gian thực. Có cần thiết phải liên kết một chuỗi thời gian thực với một LWP không? Giải thích.

Trả lời:

Vâng. Thời gian là rất quan trọng đối với các ứng dụng thời gian thực. Nếu một luồng được đánh dấu là thời gian thực nhưng không bị ràng buộc với một LWP, thì luồng đó có thể phải đợi để được gắn vào một LWP trước khi chạy. Xem xét nếu một luồng thời gian thực đang chạy (được gắn với một LWP) và sau đó tiến hành chặn (tức là phải thực hiện I / O, đã được ưu tiên bởi một luồng thời gian thực có mức độ ưu tiên cao hơn, đang chờ khóa loại trừ lẫn nhau, vv) Trong khi luồng thời gian thực bị chặn, LWP mà nó được gắn vào đã được gán cho một luồng khác. Khi luồng thời gian thực đã được lên lịch để chạy lại, trước tiên nó phải đợi để được gắn vào một LWP. Bằng cách liên kết một LWP với một luồng thời gian thực, bạn đảm bảo rằng luồng sẽ có thể chạy với độ trễ tối thiểu khi nó được lên lịch.

4.6 Một chương trình Pthread thực hiện chức năng tổng kết đã được cung cấp trong Phần 4.3.1. Viết lại chương trình này bằng Java.

Trả lời: Vui lòng tham khảo trang web hỗ trợ để biết giải pháp mã nguồn

Chương 5

5.1 Thuật toán lập lịch CPU xác định thứ tự thực hiện các quy trình đã lên lịch của nó. Cho n quá trình được lên lịch trên một bộ xử lý, có bao nhiêu lịch trình khác nhau có thể có? Đưa ra công thức về n.

Trả lời:

n! (n giai thừa = n × n - 1 × n - 2 × ... × 2 × 1).

5.2 Xác định sự khác biệt giữa lập kế hoạch ưu tiên và không đặt trước.

Trả lời:

Lập lịch trước cho phép một quá trình bị gián đoạn khi đang thực thi, lấy CPU đi và phân bổ nó cho một quá trình khác. Lập lịch không ưu tiên đảm bảo rằng một quá trình chỉ từ bỏ quyền kiểm soát CPU khi nó kết thúc với sự bùng nổ CPU hiện tại.

5.3 Giả sử rằng các quy trình sau đây được thực thi vào thời điểm

chỉ ra. Mỗi quá trình sẽ chạy trong khoảng thời gian được liệt kê. Khi trả lời các câu hỏi, hãy sử dụng lập lịch trình không tính trước và dựa trên tất cả các quyết định dựa trên thông tin bạn có tại thời điểm phải đưa ra quyết định.

Quy trình Thời gian Đến Thời gian Burst

P1 0,0 8

P2 0,4 4

P3 1.0 1

Một. Thời gian quay vòng trung bình cho các quy trình này với thuật toán lập lịch FCFS là bao nhiêu?

b. Thời gian quay vòng trung bình cho các quy trình này với thuật toán lập lịch SJF là bao nhiêu?

C. Thuật toán SJF được cho là sẽ cải thiện hiệu suất, nhưng lưu ý rằng chúng tôi đã chọn chạy quy trình P1 tại thời điểm 0 vì chúng tôi không biết rằng hai quy trình ngắn hơn sẽ đến sớm. Tính toán thời gian quay vòng trung bình sẽ là bao nhiêu nếu

CPU không hoạt động trong 1 đơn vị đầu tiên và sau đó lập lịch SJF được sử dụng. Hãy nhớ rằng các quá trình P1 và P2 đang chờ trong thời gian nhàn rỗi này, vì vậy thời gian chờ của chúng có thể tăng lên. Thuật toán này có thể được gọi là lập lịch tri thức trong tương lai.

Trả lời:

a. 10,53

b. 9.53

C. 6,86

Hãy nhớ rằng thời gian quay vòng là thời gian kết thúc trừ thời gian đến, vì vậy bạn phải trừ thời gian đến để tính thời gian quay vòng. FCFS là 11 nếu bạn quên trừ thời gian đến.

5.4 Có lợi thế gì khi có các kích thước lượng tử thời gian khác nhau ở các cấp độ khác nhau của hệ thống xếp hàng đa cấp?

Trả lời:

Các quy trình cần được bảo dưỡng thường xuyên hơn, chẳng hạn như các quy trình tương tác như trình chỉnh sửa, có thể nằm trong hàng đợi với lượng tử thời gian nhỏ. Các quy trình không cần bảo dưỡng thường xuyên có thể nằm trong hàng đợi với lượng tử lớn hơn, yêu cầu ít công tắc ngữ cảnh hơn để hoàn thành quá trình xử lý và do đó sử dụng máy tính hiệu quả hơn.

5.5 Nhiều thuật toán lập lịch trình CPU được tham số hóa. Ví dụ, thuật toán RR yêu cầu một tham số để chỉ ra lát thời gian. Hàng đợi phản hồi đa cấp yêu cầu các tham số để xác định số lượng hàng đợi, các thuật toán lập lịch cho mỗi hàng đợi, các tiêu chí được sử dụng để di chuyển các quy trình giữa các hàng đợi, v.v. Do đó, những thuật toán này thực sự là một tập hợp các thuật toán (ví dụ, tập hợp các thuật toán RR cho mọi lát cắt thời gian, v.v.). Một bộ thuật toán có thể bao gồm một bộ thuật toán khác (ví dụ, thuật toán FCFS là thuật toán RR với lượng tử thời gian vô hạn). Mối quan hệ nào (nếu có) giữ giữa các cặp tập hợp thuật toán sau đây?

a. Ưu tiên và SJF

b. Hàng đợi phản hồi đa cấp và FCFS

C. Ưu tiên và FCFS

d. RR và SJF

Trả lời:

a. Công việc ngắn nhất có mức độ ưu tiên cao nhất.

b. Cấp thấp nhất của MLFQ là FCFS.

C. FCFS dành ưu tiên cao nhất cho công việc đã tồn tại

dài nhất.

d. Không có.

5.6 Giả sử rằng một thuật toán lập lịch (ở cấp độ lập lịch CPU ngắn hạn) ủng hộ những quá trình đã sử dụng ít thời gian xử lý nhất trong quá khứ gần đây. Tại sao thuật toán này lại ưu tiên các chương trình ràng buộc I / O nhưng lại không bỏ đói các chương trình ràng buộc CPU vĩnh viễn?

Trả lời:

Nó sẽ ưu tiên các chương trình ràng buộc I / O vì yêu cầu cụm CPU tương đối ngắn của chúng; tuy nhiên, các chương trình ràng buộc CPU sẽ không chết đói vì các chương trình ràng buộc I / O sẽ tương đối thường xuyên rời bỏ CPU để thực hiện I / O của chúng.

5.7 Phân biệt giữa lập lịch PCS và SCS.

Trả lời:

Lập lịch PCS được thực hiện cục bộ cho quá trình. Đó là cách thư viện luồng lên lịch cho các luồng trên các LWP có sẵn. Lập lịch SCS là tình huống hệ điều hành lập lịch cho các luồng nhân. Trên các hệ thống sử dụng nhiều-một hoặc nhiều-nhiều, hai mô hình lập kế hoạch về cơ bản là khác nhau. Trên các hệ thống sử dụng một-một, PCS và SCS đều giống nhau.

5.8 Giả sử một hệ điều hành ánh xạ các luồng cấp người dùng tới hạt nhân bằng cách sử dụng mô hình nhiều-nhiều trong đó ánh xạ được thực hiện thông qua việc

sử dụng LWP. Hơn nữa, hệ thống cho phép các nhà phát triển chương trình tạo các luồng thời gian thực. Có cần thiết phải liên kết một chuỗi thời gian thực với một LWP không?

Trả lời:

Có, nếu không thì một luồng người dùng có thể phải cạnh tranh để giành được một LWP có sẵn trước khi được lên lịch thực sự. Bằng cách liên kết luồng người dùng với một LWP, không có độ trễ khi chờ đợi một LWP có sẵn; luồng người dùng thời gian thực có thể được lên lịch ngay lập tức.

Chương 6

6.1. Trong Phần 6.4, chúng tôi đã đề cập rằng việc tắt ngắt thường xuyên có thể ảnh hưởng đến đồng hồ của hệ thống. Giải thích tại sao nó có thể và làm thế nào những tác động như vậy có thể được giảm thiểu.
Trả lời: Đồng hồ hệ thống được cập nhật tại mỗi lần ngắt đồng hồ. Nếu ngắt bị vô hiệu hóa — đặc biệt là trong một khoảng thời gian dài — thì đồng hồ hệ thống có thể dễ dàng bị mất thời gian chính xác. Đồng hồ hệ thống cũng được sử dụng cho mục đích lập lịch trình. Ví dụ, thời gian lượng tử cho một quá trình được biểu thị bằng một số tích tắc đồng hồ. Ở mọi đồng hồ ngắt, bộ lập lịch xác định xem lượng tử thời gian cho hiện đang chạy quá trình đã hết hạn. Nếu ngắt đồng hồ bị tắt, bộ lập lịch không thể chỉ định chính xác lượng tử thời gian. Hiệu ứng này có thể được giảm thiểu bằng cách tắt ngắt đồng hồ chỉ trong khoảng thời gian rất ngắn.

6.2. Vấn đề người hút thuốc lá. Hãy xem xét một hệ thống có ba quy trình hút thuốc và một quy trình đại lý. Mỗi người hút liên tục cuộn một điếu và sau đó hút nó. Nhưng để cuộn và hút một điếu thuốc, người hút cần ba thành phần: thuốc lá, giấy và diêm. Một trong các quy trình của người hút thuốc có giấy, quy trình khác có thuốc lá, và quy trình thứ ba có diêm. Các đại lý có một nguồn cung cấp vô hạn của cả ba vật liệu. Đại lý đặt hai trong số các thành phần trên bàn. Người hút thuốc còn lại sau đó thành phần tạo ra và hút một điếu thuốc, báo hiệu cho tác nhân trên hoàn thành. Sau đó, đại lý đưa ra hai trong số ba thành phần khác, và chu trình lặp lại. Viết chương trình để đồng bộ hóa tác nhân và những người hút thuốc bằng cách sử dụng đồng bộ hóa Java.
Trả lời: Vui lòng tham khảo trang web supports Web để biết giải pháp mã nguồn.

6.3. Đưa ra lý do tại sao Solaris, Windows XP và Linux thực hiện nhiều cơ chế khóa. Mô tả các trường hợp mà họ sử dụng spinlocks, mutexes, semaphores, mutex

thích ứng và điều kiện biến. Trong mỗi trường hợp, hãy giải thích lý do tại sao cơ chế đó là cần thiết.

Trả lời: Các hệ điều hành này cung cấp các cơ chế khóa khác nhau tùy thuộc vào nhu cầu của nhà phát triển ứng dụng. Spinlocks là hữu ích cho các hệ thống đa xử lý nơi một luồng có thể chạy trong một vòng lặp bận (trong một thời gian ngắn) thay vì phải gánh chịu chi phí xếp hàng chờ ngủ. Mutexes rất hữu ích để khóa tài nguyên. Solaris 2 sử dụng mutex thích ứng, nghĩa là mutex được triển khai với khóa quay trên máy đa xử lý. Semaphores và biến điều kiện là những công cụ thích hợp hơn để đồng bộ hóa khi tài nguyên phải được giữ trong một thời gian dài, vì kéo sợi không hiệu quả đối với thời gian dài.

6.4. Giải thích sự khác biệt, về mặt chi phí, giữa ba loại lưu trữ dễ bay hơi, không bay hơi và ổn định.

Trả lời:

Bộ nhớ đa năng đề cập đến bộ nhớ chính và bộ nhớ đệm và rất Nhanh. Tuy nhiên, lưu trữ dễ bay hơi không thể tồn tại khi hệ thống gặp sự cố hoặc tắt nguồn hệ thống. Bộ nhớ không linh hoạt vẫn tồn tại sự cố hệ thống và hệ thống ngắt điện. Đĩa và băng là những ví dụ về tính không bay hơi kho. Gần đây, các thiết bị USB sử dụng bộ nhớ chỉ đọc chương trình có thể xóa được (EPROM) đã xuất hiện cung cấp khả năng lưu trữ không linh hoạt. Lưu trữ ổn định đề cập đến việc lưu trữ về mặt kỹ thuật không bao giờ có thể bị mất vì có các bản sao lưu dự phòng của dữ liệu (thường là trên đĩa).

6.5. Giải thích mục đích của cơ chế điểm kiểm tra. Bao lâu nên các trạm kiểm soát được thực hiện? Mô tả tần suất của các điểm kiểm tra như thế nào ảnh hưởng đến:

• Hiệu suất hệ thống khi không xảy ra lỗi
• Thời gian cần để khôi phục sau sự cố hệ thống
• Thời gian cần để khôi phục sau sự cố đĩa

Trả lời:

Bản ghi nhật ký điểm kiểm tra chỉ ra rằng một bản ghi nhật ký và dữ liệu sửa đổi đã được ghi vào bộ nhớ ổn định và giao dịch không cần phải làm lại trong trường hợp hệ thống gặp sự cố. Rõ ràng, thường xuyên hơn các điểm kiểm tra được thực hiện, càng ít có khả năng xảy ra các bản cập nhật thừa sẽ phải được thực hiện trong quá trình khôi phục.

• Hiệu suất hệ thống khi không xảy ra lỗi — Nếu không có lỗi nào xảy ra, hệ thống phải chịu chi phí thực hiện các trạm kiểm soát về cơ bản là không cần thiết. Trong tình huống này, thực hiện các trạm kiểm soát ít thường xuyên hơn sẽ dẫn đến hiệu suất hệ thống tốt hơn.

• Thời gian cần thiết để khôi phục sau sự cố hệ thống — Sự tồn tại của

bản ghi điểm kiểm tra có nghĩa là một hoạt động sẽ không phải làm lại trong quá trình khôi phục hệ thống. Trong tình huống này, thường xuyên hơn các điểm kiểm tra đã được thực hiện, thời gian khôi phục càng nhanh từ sự cố hệ thống.

• Thời gian cần để khôi phục sau sự cố đĩa — Sự tồn tại của bản ghi điểm kiểm tra có nghĩa là một hoạt động sẽ không phải làm lại trong quá trình khôi phục hệ thống. Trong tình huống này, thường xuyên hơn các điểm kiểm tra đã được thực hiện, thời gian khôi phục càng nhanh từ sự cố đĩa.

6.6. Giải thích khái niệm nguyên tử giao dịch.

Trả lời:

Một giao dịch là một chuỗi các thao tác đọc và ghi dựa trên một số dữ liệu được theo sau bởi một hoạt động cam kết. Nếu chuỗi hoạt động trong một giao dịch không thể được hoàn thành, giao dịch phải được hủy bỏ và các hoạt động đã diễn ra phải được quay trở lại. Nó quan trọng rằng chuỗi hoạt động trong một giao dịch xuất hiện dưới dạng một hoạt động để đảm bảo tính toàn vẹn của dữ liệu được cập nhật. Nếu không thì, dữ liệu có thể bị xâm phạm nếu hoạt động từ hai (hoặc nhiều) khác nhau các giao dịch được trộn lẫn với nhau.

6.7. Cho thấy rằng một số lịch trình có thể thực hiện được theo khóa hai giai đoạn nhưng không thể thực hiện được theo giao thức dấu thời gian và ngược lại.

Trả lời:

Lịch trình được phép trong giao thức khóa hai pha nhưng không có trong giao thức dấu thời gian là:

Lịch biểu này không được phép trong giao thức dấu thời gian vì ở bước 7, Dấu thời gian W của B là 1.

Lịch biểu được phép trong giao thức dấu thời gian nhưng không được phép trong giao thức khóa hai pha là:

bước	T0	T1	T2
1	ghi (A)		
2		ghi (A)	
3			ghi (A)
4	ghi (B)		
5		ghi (B)	

Lịch trình này không thể có hướng dẫn khóa được thêm vào để làm cho nó hợp pháp theo giao thức khóa hai pha vì T1 phải mở khóa (A) giữa các bước 2 và 3, và phải khóa (B) giữa các bước 4 và 5.

6.8. Câu lệnh wait () trong tất cả các ví dụ chương trình Java là một phần của một thời gian vòng. Giải thích lý do tại sao bạn luôn cần sử dụng câu lệnh while khi sử dụng wait () và tại sao bạn sẽ không bao giờ sử dụng câu lệnh if.

Trả lời:

Đây là một vấn đề quan trọng cần nhấn mạnh! Java chỉ cung cấp thông báo ẩn danh — bạn không thể thông báo cho một chuỗi nhất định rằng một chứng chỉ điều kiện tain là đúng. Khi một luồng được thông báo, nó có trách nhiệm để kiểm tra lại điều kiện mà nó đang chờ đợi. Nếu một chuỗi không kiểm tra lại điều kiện, nó có thể đã nhận được thông báo mà không có điều kiện đã được đáp ứng

Chương 7

7.1. Liệt kê ba ví dụ về deadlock không liên quan đến môi trường hệ thống máy tính.

Trả lời:

• Hai xe ô tô qua cầu một làn ngược chiều.
• Một người đi xuống thang trong khi người khác leo lên thang.
• Hai đoàn tàu chạy về phía nhau trên cùng một đường ray.
• Hai người thợ mộc phải giã móng tay. Có một cái búa duy nhất và một xô đinh. Bế tắc xảy ra nếu một người thợ mộc có cái búa và người thợ mộc kia đóng đinh.

7.2. Giả sử rằng một hệ thống ở trạng thái không an toàn. Chứng tỏ rằng nó có thể các quy trình để hoàn thành việc thực thi của chúng mà không gặp phải bế tắc tình trạng.

Trả lời: Trạng thái không an toàn có thể không nhất thiết dẫn đến bế tắc, nó chỉ có nghĩa là chúng tôi không thể đảm bảo rằng deadlock sẽ không xảy ra. Do đó nó có thể một hệ thống ở trạng thái không an toàn vẫn có thể cho phép tất cả các quá trình để hoàn thành mà không xảy ra bế tắc. Xem xét tình huống nơi một hệ thống có 12 tài nguyên được phân bổ giữa các quá trình P0, P1 và P2. Các các nguồn lực được phân bổ theo chính sách sau:

	Nhu cầu	hiện tại	tối đa
P0	10	5	5
P1	4	2	2
P2	9	3	6

```
for (int i = 0; i <n; i ++) {
    // đầu tiên tìm một luồng có thể
    kết thúc for (int j = 0; j <n; j ++) {
        if (! finish [j]) {
            boolean temp = true;
            for (int k = 0; k <m; k ++) {
                if (need [j] [k]> work
                    [k]) temp = false;
            }
            if (temp) { // nếu luồng này có thể kết
                thúc xong [j] = true;
                 for (int x = 0; x <m; x
                 ++) work [x] + = work
                            [j] [x];
            }
        }
    }
}
```

Hình 7.1 An toàn thuật toán của ngân hàng thuật toán.

Hiện tại có hai nguồn có sẵn. Hệ thống này ở trạng thái không an toàn vì quá trình P1 có thể hoàn thành, do đó giải phóng tổng cộng bốn tài nguyên. Nhưng chúng tôi không thể đảm bảo rằng các quá trình P0 và P2 có thể hoàn thành. Tuy nhiên, có thể một quy trình có thể giải phóng tài nguyên trước yêu cầu thêm. Ví dụ: quy trình P2 có thể giải phóng một tài nguyên, do đó nâng tổng số tài nguyên lên năm. Điều này cho phép quá trình P0 hoàn thành, điều này sẽ giải phóng tổng cộng chín tài nguyên, do đó cho phép quá trình P2 cũng hoàn thành.

7.3. Chứng minh rằng thuật toán an toàn được trình bày trong Phần 7.5.3 yêu cầu thứ tự của m × n2 phép toán.

Trả lời:
Hình 7.1 cung cấp mã Java thực thi thuật toán an toàn của thuật toán của chủ ngân hàng (việc triển khai hoàn chỉnh của thuật toán có sẵn với tải xuống mã nguồn). Như có thể thấy, các vòng ngoài lồng nhau — cả hai đều vòng qua n lần — cung cấp hiệu suất n2. Trong các vòng ngoài này là hai tuần tự vòng lặp bên trong mà vòng lặp m lần. Điểm lớn của thuật toán này do đó là O (m × n2).

7.4. Hãy xem xét một hệ thống máy tính chạy 5.000 công việc mỗi tháng mà không kế hoạch phòng ngừa bế tắc hoặc tránh bế tắc. Bế tắc xảy ra khoảng hai lần mỗi tháng và nhà điều hành phải kết thúc và chạy lại khoảng 10 công việc cho mỗi lần bế tắc. Mỗi công việc trị giá khoảng 2 đô la (tính theo thời gian CPU), và Các công việc bị chấm dứt có xu hướng được thực hiện một nửa khi chúng bị hủy bỏ. Một lập trình viên hệ thống đã ước tính rằng tránh được bế tắc thuật toán (như thuật toán của chủ ngân hàng) có thể được cài đặt trong hệ thống với sự gia tăng thời gian thực hiện trung bình cho mỗi công việc khoảng 10 phần trăm. Vì máy hiện có 30 phần trăm thời gian nhàn rỗi, tất cả 5.000 công việc trên mỗi vẫn có thể chạy trong tháng, mặc dù thời gian quay vòng sẽ tăng lên trung bình khoảng 20 phần trăm.
a.Các đối số để cài đặt tránh bế tắc là gì thuật toán?
b. Các lập luận chống lại việc cài đặt tránh bế tắc là gì thuật toán?
Trả lời:
Một đối số để cài đặt tính năng tránh deadlock trong hệ thống là chúng tôi có thể đảm bảo không bao giờ xảy ra bế tắc. Ngoài ra, mặc dù tăng thời gian quay vòng, tất cả 5.000 công việc vẫn có thể chạy. Một lập luận chống lại việc cài đặt phần mềm tránh bế tắc là bế tắc xảy ra không thường xuyên và chúng tốn rất ít chi phí khi chúng xảy ra.

7.5. Hệ thống có thể phát hiện ra rằng một số quy trình của nó đang chết đói không? Nếu bạn trả lời "Có", giải thích làm thế nào nó có thể. Nếu bạn trả lời "không", hãy giải thích cách hệ thống có thể đối phó với vấn đề chết đói.
Trả lời:
Đói là một chủ đề khó xác định vì nó có thể có nghĩa khác những thứ cho các hệ thống khác nhau. Vì mục đích của câu hỏi này, chúng tôi sẽ định nghĩa chết đói là tình huống mà theo đó một quá trình phải chờ đợi một khoảng thời gian hợp lý — có thể là vô thời hạn — trước khi nhận được một tài nguyên được yêu cầu. Một cách để phát hiện nạn đói trước tiên là xác định một khoảng thời gian — T —được coi là không hợp lý. Khi một quá trình yêu cầu một tài nguyên, một bộ đếm thời gian được bắt đầu. Nếu thời gian trôi qua vượt quá T, thì quá trình này coi như bị bỏ đói. Một chiến lược để đối phó với nạn đói là áp dụng chính sách nơi tài nguyên chỉ được chỉ định cho quá trình đã được chờ đợi dài nhất. Ví dụ: nếu quá trình Pa

đã đợi tài nguyên X lâu hơn quá trình Pb, thì yêu cầu từ quá trình Pb sẽ bị hoãn lại cho đến khi yêu cầu của Pa được đáp ứng. Một chiến lược khác sẽ ít nghiêm ngặt hơn những gì vừa được đề cập. Ở trong trong trường hợp này, một tài nguyên có thể được cấp cho một quy trình đã chờ đợi ít hơn hơn một quy trình khác, với điều kiện là quy trình khác không bị chết đói. Tuy nhiên, nếu một quá trình khác được coi là đang chết đói, yêu cầu của nó sẽ được hài lòng đầu tiên.

7.6. Xem xét chính sách phân bổ nguồn lực sau đây. Yêu cầu và phát hành cho các tài nguyên được cho phép bất cứ lúc nào. Nếu một yêu cầu tài nguyên không thể hài lòng vì tài nguyên không có sẵn, sau đó chúng tôi kiểm tra bất kỳ các quy trình bị chặn, đang chờ tài nguyên. Nếu họ có mong muốn tài nguyên, sau đó những tài nguyên này bị lấy đi và được cung cấp cho quá trình yêu cầu. Vectơ tài nguyên cho quá trình đang chờ đợi được tăng lên để bao gồm các tài nguyên đã bị lấy đi.
Ví dụ: hãy xem xét một hệ thống có ba loại tài nguyên và vectơ có sẵn được khởi tạo thành (4,2,2). Nếu quá trình P0 yêu cầu (2,2,1), nó sẽ nhận được họ. Nếu P1 yêu cầu (1,0,1), nó sẽ nhận được chúng. Sau đó, nếu P0 yêu cầu (0,0,1), nó bị chặn (tài nguyên không có sẵn). Nếu P2 bây giờ yêu cầu (2,0,0), nó sẽ nhận được có sẵn một (1,0,0) và một đã được cấp phát cho P0 (vì P0 bị chặn).
Vectơ Phân bổ của P0 đi xuống (1,2,1) và vectơ Cần của nó đi lên (1,0,1).
a. Có thể xảy ra bế tắc không? Nếu bạn trả lời "có", hãy đưa ra một ví dụ. nếu bạn trả lời “không”, chỉ rõ điều kiện cần thiết nào không thể xảy ra.
b. Chặn vô thời hạn có thể xảy ra không? Giải thich câu trả lơi của bạn.
Trả lời:
a. Bế tắc không thể xảy ra vì quyền ưu tiên tồn tại.
b. Vâng. Một quá trình có thể không bao giờ có được tất cả các tài nguyên mà nó cần nếu chúng liên tục được ưu tiên bởi một loạt các yêu cầu như của quá trình C.

7.7. Giả sử rằng bạn đã mã hóa thuật toán an toàn tránh bế tắc và bây giờ đã được yêu cầu triển khai thuật toán phát hiện bế tắc. Bạn có thể làm như vậy chỉ bằng cách sử dụng mã thuật toán an toàn và xác định lại Maxi = Waiti + Allocationi, trong đó Waiti là một vectơ xác định quy trình tài nguyên mà tôi đang chờ đợi và Allocationi là được định nghĩa trong Phần 7.5? Giải thich câu trả lơi của bạn.
Trả lời:
Vâng. Vectơ tối đa thể hiện yêu cầu tối đa mà một quy trình có thể làm. Khi tính toán thuật toán an toàn, chúng tôi sử dụng ma trận Cần, đại diện cho Max - Phân bổ. Một cách khác để nghĩ về điều này là Max = Cần + Phân bổ. Theo câu hỏi, ma trận Chờ đợi đáp ứng một vai trò tương tự như ma trận Cần, do đó Max = Chờ đợi + Phân bổ.

7.8. Có khả năng xảy ra bế tắc chỉ liên quan đến một quy trình duy nhất không? Giải thich câu trả lơi của bạn.
Trả lời: Không. Điều này xảy ra trực tiếp từ điều kiện giữ và chờ

Chương 8. SOL

8.1. Đặt tên cho hai điểm khác biệt giữa địa chỉ logic và địa chỉ vật lý.
Trả lời: Một địa chỉ logic không tham chiếu đến một địa chỉ thực hiện có; đúng hơn, nó đề cập đến một địa chỉ trừu tượng trong một không gian địa chỉ trừu tượng. Đối chiếu điều này với một địa chỉ thực đề cập đến một địa chỉ thực thực tế trong trí nhớ. Một địa chỉ logic được tạo ra bởi CPU và được dịch thành một địa chỉ vật lý bởi đơn vị quản lý bộ nhớ (MMU). Do đó, địa chỉ vật lý được tạo bởi MMU.

8.2. Hãy xem xét một hệ thống trong đó một chương trình có thể được tách thành hai phần: mã và dữ liệu. CPU biết liệu nó muốn một lệnh (tìm nạp lệnh) hay dữ liệu (tìm nạp hoặc lưu trữ dữ liệu). Do đó, hai giới hạn cơ số Các cặp thanh ghi được cung cấp: một cho hướng dẫn và một cho dữ liệu. Các Cặp thanh ghi giới hạn-cơ sở hướng dẫn tự động ở chế độ chỉ đọc, vì vậy các chương trình có thể được chia sẻ giữa những người dùng khác nhau. Thảo luận về những ưu điểm và nhược điểm của chương trình này.
Trả lời:
Ưu điểm chính của chương trình này là nó là một cơ chế chia sẻ mã và dữ liệu. Ví dụ: chỉ một bản sao của một trình soạn thảo hoặc trình biên dịch cần được lưu trong bộ nhớ và mã này có thể được chia sẻ bởi tất cả các quy trình cần quyền truy cập vào trình soạn thảo hoặc mã trình biên dịch. Một ưu điểm khác là bảo vệ mã khỏi sửa đổi sai. Điểm bất lợi duy nhất là mã và dữ liệu phải được tách biệt, thường được gắn vào mã do trình biên dịch tạo ra.

8.3. Tại sao kích thước trang luôn là lũy thừa của 2?
Trả lời: Nhớ lại rằng phân trang được thực hiện bằng cách chia nhỏ một địa chỉ thành một trang và số bù đắp. Cách hiệu quả nhất là ngắt địa chỉ thành các bit trang X và bit bù Y, thay vì thực hiện số học trên địa chỉ để tính số trang và bù đắp. Vì mỗi chút vị trí đại diện cho một lũy thừa của 2, tách một địa chỉ giữa các bit kết quả ở kích thước trang là lũy thừa của 2.

8.4. Hãy xem xét một không gian địa chỉ logic gồm tám trang, mỗi trang 1024 từ, ánh xạ vào bộ nhớ vật lý gồm 32 khung hình.
a.Có bao nhiêu bit trong địa chỉ logic?
b. Có bao nhiêu bit trong địa chỉ vật lý?

Trả lời:
Một. Địa chỉ logic: 13 bit
b. Địa chỉ vật lý: 15 bit

8.5. Tác dụng của việc cho phép hai mục nhập trong bảng trang trỏ đến cùng một khung trang trong bộ nhớ? Giải thích cách hiệu ứng này có thể được sử dụng để giảm lượng thời gian cần thiết để sao chép một lượng lớn bộ nhớ từ nơi này đến nơi khác. Việc cập nhật một số byte sẽ ảnh hưởng gì đến một trang có trên trang kia?
Trả lời:
Bằng cách cho phép hai mục nhập trong một bảng trang trỏ đến cùng một khung trang trong bộ nhớ, người dùng có thể chia sẻ mã và dữ liệu. Nếu mã được nhập lại, nhiều không gian bộ nhớ có thể được tiết kiệm thông qua việc sử dụng chung các chương trình như trình soạn thảo văn bản, trình biên dịch và hệ thống cơ sở dữ liệu. Việc "sao chép" một lượng lớn bộ nhớ có thể được thực hiện bằng cách có bảng trang trỏ đến cùng một vị trí bộ nhớ. Tuy nhiên, việc chia sẻ mã hoặc dữ liệu không quan tâm có nghĩa là bất kỳ người dùng nào có quyền truy cập vào mã có thể sửa đổi nó và những sửa đổi này sẽ được phản ánh trong "bản sao" của người dùng khác.

8.6. Mô tả cơ chế mà một phân đoạn có thể thuộc về địa chỉ không gian của hai quá trình khác nhau.
Trả lời: Vì bảng phân đoạn là tập hợp các thanh ghi giới hạn cơ sở, các phân đoạn có thể được chia sẻ khi các mục nhập trong bảng phân đoạn của hai các công việc trỏ đến cùng một vị trí thực tế. Hai bảng phân đoạn phải có con trỏ cơ sở giống hệt nhau và số phân đoạn được chia sẻ phải là giống nhau trong hai quá trình.

8.7. Chia sẻ phân đoạn giữa các quá trình mà không yêu cầu cùng một phân đoạn có thể có trong một hệ thống phân đoạn được liên kết động.
a. Xác định một hệ thống cho phép liên kết tĩnh và chia sẻ các phân đoạn mà không yêu cầu số phân đoạn phải giống nhau.
b. Mô tả một lược đồ phân trang cho phép các trang được chia sẻ mà không cần yêu cầu số trang phải giống nhau.
Trả lời:
Cả hai vấn đề này đều giảm xuống mức chương trình có thể tham chiếu cả mã riêng và dữ liệu của nó mà không cần biết phân đoạn hoặc số trang liên kết với địa chỉ. MULTICS đã giải quyết vấn đề này bằng cách liên kết bốn thanh ghi với mỗi quá trình. Một sổ đăng ký có địa chỉ của phân đoạn chương trình hiện tại, một phân đoạn khác có địa chỉ cơ sở cho ngăn xếp, ngăn xếp khác có địa chỉ cơ sở cho dữ liệu chung, v.v. Các ý tưởng là tất cả các tham chiếu phải gián tiếp thông qua một

sổ đăng ký ánh xạ đến phân đoạn hoặc số trang hiện tại. Bằng cách thay đổi các thanh ghi này, cùng một mã có thể thực thi cho các quy trình khác nhau mà không có cùng một trang hoặc số phân đoạn.

8.8. Trong IBM / 370, bảo vệ bộ nhớ được cung cấp thông qua việc sử dụng các phím.
Khóa là một số lượng 4 bit. Mỗi khối bộ nhớ 2K có một khóa (bộ nhớ key) được liên kết với nó. CPU cũng có một chìa khóa (chìa khóa bảo vệ) Liên kết với nó. Hoạt động cửa hàng chỉ được phép nếu cả hai khóa đều bằng nhau, hoặc nếu một trong hai bằng 0. Quản lý bộ nhớ nào sau đây chương trình có thể được sử dụng thành công với phần cứng này?
a. Máy trần
b. Hệ thống một người dùng
C. Đa chương trình với một số quy trình cố định
d. Đa chương trình với một số quy trình thay đổi
e. Phân trang
f. Phân đoạn

Trả lời:
a.Bảo vệ không cần thiết, đặt khóa hệ thống thành 0.
b. Đặt khóa hệ thống thành 0 khi ở chế độ giám sát.
C. Kích thước vùng phải được cố định theo gia số 2k byte, cấp phát khóa với các khối bộ nhớ.
d. Giống như trên.
e. Kích thước khung hình phải có gia số là 2k byte, phân bổ khóa bằng các trang.
f. Kích thước phân đoạn phải có gia số là 2k byte, cấp phát khóa bằng các phân đoạn.

Chương 9

9.1. Lỗi trang xảy ra trong những trường hợp nào? Mô tả các hành động được thực hiện bởi hệ điều hành khi xảy ra lỗi trang.

Trả lời:
Lỗi trang xảy ra khi truy cập vào một trang không được đưa vào bộ nhớ chính diễn ra. Hệ điều hành xác minh truy cập bộ nhớ, hủy bỏ chương trình nếu nó không hợp lệ. Nếu nó hợp lệ, a khung miễn phí được định vị và I / O được yêu cầu để đọc trang cần thiết vào khung miễn phí. Sau khi hoàn thành I / O, bảng quy trình và bảng trang được cập nhật và hướng dẫn được khởi động lại.

9.2. Giả sử rằng bạn có một chuỗi tham chiếu trang cho một quy trình với m khung (ban đầu tất cả đều trống). Chuỗi tham chiếu trang có độ dài p; n số trang riêng biệt xuất hiện trong đó. Trả lời các câu hỏi này cho bất kỳ thuật toán thay thế trang nào:
a. Giới hạn dưới về số lỗi trang là gì?
b. Giới hạn trên về số lỗi trang là gì?

Trả lời: a. n
b. P

9.3. Kỹ thuật và cấu trúc lập trình nào sau đây là
"Tốt" cho môi trường phân trang theo yêu cầu? Cái nào là "không tốt"? Giải thích câu trả lời của bạn.
a. Cây rơm
b. Bảng ký hiệu băm
c. Tìm kiếm tuần tự
d. Tìm kiếm nhị phân
e. Mã thuần túy
f. Phép toán vectơ
g. Chuyển hướng

Trả lời:
a. Ngăn xếp — tốt.
b. Bảng ký hiệu băm — không tốt.
C. Tìm kiếm tuần tự — tốt.
d. Tìm kiếm nhị phân — không tốt.
e. Mã thuần túy — tốt.
f. Phép toán vectơ — tốt.
g. Chỉ đạo — không tốt.

9.4. Xem xét các thuật toán thay thế trang sau đây. Xếp hạng các thuật toán này theo thang điểm năm từ "xấu" đến "hoàn hảo" theo tỷ lệ lỗi trang của họ. Tách các thuật toán bị Belady’s bất thường từ những người không.
Một. Thay thế LRU
b. FIFO thay thế
C. Thay thế tối ưu
d. Cơ hội thứ hai thay thế

Trả lời:
Thuật toán xếp hạng gặp phải sự bất thường của Belady
1 Tối ưu không
2 LRU không
3 Cơ hội thứ hai có
4 FIFO có

9.5. Khi bộ nhớ ảo được triển khai trong một hệ thống máy tính, có chi phí nhất định liên quan đến kỹ thuật và lợi ích nhất định. Liệt kê chi phí và lợi ích. Có khả năng chi phí vượt quá lợi ích không?
Nếu có, những biện pháp nào có thể được thực hiện để đảm bảo rằng điều này không xảy ra?

Trả lời:

Chi phí là phần cứng bổ sung và thời gian truy cập chậm hơn. Các lợi ích là sử dụng tốt bộ nhớ và không gian địa chỉ logic lớn hơn hơn không gian địa chỉ vật lý.

9.6. Hệ điều hành hỗ trợ bộ nhớ ảo được phân trang, sử dụng trung tâm bộ xử lý với thời gian chu kỳ là 1 micro giây. Nó tốn thêm 1 micro giây để truy cập một trang khác với trang hiện tại. Các trang có 1000 từ, và thiết bị phân trang là một cái trống quay với tốc độ 3000 vòng mỗi phút và chuyển 1 triệu từ mỗi giây. Sau các phép đo thống kê thu được từ hệ thống:

• 1 phần trăm của tất cả các hướng dẫn được thực thi đã truy cập vào một trang không phải là trang hiện tại.

• Trong số các hướng dẫn đã truy cập vào một trang khác, 80 phần trăm đã truy cập vào một trang đã có trong bộ nhớ.

• Khi cần trang mới, trang thay thế đã được sửa đổi 50 phần trăm thời gian.

Tính thời gian hướng dẫn hiệu quả trên hệ thống này, giả sử rằng hệ thống chỉ đang chạy một quy trình và bộ xử lý không hoạt động trong trống chuyển.

Trả lời:

thời gian truy cập hiệu quả = 0,99 × (1 giây + 0,008 × (2 giây)
+ 0,002 × (10.000 giây + 1.000 giây)
+ 0,001 × (10.000 giây + 1.000 giây)
= (0,99 + 0,016 + 22,0 + 11,0) giây
= 34.0 giây

9.7. Xét mảng hai chiều A:
int A [] [] = new int [100] [100];
trong đó A [0] [0] ở vị trí 200, trong hệ thống bộ nhớ được phân trang với các trang có kích thước 200. Một quy trình nhỏ ở trang 0 (vị trí từ 0 đến 199) để thao tác với ma trận; do đó, mọi lần tìm nạp hướng dẫn sẽ từ trang 0.

Đối với ba khung trang, có bao nhiêu lỗi trang được tạo ra bởi các vòng lặp khởi tạo mảng sau, sử dụng thay thế LRU và giả sử khung trang 1 có quá trình trong đó, và hai khung còn lại ban đầu trống?

a. for (int j = 0; j <100; j ++)
for (int i = 0; i <100; i ++)
A [i] [j] = 0;

b. for (int i = 0; i <100; i ++)
for (int j = 0; j <100; j ++)
A [i] [j] = 0;

Trả lời: a. 50
b. 5.000

9.8. Xem xét chuỗi tham chiếu trang sau:
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6.
Có bao nhiêu lỗi trang xảy ra đối với các thuật toán thay thế sau đây, giả sử một, hai, ba, bốn, năm, sáu hoặc bảy khung?
Hãy nhớ rằng ban đầu tất cả các khung đều trống, vì vậy các trang duy nhất đầu tiên của bạn sẽ tất cả chi phí một lỗi mỗi.

- Thay thế LRU
- FIFO thay thế
- Thay thế tối ưu

Trả lời:

Số khung hình	LRU	FIFO	Tối ưu
1	20	20	20
2	18	18	15
3	15	16	11
4	10	14	8
5	8	10	7
6	7	10	7
7	7	7	7

9.9. Giả sử rằng bạn muốn sử dụng thuật toán phân trang yêu cầu một bit tham chiếu (chẳng hạn như thay thế cơ hội thứ hai hoặc mô hình tập hợp làm việc), nhưng phần cứng không cung cấp một. Phác thảo cách bạn có thể mô phỏng một bit tham chiếu ngay cả khi một bit không được cung cấp bởi phần cứng hoặc giải thích tại sao nó không thể làm như vậy. Nếu có thể, hãy tính toán chi phí sẽ được.

Trả lời: Bạn có thể sử dụng bit hợp lệ / không hợp lệ được hỗ trợ trong phần cứng để mô phỏng bit tham chiếu. Ban đầu đặt bit thành không hợp lệ. Lần tham khảo

đầu tiên một cái bẫy đối với hệ điều hành được tạo ra. Hệ điều hành sẽ đặt một bit phần mềm thành 1 và đặt lại bit hợp lệ / không hợp lệ thành hợp lệ.

9.10. Bạn đã nghĩ ra một thuật toán thay thế trang mới mà bạn nghĩ có thể được tối ưu. Trong một số trường hợp thử nghiệm méo mó, sự bất thường của Belady xảy ra. Là thuật toán mới tối ưu? Giải thich câu trả lơi của bạn.

Trả lời: Không. Một thuật toán tối ưu sẽ không bị Belady's bất thường bởi vì — theo định nghĩa — một thuật toán tối ưu thay thế trang sẽ không được sử dụng trong thời gian dài nhất. Sự bất thường của Belady xảy ra khi thuật toán thay thế trang loại bỏ một trang sẽ cần trước mắt. Một thuật toán tối ưu sẽ không được chọn một trang như vậy.

9.11. Phân đoạn tương tự như phân trang nhưng sử dụng "các trang" có kích thước thay đổi. Xác định hai thuật toán thay thế phân đoạn dựa trên các lược đồ thay thế trang FIFO và LRU. Hãy nhớ rằng vì các phân đoạn không giống nhau kích thước, phân đoạn được chọn để thay thế có thể không đủ lớn để để lại đủ vị trí liên tiếp cho phân đoạn cần thiết. Xem xét chiến lược cho các hệ thống không thể di dời các phân đoạn và những cho các hệ thống mà họ có thể.

Trả lời:
a. FIFO. Tìm phân đoạn đầu tiên đủ lớn để chứa phân đoạn đến. Nếu không thể di dời và không có một đoạn
đủ lớn, hãy chọn một tổ hợp các phân đoạn có ký ức liền kề, "gần nhất với phần đầu tiên trong danh sách"
và có thể phù hợp với phân khúc mới. Nếu di dời là
có thể, sắp xếp lại bộ nhớ để N phân đoạn đầu tiên lớn
đủ cho phân đoạn đến liền kề trong bộ nhớ. Thêm vào
mọi dung lượng còn lại trong danh sách dung lượng trống trong cả hai trường hợp.
b. LRU. Chọn phân đoạn lâu nhất không được sử dụng
khoảng thời gian và đủ lớn, thêm bất kỳ không gian còn lại nào
vào danh sách dung lượng trống. Nếu không có một phân đoạn nào đủ lớn, hãy chọn
sự kết hợp của các phân đoạn "lâu đời nhất" tiếp giáp trong
bộ nhớ (nếu không thể di chuyển lại) và đủ lớn.
Nếu có khả năng di dời, hãy sắp xếp lại N phân đoạn cũ nhất để được
liền kề trong bộ nhớ và thay thế chúng bằng phân đoạn mới.

9.12. Hãy xem xét một hệ thống máy tính phân trang theo yêu cầu mà mức độ đa chương trình hiện đang được cố định ở mức bốn. Hệ thống gần đây đã được đo để

xác định việc sử dụng CPU và đĩa phân trang. Kết quả là một trong những lựa chọn thay thế sau đây. Đối với mỗi trường hợp, điều gì đang xảy ra?
Có thể tăng mức độ đa chương trình để tăng CPU không
sử dụng? Phân trang có hữu ích không?
Một. Sử dụng CPU 13 phần trăm; sử dụng đĩa 97 phần trăm
b. Sử dụng CPU 87 phần trăm; sử dụng đĩa 3 phần trăm
C. Sử dụng CPU 13 phần trăm; sử dụng đĩa 3 phần trăm

Trả lời:
a. Đang xảy ra va chạm.
b. Việc sử dụng CPU đủ cao để mọi thứ được yên và tăng mức độ đa chương trình.
C. Tăng mức độ đa chương trình

9.13. Chúng tôi có hệ điều hành cho máy sử dụng cơ sở và giới hạn đăng ký, nhưng chúng tôi đã sửa đổi máy để cung cấp một bảng trang. Các bảng trang có thể được thiết lập để mô phỏng các thanh ghi cơ sở và giới hạn không? Làm sao họ có thể được, hoặc tại sao họ có thể không?

Trả lời: Bảng trang có thể được thiết lập để mô phỏng các thanh ghi cơ sở và giới hạn với điều kiện là bộ nhớ được cấp phát trong các phân đoạn có kích thước cố định. Trong này theo cách này, cơ sở của một phân đoạn có thể được nhập vào bảng trang và bit hợp lệ / không hợp lệ được sử dụng để chỉ ra rằng phần của phân đoạn là cư dân trong ký ức. Sẽ có một số vấn đề với sự phân mảnh nội bộ.

Chương 10

10.1. Một số hệ thống tự động xóa tất cả các tệp người dùng khi người dùng đăng xuất hoặc một công việc chấm dứt, trừ khi người dùng yêu cầu rõ ràng rằng chúng được giữ lại; các hệ thống khác giữ tất cả các tệp trừ khi người dùng xóa chúng một cách rõ ràng. Thảo luận về giá trị tương đối của mỗi cách tiếp cận.

Trả lời: Xóa tất cả các tệp không được người dùng lưu cụ thể có lợi thế của việc giảm thiểu dung lượng tệp cần thiết cho mỗi người dùng bằng cách không lưu các tệp không mong muốn hoặc không cần thiết. Lưu tất cả các tệp trừ khi cụ thể đã xóa an toàn hơn cho người dùng vì không thể mất tệp vô tình do quên lưu chúng.

10.2
Tại sao một số hệ thống theo dõi loại tệp, trong khi những hệ thống khác lại nó cho

người dùng hay đơn giản là không triển khai nhiều loại tệp? Cái mà hệ thống "tốt hơn?

Trả lời: Một số hệ thống cho phép các hoạt động tệp khác nhau dựa trên loại tệp (ví dụ: tệp ascii có thể được đọc dưới dạng một luồng trong khi tệp cơ sở dữ liệu có thể được đọc thông qua một chỉ mục cho một khối). Các hệ thống khác rời đi việc giải thích dữ liệu của tệp như vậy đối với quá trình và không cung cấp trợ giúp trong truy cập dữ liệu. Phương pháp "tốt hơn" tùy thuộc vào nhu cầu của các quy trình trên hệ thống và các yêu cầu mà người dùng đặt ra đối với hệ điều hành. Nếu một hệ thống chạy hầu hết các ứng dụng cơ sở dữ liệu, nó có thể hiệu quả hơn để hệ điều hành triển khai tệp loại dữ liệu và cung cấp các hoạt động, thay vì tạo từng chương trình thực hiện cùng một điều (có thể theo nhiều cách khác nhau). Đối với các hệ thống đa năng, có thể tốt hơn nếu chỉ triển khai các loại tệp cơ bản để giữ kích thước hệ điều hành nhỏ hơn và cho phép tự do tối đa đến các quy trình trên hệ thống

10.3. Tương tự, một số hệ thống hỗ trợ nhiều loại cấu trúc cho một tệp dữ liệu, trong khi những người khác chỉ hỗ trợ một luồng byte. Là những gì ưu điểm và nhược điểm?

Trả lời:

Một lợi thế của việc hệ thống hỗ trợ các tệp khác nhau cấu trúc là sự hỗ trợ đến từ hệ thống; các ứng dụng riêng lẻ không bắt buộc phải cung cấp hỗ trợ. Ngoài ra, nếu hệ thống cung cấp hỗ trợ cho các cấu trúc tệp khác nhau, nó có thể thực hiện hỗ trợ có lẽ hiệu quả hơn một ứng dụng. Điểm bất lợi của việc hệ thống cung cấp hỗ trợ cho tệp đã xác định loại là nó làm tăng kích thước của hệ thống. Ngoài ra, các ứng dụng có thể yêu cầu các loại tệp khác với những gì được cung cấp bởi hệ thống có thể không chạy trên các hệ thống như vậy. Một chiến lược thay thế là để hệ điều hành xác định không hỗ trợ cho cấu trúc tệp và thay vào đó coi tất cả các tệp là một chuỗi byte. Đây là cách tiếp cận được thực hiện bởi hệ thống UNIX. Ưu điểm của phương pháp này là nó đơn giản hóa việc hỗ trợ hệ điều hành cho các hệ thống tệp, vì hệ thống không còn phải cung cấp cấu trúc cho các loại tệp khác nhau. Hơn nữa, nó cho phép các ứng dụng xác định cấu trúc tệp, do đó giảm bớt tình huống hệ thống có thể không cung cấp định nghĩa tệp cần thiết cho một ứng dụng cụ thể.

10.4. Bạn có thể mô phỏng cấu trúc thư mục đa cấp với một cấp đơn không cấu trúc thư mục trong đó các tên dài tùy ý có thể được sử dụng? Nếu là của bạn câu trả lời là có, giải thích cách bạn có thể làm như vậy và đối chiếu sơ đồ này với lược đồ thư mục đa cấp. Nếu câu trả lời của bạn là không, hãy giải thích điều gì ngăn cản sự thành công của mô phỏng của bạn. Câu trả lời của bạn sẽ thay đổi như thế nào nếu tên tệp được giới hạn trong bảy ký tự?

Trả lời: Nếu các tên dài tùy ý có thể được sử dụng thì có thể mô phỏng một cấu trúc thư mục đa cấp. Điều này có thể được thực hiện, ví dụ, bằng cách sử dụng ký tự "." để chỉ ra phần cuối của một thư mục con. Vì vậy cho ví dụ, tên jim.java.F1 chỉ định rằng F1 là một tệp trong thư mục con java lần lượt nằm trong thư mục gốc jim. Nếu tên tệp được giới hạn trong bảy ký tự, thì lược đồ trên không thể được sử dụng và do đó, nói chung, câu trả lời là không. Điều tốt nhất tiếp theo cách tiếp cận trong tình huống này sẽ là sử dụng một tệp cụ thể làm biểu tượng bảng (thư mục) để ánh xạ các tên dài tùy ý (chẳng hạn như jim.java.F1) thành các tên tùy ý ngắn hơn (chẳng hạn như XX00743), sau đó được sử dụng để truy cập tệp thực tế.

10.5. Giải thích mục đích của các hoạt động open () và close ()

Trả lời:
• Thao tác open () thông báo cho hệ thống rằng tệp được đặt tên sắp hoạt động.
• Thao tác close () thông báo cho hệ thống rằng tệp được đặt tên không còn được sử dụng tích cực bởi người dùng đã ban hành thao tác đóng.

10.6. Đưa ra một ví dụ về một ứng dụng trong đó dữ liệu trong tệp sẽ được truy cập theo thứ tự sau:
a. Bài tập Thực hành Tuần tự
b. Ngẫu nhiên

Trả lời:
a. In nội dung của tệp.
b. In nội dung của bản ghi i. Bản ghi này có thể được tìm thấy bằng cách sử dụng kỹ thuật băm hoặc chỉ mục.

10.7. Trong một số hệ thống, một thư mục con có thể được đọc và ghi bởi người dùng được ủy quyền, giống như các tệp thông thường.
a. Mô tả các vấn đề bảo vệ có thể phát sinh.
b. Đề xuất phương án giải quyết từng vấn đề bảo vệ mà bạn đã nêu trong phần a.

Trả lời:
a. Một phần thông tin được lưu giữ trong mục nhập thư mục là vị trí tệp. Nếu người dùng có thể sửa đổi vị trí này, thì anh ta có thể truy cập các tệp khác đánh bại kế hoạch bảo vệ quyền truy cập.
b. Không cho phép người dùng ghi trực tiếp vào thư mục con. Thay vào đó, cung cấp các hoạt động hệ thống để làm như vậy.

10.8. Xem xét một hệ thống hỗ trợ 5000 người dùng. Giả sử rằng bạn muốn cho phép 4990 người dùng này có thể truy cập vào một tệp.
a. Bạn sẽ chỉ định sơ đồ bảo vệ này như thế nào trong UNIX?
b. Bạn có thể đề xuất một sơ đồ bảo vệ khác có thể được sử dụng hiệu quả hơn cho mục đích này so với sơ đồ do UNIX cung cấp không?

Trả lời:
a. Có hai phương pháp để đạt được điều này:
1. Tạo danh sách kiểm soát truy cập với tên của tất cả 4990 người dùng. 2. Đặt 4990 người dùng này vào một nhóm và đặt quyền truy cập nhóm cho phù hợp. Không phải lúc nào cũng có thể triển khai lược đồ này vì các nhóm người dùng bị hệ thống hạn chế.
b. Quyền truy cập chung vào tệp áp dụng cho tất cả người dùng trừ khi tên của họ xuất hiện trong danh sách kiểm soát truy cập với quyền truy cập khác. Với lược đồ này, bạn chỉ cần đưa tên của mười người dùng còn lại vào danh sách kiểm soát truy cập nhưng không có đặc quyền truy cập nào được phép.

10.9. Các nhà nghiên cứu đã gợi ý rằng, thay vì có danh sách truy cập được liên kết với từng tệp (chỉ định người dùng nào có thể truy cập tệp và cách thức), chúng ta nên có danh sách kiểm soát người dùng được liên kết với từng người dùng (chỉ định tệp người dùng có thể truy cập và Làm sao). Thảo luận về giá trị tương đối của hai chương trình này.

Trả lời: Giao diện hệ thống tệp
• Danh sách kiểm soát tập tin. Vì thông tin kiểm soát truy cập được tập trung ở một nơi duy nhất, nên việc thay đổi thông tin kiểm soát truy cập sẽ dễ dàng hơn và điều này đòi hỏi ít không gian hơn.
• Danh sách kiểm soát người dùng. Điều này yêu cầu ít chi phí hơn khi mở một tệp.

Chương 11

11.1. Hãy xem xét một tệp hiện có 100 khối. Giả sử rằng
khối điều khiển tệp (và khối chỉ mục, trong trường hợp cấp phát được lập chỉ mục) đã có trong bộ nhớ. Tính toán có bao nhiêu thao tác vào / ra đĩa là bắt buộc đối với các chiến lược phân bổ liền kề, được liên kết và được lập chỉ mục (cấp đơn), nếu, đối với một khối, các điều kiện sau đây được giữ nguyên. bên trong trường hợp phân bổ liền kề, giả sử rằng không có chỗ để phát triển trong bắt đầu, nhưng vẫn có chỗ để phát triển cuối cùng. Giả sử rằng khối thông tin cần thêm được lưu trữ trong bộ nhớ.

a. Khối được thêm vào ở đầu.
b. Khối được thêm vào giữa.
C. Khối được thêm vào cuối.
d. Khối được xóa ngay từ đầu.
e. Khối được loại bỏ từ giữa.
f. Khối được loại bỏ từ cuối.

Trả lời:
Được liên kết liền kề được lập chỉ mục
a. 201 1 1
b. 101 52 1
C. 1 3 1
d. 198 1 0
e. 98 52 0
f. 0 100 0

11.2. Những vấn đề nào có thể xảy ra nếu một hệ thống cho phép một hệ thống tệp được gắn đồng thời tại nhiều vị trí?

Trả lời: Sẽ có nhiều đường dẫn đến cùng một tệp, có thể
gây nhầm lẫn cho người dùng hoặc khuyến khích những sai lầm (xóa tệp bằng một đường dẫn
xóa tệp trong tất cả các đường dẫn khác).

11.3. Tại sao bản đồ bit để cấp phát tệp phải được lưu trữ trên bộ nhớ chung, thay vì hơn trong bộ nhớ chính?

Trả lời: Trong trường hợp hệ thống gặp sự cố (lỗi bộ nhớ), danh sách dung lượng trống sẽ không bị mất nếu bản đồ bit được lưu trữ trong main kỉ niệm.

11.4. Xem xét một hệ thống hỗ trợ các chiến lược tiếp giáp, liên kết, và phân bổ được lập chỉ mục. Tiêu chí nào nên được sử dụng để quyết định chiến lược được sử dụng tốt nhất cho một tệp cụ thể?

Trả lời:
• Liền kề — nếu tệp thường được truy cập tuần tự, nếu tệp
mối quan hệ nhỏ.
• Được liên kết — nếu tệp lớn và thường được truy cập tuần tự.
• Được lập chỉ mục — nếu tệp lớn và thường được truy cập ngẫu nhiên

11.5. Một vấn đề với phân bổ liền kề là người dùng phải phân bổ trước đủ dung lượng cho mỗi tệp. Nếu tệp phát triển lớn hơn không gian được phân bổ cho nó, các hành động đặc biệt phải được thực hiện. Một giải pháp cho vấn đề này là xác định một cấu trúc tệp bao gồm một vùng tiếp giáp ban đầu (có kích thước xác định). Nếu khu vực này được lấp đầy, hoạt động hệ thống tự động xác định vùng tràn được liên kết với vùng tiếp giáp ban đầu. Nếu vùng tràn được lấp đầy, vùng tràn khác được phân bổ. So sánh việc triển khai tệp này với tiêu chuẩn triển khai liền kề và liên kết.

Trả lời: Phương pháp này yêu cầu chi phí cao hơn so với phân bổ liền kề tiêu chuẩn. Nó yêu cầu ít chi phí hơn so với tiêu chuẩn được liên kết sự phân bổ.

11.6. Làm cách nào để bộ nhớ đệm giúp cải thiện hiệu suất? Tại sao hệ thống không sử dụng bộ nhớ đệm nhiều hơn hoặc lớn hơn nếu chúng rất hữu ích?

Trả lời: Bộ nhớ đệm cho phép các thành phần có tốc độ khác nhau giao tiếp hiệu quả hơn bằng cách lưu trữ dữ liệu từ thiết bị chậm hơn, tạm thời trong một thiết bị nhanh hơn (bộ nhớ cache). Bộ nhớ đệm, gần như theo định nghĩa, nhiều hơn đắt hơn thiết bị mà họ đang lưu vào bộ nhớ đệm, do đó, việc tăng số lượng hoặc kích thước của bộ nhớ đệm sẽ làm tăng chi phí hệ thống.

11.7. Tại sao việc phân bổ động các bảng bên trong của hệ điều hành lại có lợi cho người dùng? Các hình phạt đối với hoạt động là gì hệ thống để làm như vậy?

Trả lời: Bảng động cho phép linh hoạt hơn trong tăng trưởng sử dụng hệ thống—bảng không bao giờ được vượt quá, tránh giới hạn sử dụng giả tạo. Không may, cấu trúc hạt nhân và mã phức tạp hơn, vì vậy có nhiều tiềm năng cho lỗi. Việc sử dụng một tài nguyên có thể lấy đi nhiều hệ thống hơn tài nguyên (bằng cách phát triển để đáp ứng các yêu cầu) so với tĩnh những cái bàn. Bài tập thực hành 41

11.8. Giải thích cách lớp VFS cho phép hệ điều hành dễ dàng hỗ trợ nhiều loại hệ thống tệp.

Trả lời: VFS giới thiệu một lớp chuyển hướng trong việc triển khai hệ thống tệp. Về nhiều mặt, nó tương tự như lập trình hướng đối tượng kỹ thuật. Các cuộc gọi hệ thống có thể được thực hiện một cách chung chung (độc lập với tệp loại hệ thống). Mỗi loại hệ thống tệp cung cấp các lệnh gọi hàm và dữ liệu của nó cấu trúc của lớp VFS. Một lệnh gọi hệ thống được dịch thành các chức năng cụ thể cho hệ thống tệp đích ở lớp VFS. Cuộc gọi chương trình không có mã dành riêng cho hệ thống

tệp và các cấp trên của cấu trúc cuộc gọi hệ thống cũng tương tự như vậy là độc lập với hệ thống tệp. Bản dịch ở lớp VFS biến các lệnh gọi chung này thành tệp dành riêng cho hệ thống các hoạt động.

Chương 12

12.1. Tìm kiếm tăng tốc được mô tả trong Bài tập 12.3 là điển hình của đĩa cứng ổ đĩa. Ngược lại, đĩa mềm (và nhiều đĩa cứng được sản xuất trước giữa những năm 1980) thường tìm kiếm ở một tỷ lệ cố định. Giả sử rằng đĩa trong Bài tập 12.3 có một tìm kiếm tốc độ không đổi chứ không phải một tìm kiếm gia tốc không đổi, vì vậy thời gian tìm kiếm có dạng t = x + yL, trong đó t là thời gian tính bằng mili giây và L là khoảng cách tìm kiếm. Giả sử rằng thời gian để tìm kiếm một hình trụ liền kề là 1 mili giây, như trước đây và là 0,5 mili giây cho mỗi xi lanh bổ sung.
a. Viết một phương trình cho thời gian tìm kiếm này dưới dạng một hàm của tìm kiếm khoảng cách.
b. Sử dụng hàm thời gian tìm kiếm từ phần a, tính tổng số lần tìm kiếm thời gian cho mỗi lịch trình trong Bài tập 12.2. Là câu trả lời của bạn giống như đối với Bài tập 12.3 (c)?
c. Tốc độ phần trăm của lịch trình nhanh nhất qua FCFS là bao nhiêu trong trường hợp này?

Trả lời:
a. t = 0,95 + 0,05L
b. FCFS 362.60; SSTF 95,80; QUÉT 497,95; NHÌN 174.50; C-QUÉT 500,15; (và C-LOOK 176.70). SSTF vẫn là người chiến thắng và LOOK là á quân.
c. (362,60 - 95,80) /362,60 = 0,74 Phần trăm tốc độ tăng của SSTF trên FCFS là 74%, liên quan đến thời gian tìm kiếm. Nếu chúng tôi bao gồm tổng thời gian chờ quay và truyền dữ liệu, tỷ lệ phần trăm tốc độ sẽ ít hơn.

12.2. Lập lịch đĩa, khác với lập lịch FCFS, hữu ích cho một người dùng Môi trường? Giải thich câu trả lơi của bạn.

Trả lời: Trong môi trường một người dùng, hàng đợi I / O thường trống. Các yêu cầu thường đến từ một quy trình duy nhất cho một khối hoặc cho một dãy các khối liên tiếp. Trong những trường hợp này, FCFS là một phương pháp lập lịch đĩa. Nhưng LOOK gần như dễ lập trình và sẽ mang lại hiệu suất tốt hơn nhiều khi nhiều quy trình thực hiện I / O đồng thời, chẳng hạn như khi trình duyệt Web truy xuất dữ liệu trong nền trong khi hệ điều hành đang phân trang và một ứng dụng đang hoạt động ở nền trước.

12.3. Giải thích lý do tại sao lập kế hoạch SSTF có xu hướng ưu tiên các xi lanh ở giữa hơn hình trụ trong cùng và ngoài cùng.

Trả lời: Tâm của đĩa là vị trí có độ lớn nhỏ nhất khoảng cách trung bình đến tất cả các bản nhạc khác. Do đó đầu đĩa có xu hướng di chuyển ra khỏi các cạnh của đĩa. Đây là một cách khác để nghĩ về nó. Các vị trí hiện tại của đầu chia các xi lanh thành hai nhóm. Nếu đầu không nằm ở trung tâm của đĩa và một yêu cầu mới đến, yêu cầu mới có nhiều khả năng nằm trong nhóm bao gồm trung tâm của đĩa; do đó, đầu có nhiều khả năng di chuyển theo hướng đó.

12.4. Tại sao độ trễ quay thường không được xem xét trong lập lịch đĩa? Bạn sẽ sửa đổi SSTF, SCAN và C-SCAN như thế nào để bao gồm độ trễ tối ưu hóa?

Trả lời: Hầu hết các đĩa không xuất thông tin vị trí quay của chúng cho máy chủ. Ngay cả khi họ đã làm vậy, thời gian để thông tin này đến người lập lịch sẽ phải không chính xác và thời gian tiêu tốn bởi bộ lập lịch có thể thay đổi, vì vậy thông tin vị trí xoay sẽ trở nên không chính xác. Hơn nữa, các yêu cầu đĩa thường được đưa ra trong các điều kiện số khối logic và ánh xạ giữa các khối logic và địa điểm vật lý là rất phức tạp.

12.5. Việc sử dụng đĩa RAM sẽ ảnh hưởng như thế nào đến việc lựa chọn lập lịch đĩa của bạn thuật toán? Bạn cần xem xét những yếu tố nào? Làm điều tương tự các cân nhắc áp dụng cho việc lập lịch đĩa cứng, vì hệ thống tệp lưu trữ các khối được sử dụng gần đây trong một bộ đệm đệm trong bộ nhớ chính?

Trả lời: Việc lên lịch đĩa cố gắng giảm thiểu thời gian trên định vị đầu đĩa. Vì đĩa RAM có thời gian truy cập đồng nhất, lập kế hoạch phần lớn là không cần thiết. Sự so sánh giữa đĩa RAM và bộ nhớ đệm đĩa nhớ chính không có liên quan gì đến đĩa cứng lập lịch bởi vì chúng tôi chỉ lập lịch bộ nhớ đệm bộ đệm bị bỏ lỡ, không phải yêu cầu tìm dữ liệu của chúng trong bộ nhớ chính. 12.6 Tại sao điều quan trọng là phải cân bằng I / O hệ thống tệp giữa các đĩa và bộ điều khiển trên hệ thống trong môi trường đa nhiệm? Trả lời: Một hệ thống chỉ có thể hoạt động ở tốc độ tắc nghẽn chậm nhất của nó. Đĩa hoặc bộ điều khiển đĩa thường là nút thắt cổ chai trong hiện đại hệ thống vì hiệu suất riêng lẻ của chúng không thể theo kịp với CPU và bus hệ thống. Bằng cách cân bằng I / O giữa các đĩa và bộ điều khiển, không có đĩa riêng lẻ hay bộ điều khiển nào bị quá tải, do đó nút thắt cổ chai được tránh. Bài tập thực hành 45

12.7. Những đánh đổi liên quan đến việc đọc lại các trang mã từ tệp là gì hệ thống so với sử dụng không gian hoán đổi để lưu trữ chúng?

Trả lời: Nếu các trang mã được lưu trữ trong không gian hoán đổi, chúng có thể được chuyển nhanh hơn vào bộ nhớ chính (vì phân bổ không gian hoán đổi được điều chỉnh cho hiệu suất nhanh hơn so với phân bổ hệ thống tệp chung). Sử dụng hoán đổi không gian có thể yêu cầu thời gian khởi động nếu các trang được sao chép ở đó trong quá trình xử lý thay vì chỉ được phân trang ra để hoán đổi không gian theo yêu cầu. Ngoài ra, nhiều không gian hoán đổi hơn phải được phân bổ nếu nó được sử dụng cho cả mã và các trang dữ liệu.

12.8. Có cách nào để triển khai bộ nhớ thực sự ổn định không? Giải thich câu trả lơi của bạn.

Trả lời: Bộ nhớ thực sự ổn định sẽ không bao giờ mất dữ liệu. Cơ bản kỹ thuật để lưu trữ ổn định là duy trì nhiều bản sao của dữ liệu, để nếu một bản sao bị hủy, một số bản sao khác vẫn có sẵn cho sử dụng. Nhưng đối với bất kỳ kế hoạch nào, chúng ta có thể tưởng tượng một thảm họa đủ lớn tất cả các bản sao bị phá hủy.

12.9. Thuật ngữ "SCSI-II rộng nhanh" biểu thị một bus SCSI hoạt động tại một dữ liệu tốc độ 20 megabyte mỗi giây khi nó di chuyển một gói byte giữa máy chủ và thiết bị. Giả sử rằng ổ đĩa SCSI-II rộng nhanh quay ở tốc độ 7200 RPM, có kích thước khu vực là 512 byte và chứa 160 khu vực mỗi bản nhạc.

a. Ước tính tốc độ truyền liên tục của ổ đĩa này tính bằng megabyte môi giây.

b. Giả sử rằng ổ đĩa có 7000 xi lanh, 20 rãnh trên mỗi xi lanh, thời gian chuyển đổi đầu (từ đĩa này sang đĩa khác) là 0,5 mili giây và thời gian tìm kiếm hình trụ liền kề là 2 mili giây. Sử dụng thông tin bổ sung này để đưa ra ước tính chính xác về tốc độ truyền liên tục cho một khoản chuyển lớn.

c. Giả sử rằng thời gian tìm kiếm trung bình cho ổ đĩa là 8 mili giây. Ước tính I / Os mỗi giây và tốc độ truyền tải hiệu quả cho một khối lượng công việc truy cập ngẫu nhiên đọc các khu vực riêng lẻ rải rác trên đĩa.

d. Tính I / Os truy cập ngẫu nhiên mỗi giây và tốc độ truyền cho các kích thước I / O là 4 kilobyte, 8 kilobyte và 64 kilobyte.

e. Nếu nhiều yêu cầu nằm trong hàng đợi, một thuật toán lập lịch như vì SCAN sẽ có thể giảm khoảng cách tìm kiếm trung bình. Giả sử rằng khối lượng công việc truy cập ngẫu nhiên đang đọc các trang 8 kilobyte, độ dài hàng đợi trung bình là 10 và thuật toán lập lịch giảm thời gian tìm kiếm trung bình xuống 3 mili giây. Bây giờ hãy tính toán I / Os mỗi giây và tốc độ truyền tải hiệu quả của ổ đĩa.

Trả lời:

a. Đĩa quay 120 lần mỗi giây và mỗi vòng quay chuyển một theo dõi 80 KB. Do đó, tốc độ truyền duy trì có thể được tính gần đúng là 9600 KB / s.

b. Giả sử rằng 100 xi lanh là một sự chuyển giao rất lớn. Tốc độ truyền là tổng số byte chia cho tổng thời gian. Số byte: 100 cyl * 20 trk / cyl * 80 KB / trk, tức là 160.000 KB. Thời gian: thời gian quay + thời gian chuyển đổi theo dõi + thời gian chuyển đổi xi lanh. Thời gian quay vòng là 2000 trks / 120 trks mỗi giây, tức là 16,667 giây. Thời gian chuyển mạch theo dõi là 19 công tắc mỗi cyl * 100 cyl * 0,5 mili giây, tức là 950 mili giây. Thời gian chuyển đổi xi lanh là 99 * 2 ms, tức là 198 ms. Như vậy, tổng thời gian là 16,667 + 0,950 + 0,198, tức là 17,815 giây. (Chúng tôi đang bỏ qua mọi tìm kiếm ban đầu và độ trễ xoay vòng, điều này có thể thêm khoảng 12 mili giây vào lịch trình, tức là 0,1%.) Do đó, tốc độ truyền là 8981,2 KB / s. Chi phí của việc chuyển đổi đường ray và xi lanh là khoảng 6,5%.

c. Thời gian mỗi lần chuyển là 8 ms để tìm kiếm + vòng quay trung bình 4,167 ms độ trễ + 0,052 ms (được tính từ 1 / (120 trk mỗi giây * 160 sector per trk)) để xoay một sector qua đầu đĩa trong đọc hiểu. Chúng tôi tính số lần chuyển mỗi giây là 1 / (0,012219), tức là, 81,8. Vì mỗi lần truyền là 0,5 KB nên tốc độ truyền là 40,9 KB / s.

d. Chúng tôi bỏ qua các giao cắt đường ray và xi lanh để đơn giản hóa. Để đọc kích thước 4 KB, 8 KB và 64 KB, I / Os tương ứng mỗi giây là được tính toán từ lượt tìm kiếm, độ trễ quay và thời gian truyền xoay vòng như trong mục trước, cho (tương ứng) 1 / (0,0126), 1 / (0,013) và 1 / (0,019). Do đó, chúng tôi nhận được lần lượt 79,4, 76,9 và 52,6 chuyển mỗi giây. Tỷ lệ chuyển khoản thu được từ 4, 8 và 64 lần các tốc độ I / O này, cho 318 KB / s, 615 KB / s và 3366 KB / s, tương ứng.

e. Từ 1 / (3 + 4,167 + 0,83), chúng tôi thu được 125 I / Os mỗi giây. Từ 8 KB mỗi I / O, chúng tôi thu được 1000 KB / s.

12.10. Có thể gắn nhiều ổ đĩa vào một bus SCSI. Đặc biệt, một bus SCSI-II rộng nhanh (xem Bài tập 12.9) có thể được kết nối với tối đa 15 ổ đĩa. Nhớ lại rằng bus này có băng thông 20 megabyte môi giây. Tại bất kỳ thời điểm nào, chỉ có thể chuyển một gói trên xe buýt giữa một số bộ nhớ cache bên trong của đĩa và máy chủ. Tuy nhiên, một chiếc đĩa có thể đang di chuyển cánh tay đĩa của nó trong khi một số đĩa khác đang chuyển một gói trên xe buýt. Ngoài ra, một đĩa có thể truyền dữ liệu giữa các đĩa cứng và bộ nhớ cache bên trong của nó trong khi một số đĩa

khác đang chuyển gói trên xe buýt. Xem xét tốc độ chuyển tiền mà bạn đã tính toán đối với các khối lượng công việc khác nhau trong Bài tập 12.9, hãy thảo luận xem có bao nhiêu đĩa có thể được sử dụng hiệu quả bởi một bus SCSI-II rộng nhanh.

Trả lời:
Đối với I / Os ngẫu nhiên 8 KB trên một đĩa được tải nhẹ, nơi truy cập ngẫu nhiên thời gian được tính là khoảng 13 ms (xem Bài tập 12.9), hiệu quả tốc độ truyền khoảng 615 MB / s. Trong trường hợp này, 15 đĩa sẽ có tốc độ truyền tổng hợp dưới 10 MB / s, không bão hòa xe buýt. Đối với các lần đọc ngẫu nhiên 64 KB vào một đĩa được tải nhẹ, việc chuyển tốc độ khoảng 3,4 MB / s, do đó, năm ổ đĩa trở xuống sẽ bão hòa xe buýt. Đối với 8 KB lượt đọc với hàng đợi đủ lớn để giảm mức trung bình tìm kiếm đến 3 ms, tốc độ truyền khoảng 1 MB / s, do đó băng thông xe buýt có thể đủ để chứa 15 đĩa.

12.11. Việc loại bỏ các khối xấu bằng cách tiết kiệm khu vực hoặc trượt ngành có thể ảnh hưởng đến hiệu suất. Giả sử rằng ổ đĩa trong Bài tập 12.9 có tổng số 100 khu vực xấu tại các vị trí ngẫu nhiên và mỗi khu vực xấu là được ánh xạ tới một phụ tùng nằm trên một tuyến đường khác, nhưng trong cùng một hình trụ. Ước tính số I / Os mỗi giây và tốc độ truyền hiệu quả cho khối lượng công việc truy cập ngẫu nhiên bao gồm 8 kilobyte đọc, với độ dài hàng đợi là 1 (nghĩa là sự lựa chọn của thuật toán lập lịch không phải là một yếu tố). Ảnh hưởng của một khu vực xấu đến hiệu suất là gì?

Trả lời:
Vì đĩa chứa 22.400.000 cung nên xác suất yêu cầu một trong 100 lĩnh vực được ánh xạ lại là rất nhỏ. Một ví dụ trong trường hợp xấu nhất là chúng tôi cố đọc một trang 8 KB, nhưng một khu vực từ giữa bị lỗi và đã được ánh xạ lại thành vị trí xấu nhất có thể trên một đường ray khác trong hình trụ đó. Trong trường hợp này, thời gian để truy xuất có thể là 8 mili giây để tìm kiếm, cộng với hai công tắc theo dõi và hai độ trễ xoay đầy đủ. Có khả năng là một bộ điều khiển hiện đại sẽ đọc tất cả các sector tốt được yêu cầu từ bản nhạc gốc trước đây chuyển sang rãnh dự phòng để truy xuất khu vực được ánh xạ lại và do đó sẽ chỉ phải chịu một công tắc theo dõi và độ trễ quay. Vì vậy, thời gian sẽ là 8 ms tìm kiếm + độ trễ quay trung bình 4,17 ms + 0,05 ms công tắc theo dõi + độ trễ xoay 8,3 ms + thời gian đọc 0,83 ms (8 KBis 16 các cung, 1/10 của một vòng quay theo dõi). Do đó, thời gian để thực hiện yêu cầu này sẽ là 21,8 mili giây, cho tốc độ I / O là 45,9 yêu cầu mỗi giây và băng thông hiệu dụng là 367 KB / s. Đối với một thời gian bị hạn chế nghiêm trọng ứng dụng này có thể quan trọng, nhưng tác động tổng thể trong trung bình của 100 lĩnh vực được ánh xạ lại và 22,4 triệu lĩnh vực tốt là con số không.

12.12. Trong máy hát đĩa, tác dụng của việc có nhiều tệp mở hơn so với số lượng ổ đĩa trong máy hát tự động?

Trả lời: Hai kết quả xấu có thể dẫn đến. Một khả năng là chết đói trong số các ứng dụng gây ra sự cố chặn I / Os đối với các băng không được gắn kết trong các ổ đĩa. Một khả năng khác là đập, khi máy hát tự động được chỉ huy để chuyển đổi băng sau mỗi thao tác I / O.

12.13. Nếu đĩa cứng từ tính cuối cùng có cùng chi phí cho mỗi gigabyte băng, băng sẽ trở nên lỗi thời, hay chúng vẫn còn cần thiết? Giải thích câu trả lời của bạn.
Trả lời:
Băng keo có thể tháo rời dễ dàng, vì vậy chúng rất hữu ích cho các công việc bên ngoài sao lưu và để chuyển hàng loạt dữ liệu (bằng cách gửi hộp mực). Như một quy tắc, một đĩa cứng từ tính không phải là một phương tiện di động.

12.14. Đôi khi người ta nói rằng băng là một phương tiện truy cập tuần tự, trong khi đĩa từ là một phương tiện truy cập ngẫu nhiên. Trên thực tế, sự phù hợp của một thiết bị lưu trữ để truy cập ngẫu nhiên phụ thuộc vào kích thước truyền. Các thuật ngữ tốc độ truyền trực tuyến biểu thị tốc độ dữ liệu cho một lần truyền đang được tiến hành, loại trừ ảnh hưởng của độ trễ truy cập. Ngược lại, tốc độ truyền tải hiệu quả là tỷ lệ tổng số byte trên tổng số giây, bao gồm thời gian trên không, chẳng hạn như độ trễ truy cập. Giả sử rằng, trong máy tính, bộ đệm cấp 2 có độ trễ truy cập 8 nano giây và tốc độ truyền trực tuyến 800 megabyte mỗi thứ hai, bộ nhớ chính có độ trễ truy cập là 60 nano giây và tốc độ truyền trực tuyến 80 megabyte mỗi giây, đĩa từ tính có độ trễ truy cập là 15 mili giây và tốc độ truyền trực tuyến là 5 megabyte mỗi giây và ổ băng có độ trễ truy cập là 60 giây và tốc độ truyền trực tuyến là 2 megabyte mỗi giây.

a. Truy cập ngẫu nhiên làm cho tốc độ truyền tải hiệu quả của một thiết bị giảm, vì không có dữ liệu nào được chuyển trong thời gian truy cập. Đối với đĩa được mô tả, tốc độ truyền tải hiệu quả là bao nhiêu nếu một truy cập trung bình được theo sau bởi truyền trực tuyến 512 byte, 8 kilobyte, 1 megabyte và 16 megabyte?

b. Việc sử dụng một thiết bị là tỷ lệ truyền hiệu quả tốc độ truyền trực tuyến. Tính toán việc sử dụng ổ đĩa để truy cập ngẫu nhiên thực hiện chuyển trong mỗi bốn kích thước được đưa ra trong phần a.

c. Giả sử rằng mức sử dụng 25 phần trăm (hoặc cao hơn) được coi là có thể chấp nhận được. Sử dụng các số liệu hiệu suất đã cho, tính toán kích thước truyền nhỏ nhất cho đĩa mang lại hiệu quả sử dụng có thể chấp nhận được.

d. Hoàn thành câu sau: Đĩa là thiết bị truy cập ngẫu nhiên để truyền lớn hơn byte và là thiết bị truy cập tuần tự để truyền nhỏ hơn.

e. Tính toán kích thước truyền tải tối thiểu để có thể sử dụng được cho bộ nhớ đệm, bộ nhớ và băng từ.

f. Khi nào thì một cuộn băng là thiết bị truy cập ngẫu nhiên và khi nào thì nó là thiết bị truy cập tuần tự?

Trả lời:

a. Đối với 512 byte, tốc độ truyền tải hiệu quả được tính như sau. ETR = quy mô chuyển nhượng / thời gian chuyển nhượng. Nếu X là kích thước truyền, thì thời gian truyền là ((X / STR) + độ trễ). Thời gian truyền là 15ms + (512B / 5MB mỗi giây) = 15.0097ms. Do đó, tốc độ truyền tải hiệu quả là 512B / 15.0097ms = 33.12 KB / giây. ETR cho 8KB = .47MB / giây. ETR cho 1MB = 4,65MB / giây. ETR cho 16MB = 4,98MB / giây.

b. Sử dụng thiết bị cho 512B = 33,12 KB / giây / 5MB / giây = .0064 = .64 Đối với 8KB = 9,4%. Đối với 1MB = 93%. Đối với 16MB = 99,6%.

c. Tính .25 = ETR / STR, giải quyết kích thước truyền X. STR = 5MB, vì vậy 1,25MB / S = ETR. 1.25MB / S * ((X / 5) + .015) = X. .25X + .01875 = X. X = 0,025 MB.

d. Đĩa là thiết bị truy cập ngẫu nhiên để truyền lớn hơn K byte (trong đó K> kích thước khối đĩa), và là thiết bị truy cập tuần tự cho chuyển khoản nhỏ hơn. Bài tập thực hành 49

e. Tính toán kích thước truyền tối thiểu để sử dụng có thể chấp nhận được bộ nhớ cache: STR = 800MB, ETR = 200, độ trễ = 8 * 10−9. 200 (XMB / 800 + 8 X 10−9) = XMB. .25XMB + 1600 * 10−9 = XMB. X = 2,24 byte. Tính toán cho bộ nhớ: STR = 80MB, ETR = 20, L = 60 * 10−9. 20 (XMB / 80 + 60 * 10−9) = XMB. .25XMB + 1200 * 10−9 = XMB. X = 1,68 byte. Tính toán cho băng: STR = 2MB, ETR = 0,5, L = 60 giây. .5 (XMB / 2 + 60) = XMB. .25XMB + 30 = XMB. X = 40 MB.

f. Nó phụ thuộc vào cách nó được sử dụng. Giả sử chúng tôi đang sử dụng băng để khôi phục bản sao lưu. Trong trường hợp này, cuộn băng hoạt động như một thiết bị truy cập tuần tự nơi chúng tôi đang đọc tuần tự nội dung của băng. Như một ví dụ khác, giả sử chúng ta đang sử dụng băng để truy cập nhiều bản ghi được lưu trữ trên băng. Trong này ví dụ, quyền truy cập vào băng là tùy ý và do đó được coi là ngẫu nhiên

12.15. Giả sử rằng chúng ta đồng ý rằng 1 kilobyte là 1,024 byte, 1 megabyte là 1,0242 byte và 1 gigabyte là 1,0243 byte. Tiến trình này tiếp tục thông qua terabyte, petabyte và exabyte (1,0246). Hiện tại có một số dự án khoa học mới được đề xuất có kế hoạch ghi lại và lưu trữ vài exabyte dữ liệu trong thập kỷ tới.

Để trả lời những điều sau câu hỏi, bạn sẽ cần phải đưa ra một vài giả định hợp lý; tình trạng những giả định mà bạn đưa ra.
a. Cần bao nhiêu ổ đĩa để chứa 4 exabyte dữ liệu?
b. Cần bao nhiêu băng từ để chứa 4 exabyte Dữ liệu?
c. Cần bao nhiêu băng quang để chứa 4 exabyte dữ liệu (xem Bài tập 12.21)?
d. Cần bao nhiêu hộp mực lưu trữ ba chiều để giữ 4 exabyte dữ liệu (xem Bài tập 12.20)?
e. Mỗi tùy chọn sẽ yêu cầu bao nhiêu feet khối dung lượng lưu trữ?

Trả lời:

a. Giả sử rằng ổ đĩa chứa 10 GB. Sau đó 100 đĩa giữ 1 TB, 100.000 đĩa chứa 1 PB và 100.000.000 đĩa chứa 1 EB. Đến cửa hàng 4 EB sẽ yêu cầu khoảng 400 triệu đĩa. Nếu một từ tính băng chứa 40 GB, chỉ cần 100 triệu băng. Nếu Cấu trúc lưu trữ khối một băng quang chứa dữ liệu nhiều hơn 50 lần so với băng từ, 2 hàng triệu băng quang học là đủ. Nếu một hộp mực ba chiều có thể lưu trữ 180 GB, khoảng 22,2 triệu hộp mực sẽ được yêu cầu.

b. Ổ đĩa 3,5 "cao khoảng 1", rộng 4 "và sâu 6". Bằng đơn vị feet, đây là 1/12 x 1/3 x 1/2, hoặc 1/72 feet khối. Đóng gói dày đặc, 400 triệu đĩa sẽ chiếm 5,6 triệu feet khối. Nếu chúng ta cho phép một hệ số hai đối với không gian không khí và không gian cho nguồn cung cấp điện, công suất cần thiết là khoảng 11 triệu feet khối.

c. Hộp băng 1/2 "có kích thước khoảng 1" cao và 4,5 "hình vuông. Thể tích khoảng 1/85 feet khối. Đối với 100 triệu băng từ được đóng gói dày đặc, khối lượng khoảng 1,2 triệu feet khối. Dành cho 2 triệu băng quang học, thể tích là 23.400 feet khối.

d. Một đĩa CD-ROM có đường kính 4,75 "và dày khoảng 1/16". Nếu chúng ta giả sử rằng một đĩa holo store được lưu trữ trong một khe thư viện là 5 " hình vuông và rộng 1/8 ", chúng tôi tính toán khối lượng là 22,2 triệu đĩa có kích thước khoảng 40.000 feet khối.

Chương 13: I/O Systems

13.1 : Nêu ba ưu điểm và nhược điểm của việc đặt chức năng trong bộ điều khiển thiết bị, chứ không phải trong hạt nhân.

Trả lời:

Ba ưu điểm:

- Lỗi ít có khả năng gây ra hoạt động sự cố hệ thống

- Hiệu suất có thể được cải thiện bằng cách sử dụng phần cứng chuyên dụng và các thuật toán mã hóa cứng
- Kernel được đơn giản hóa bằng cách chuyển các thuật toán ra khỏi nó

Ba nhược điểm:

- Lỗi khó sửa hơn — một phiên bản chương trình cơ sở mới hoặc phần cứng mới là cần thiết
- Việc cải thiện các thuật toán cũng yêu cầu cập nhật phần cứng hơn là chỉ là bản cập nhật nhân hoặc trình điều khiển thiết bị
- Các thuật toán được nhúng có thể xung đột với việc sử dụng thiết bị của ứng dụng, làm giảm hiệu suất

13.2 : Ví dụ về bắt tay trong Phần 13.2 sử dụng 2 bit: một bit bận và một bit sẵn sàng lệnh. Có thể thực hiện việc bắt tay này với chỉ 1 chút? Nếu có, hãy mô tả giao thức. Nếu không, hãy giải thích tại sao 1 bit không đủ ?

Trả lời : Có thể, bằng cách sử dụng thuật toán sau đây. Hãy giả sử chúng tôi chỉ đơn giản sử dụng bit bận (hoặc bit sẵn sàng cho lệnh; câu trả lời này là như nhau không phân biệt). Khi bit tắt, bộ điều khiển không hoạt động. Các máy chủ ghi vào dữ liệu ra và đặt bit để báo hiệu rằng một hoạt động đang sẵn sàng (tương đương với việc thiết lập bit sẵn sàng cho lệnh). Khi mà bộ điều khiển kết thúc, nó sẽ xóa bit bận. Máy chủ sau đó khởi tạo hoạt động tiếp theo. Giải pháp này yêu cầu cả máy chủ và bộ điều khiển phải đọc và ghi quyền truy cập vào cùng một bit, điều này có thể làm phức tạp mạch và tăng giá thành của bộ điều khiển.

13.3 : Tại sao một hệ thống có thể sử dụng I / O hướng ngắt để quản lý một chuỗi đơn cổng, nhưng thăm dò I / O để quản lý bộ xử lý front-end, chẳng hạn như thiết bị đầu cuối máy tập trung?

Trả lời : Thăm dò ý kiến có thể hiệu quả hơn I / O điều khiển gián đoạn. Cái này là trường hợp I / O diễn ra thường xuyên và trong thời gian ngắn. Mặc dù một cổng nối tiếp duy nhất sẽ thực hiện I / O tương đối không thường xuyên và nên do đó sử dụng ngắt, một tập hợp các cổng nối tiếp, chẳng hạn như các cổng trong một thiết bị đầu cuối bộ tập trung có thể tạo ra nhiều hoạt động I / O ngắn và làm gián đoạn đối với mỗi cái có thể tạo ra một tải nặng trên hệ thống. Đúng lúc vòng lặp thăm dò có thể làm giảm tải mà không lãng phí nhiều tài nguyên thông qua vòng lặp mà không cần I / O.

13.4: Thăm dò ý kiến để hoàn thành I / O có thể lãng phí một số lượng lớn chu kỳ CPU nếu bộ xử lý lặp lại một vòng lặp chờ bận nhiều lần trước I / O hoàn thành. Nhưng nếu thiết bị I / O đã sẵn sàng để phục vụ, việc thăm dò ý kiến có thể hiệu quả hơn là bắt và điều phối một ngắt. Diễn tả

một chiến lược kết hợp kết hợp thăm dò ý kiến, ngủ và ngắt quãng cho Dịch vụ thiết bị I / O. Đối với mỗi chiến lược trong số ba chiến lược này (bỏ phiếu thuần túy, thuần túy ngắt, kết hợp), mô tả một môi trường máy tính trong đó chiến lược hiệu quả hơn một trong hai chiến lược khác ?

Trả lời: Một phương pháp kết hợp có thể chuyển đổi giữa bỏ phiếu và ngắt tùy thuộc vào thời gian chờ thao tác I / O. Ví dụ, chúng tôi có thể thăm dò và lặp lại N lần và nếu thiết bị vẫn bận ở N + 1, chúng tôi có thể thiết lập một gián đoạn và ngủ. Cách tiếp cận này sẽ tránh được lâu chu kỳ chờ đợi bận rộn. Phương pháp này sẽ tốt nhất trong thời gian dài hoặc rất thời gian bận rộn ngắn. Sẽ không hiệu quả nếu I / O hoàn thành ở N + T (trong đó T là một số chu kỳ nhỏ) do chi phí bỏ phiếu cộng với việc thiết lập và bắt ngắt. Bỏ phiếu thuần túy là tốt nhất với thời gian chờ đợi rất ngắn. Ngắt tốt nhất với đã biết thời gian chờ đợi lâu.

13.5 : DMA làm tăng tính đồng thời của hệ thống như thế nào? Nó phức tạp như thế nào thiết kế phần cứng?

Trả lời: DMA tăng tính đồng thời của hệ thống bằng cách cho phép CPU để thực hiện các tác vụ trong khi hệ thống DMA truyền dữ liệu qua hệ thống và xe buýt bộ nhớ. Thiết kế phần cứng phức tạp vì DMA bộ điều khiển phải được tích hợp vào hệ thống và hệ thống phải cho phép bộ điều khiển DMA làm bộ điều khiển bus. Ăn cắp chu kỳ cũng có thể cần thiết để cho phép CPU và bộ điều khiển DMA chia sẻ quyền sử dụng bus bộ nhớ.

13.6: Tại sao việc mở rộng tốc độ bus hệ thống và thiết bị lại quan trọng vì Tốc độ CPU tăng lên?

Trả lời: Hãy xem xét một hệ thống thực hiện 50% I / O và 50% tính toán. Tăng gấp đôi hiệu suất CPU trên hệ thống này sẽ tang tổng hiệu suất hệ thống chỉ giảm 50%. Nhân đôi cả hai khía cạnh hệ thống sẽ tăng hiệu suất lên 100%. Nói chung, điều quan trọng là phải loại bỏ điểm nghẽn hệ thống hiện tại và tăng hệ thống tổng

thể hiệu suất, thay vì tăng hiệu suất của các thành phần hệ thống riêng lẻ một cách mù quáng.

13.7 Phân biệt giữa trình điều khiển STREAMS và mô-đun STREAMS.

Trả lời: Trình điều khiển STREAMS điều khiển một thiết bị vật lý có thể tham gia vào hoạt động STREAMS. Mô-đun STREAMS sửa đổi luồng dữ liệu giữa phần đầu (giao diện người dùng) và trình điều khiển.

Chương 14: Sự bảo vệ

14.1. Sự khác biệt chính giữa danh sách khả năng và danh sách truy cập là gì?

Trả lời: Danh sách truy cập là danh sách cho mỗi đối tượng bao gồm các miền với một tập hợp quyền truy cập không có gì cho đối tượng đó. Một danh sách khả năng là danh sách các đối tượng và các hoạt động được phép trên các đối tượng đó cho mỗi miền.

14.2. Tệp MCP Burroughs B7000 / B6000 có thể được gắn thẻ là dữ liệu nhạy cảm. Khi một tệp như vậy bị xóa, vùng lưu trữ của nó sẽ bị ghi đè bởi một số các bit ngẫu nhiên. Vì mục đích gì mà một kế hoạch như vậy sẽ hữu ích?

Trả lời: Điều này sẽ hữu ích như một biện pháp bảo mật bổ sung để nội dung cũ của bộ nhớ không thể được truy cập, dù cố ý hay bằng cách tai nạn, bởi một chương trình khác. Điều này đặc biệt hữu ích cho bất kỳ thông tin mật.

14.3. Trong hệ thống bảo vệ vòng, cấp 0 có quyền truy cập lớn nhất vào các đối tượng, và cấp n (lớn hơn 0) có ít quyền truy cập hơn. Quyền truy cập của một chương trình ở một cấp cụ thể trong cấu trúc vòng được coi là một tập hợp các khả năng. Mối quan hệ giữa các khả năng là gì của miền ở cấp j và miền ở cấp i đối với một đối tượng (đối với j> i)?

Trả lời: Dj là tập con của Di.

14.4. Hệ thống RC 4000 (và các hệ thống khác) đã xác định một cây các quy trình (được gọi là cây quy trình) sao cho tất cả các con của một quy trình được đưa ra tài nguyên (đối tượng) và quyền truy cập chỉ bởi tổ tiên của họ. Do đó, một con cháu không bao giờ có thể có khả năng làm bất cứ điều gì mà tổ tiên của nó không thể

làm. Gốc của cây là hệ điều hành, có khả năng làm bất cứ điều gì. Giả sử tập hợp các quyền truy cập đã được đại diện bởi một ma trận truy cập, A. A (x, y) xác định các quyền truy cập của quá trình x đối với đối tượng y. Nếu x là con của z thì mối quan hệ giữa A (x, y) là gì và A (z, y) cho một đối tượng tùy ý y?

Trả lời: A (x, y) là tập con của A (z, y).

14.5. Những vấn đề bảo vệ nào có thể phát sinh nếu một ngăn xếp dùng chung được sử dụng để truyền tham số?

Trả lời: Nội dung của ngăn xếp có thể bị xâm phạm bởi người khác (các) quy trình chia sẻ ngăn xếp.

14.6. Xem xét một môi trường máy tính trong đó một số duy nhất được liên kết với mỗi quá trình và mỗi đối tượng trong hệ thống. Giả sử rằng chúng ta cho phép một quá trình có số n truy cập vào một đối tượng chỉ có số m nếu n> m. Chúng ta có loại cấu trúc bảo vệ nào?

Trả lời: Cấu trúc thứ bậc.

14.7. Hãy xem xét một môi trường máy tính trong đó một tiến trình chỉ được cấp đặc quyền truy cập một đối tượng n lần. Đề xuất phương án thực hiện chính sách này.

Trả lời: Thêm một bộ đếm số nguyên với khả năng.

14.8. Nếu tất cả các quyền truy cập đối với một đối tượng bị xóa, đối tượng đó không còn có thể được truy cập. Tại thời điểm này, đối tượng cũng sẽ bị xóa và không gian nó chiếm sẽ được trả lại cho hệ thống. Đề xuất một hiệu quả thực hiện Đề án này.

Trả lời: Số lượng tham chiếu.

14.9. Tại sao khó bảo vệ hệ thống mà người dùng được phép làm I / O của riêng họ?

Trả lời: Trong các chương trước, chúng tôi đã xác định sự khác biệt giữa hạt nhân và chế độ người dùng trong đó chế độ hạt nhân được sử dụng để thực hiện đặc

quyền chẳng hạn như I / O. Một lý do tại sao I / O phải được thực hiện trong chế độ hạt nhân là I / O yêu cầu truy cập vào phần cứng và quyền truy cập vào phần cứng là cần thiết để đảm bảo tính toàn vẹn của hệ thống. Nếu chúng tôi cho phép người dùng thực hiện I / O của riêng họ, chúng tôi không thể đảm bảo tính toàn vẹn của hệ thống.

14.10. Danh sách khả năng thường được giữ trong không gian địa chỉ của người dùng. Làm thế nào để hệ thống đảm bảo rằng người dùng không thể sửa đổi nội dung của danh sách?

Trả lời: Danh sách khả năng được coi là "đối tượng được bảo vệ" và chỉ được truy cập gián tiếp bởi người dùng. Hệ điều hành đảm bảo người dùng không thể truy cập trực tiếp vào danh sách khả năng.

Chương 16: Được phân phối - Hệ thống - cấu trúc

16.1. Hầu hết các mạng WAN chỉ sử dụng cấu trúc liên kết được kết nối một phần. Tại sao cái này Vì thế?

Trả lời: Chi phí. Một mạng được kết nối đầy đủ yêu cầu liên kết giữa

mọi nút trong mạng. Đối với mạng WAN, điều này có thể quá tốn kém vì các liên kết giao tiếp giữa các máy chủ ở xa về mặt vật lý có thể tốn kém.

16.2. Trong những trường hợp nào thì mạng chuyển mã thông báo hiệu quả hơn hơn mạng Ethernet?

Trả lời: Vòng mã thông báo rất hiệu quả trong điều kiện tải liên tục cao, như không có va chạm có thể xảy ra và mỗi vị trí có thể được sử dụng để mang một thông điệp, cung cấp thông lượng cao. Vòng mã thông báo kém hiệu quả hơn khi tải nhẹ (xử lý mã thông báo mất nhiều thời gian hơn so với truy cập xe buýt, vì vậy bất kỳ gói có thể mất nhiều thời gian hơn để đến đích) hoặc rời rạc.

16.3. Tại sao việc các cổng truyền các gói tin quảng bá lại là một ý tưởng tồi giữa các mạng? Lợi ích của việc làm như vậy là gì?

Trả lời: Tất cả các chương trình phát sóng sẽ được truyền tới tất cả các mạng, khiến rất nhiều lưu lượng mạng. Nếu lưu lượng phát sóng bị giới hạn ở mức quan trọng dữ liệu (và rất ít trong số đó), sau đó truyền phát quảng bá sẽ tiết kiệm các cổng từ việc phải chạy phần mềm đặc biệt để theo dõi dữ liệu này (chẳng hạn như thông tin định tuyến mạng) và phát lại nó.

16.4. Thảo luận về những ưu điểm và nhược điểm của các bản dịch tên bộ đệm cho các máy tính đặt trong các miền từ xa.

Trả lời: Có một lợi thế về hiệu suất đối với các bản dịch tên trong bộ nhớ đệm cho các máy tính nằm trong các miền từ xa: độ phân giải lặp đi lặp lại của cùng một tên từ các máy tính khác nhau nằm trong miền cục bộ có thể được thực hiện cục bộ mà không yêu cầu tra cứu tên từ xa hoạt động. Điểm bất lợi là có thể có sự mâu thuẫn trong bản dịch tên khi cập nhật được thực hiện trong ánh xạ tên tới Các địa chỉ IP. Những vấn đề về tính nhất quán này có thể được giải quyết bằng cách làm mất hiệu lực của các bản dịch, điều này sẽ yêu cầu quản lý nhà nước về máy tính nào đang lưu vào bộ nhớ đệm một bản dịch nhất định và cũng sẽ yêu cầu một số thông báo vô hiệu hoặc bằng cách sử dụng hợp đồng thuê thực thể bộ nhớ đệm làm mất hiệu lực bản dịch sau một khoảng thời gian nhất định. Cách tiếp cận thứ hai yêu cầu ít trạng thái hơn và không có thông báo vô hiệu nhưng có thể bị mâu thuẫn tạm thời.

16.5. Ưu điểm và nhược điểm của việc sử dụng chuyển mạch kênh là gì? Đối với những loại ứng dụng nào, chuyển mạch kênh là một chiến lược khả thi?

Trả lời: Chuyển mạch kênh đảm bảo rằng các tài nguyên mạng cần thiết cho quá trình truyền được dự trữ trước khi quá trình truyền diễn ra. Điều này đảm bảo rằng các gói sẽ không bị rơi và việc phân phối của chúng sẽ đáp ứng yêu cầu về chất lượng dịch vụ. Nhược điểm của mạch chuyển đổi là nó yêu cầu một tin nhắn khứ hồi để thiết lập đặt chỗ và nó cũng có thể cung cấp quá mức tài nguyên, do đó dẫn đến sử dụng tài nguyên dưới mức tối ưu. Chuyển mạch kênh là một chiến lược khả thi cho các ứng dụng có nhu cầu liên tục về mạng tài nguyên và sẽ yêu cầu tài nguyên trong thời gian dài, do đó phân bổ chi phí chung ban đầu.

16.6. Hai vấn đề ghê gớm mà các nhà thiết kế phải giải quyết để triển khai một hệ thống mạng trong suốt là gì?

Trả lời: Một vấn đề như vậy là làm cho tất cả các bộ xử lý và bộ nhớ thiết bị có vẻ trong suốt trên mạng. Nói cách khác, hệ thống phân tán nên xuất hiện như một hệ thống tập trung cho người dùng. Các Hệ thống tệp Andrew và NFS cung cấp tính năng này: tệp phân tán hệ thống xuất hiện với người dùng dưới dạng một hệ thống tệp duy nhất nhưng trên thực tế, nó có thể được phân phối trên một mạng lưới. Một vấn đề khác liên quan đến tính di động của người dùng. Chúng tôi muốn cho phép người dùng để kết nối với "hệ thống" thay vì với một máy cụ thể (mặc dù trong thực tế, họ có thể đang đăng nhập vào một máy cụ thể ở đâu đó trong hệ thống phân tán).

16.7. Di chuyển quy trình trong một mạng không đồng nhất thường là không thể, do sự khác biệt về kiến trúc và hệ điều hành. Mô tả một phương pháp để di chuyển quy trình trên các kiến trúc khác nhau

đang chạy:

a. Hệ điều hành giống nhau

b. Hệ điều hành khác nhau

Trả lời: Đối với cùng một hệ điều hành, quá trình di chuyển tương đối đơn giản, vì trạng thái của quá trình cần di chuyển từ bộ xử lý này sang bộ xử lý khác. Điều này liên quan đến việc di chuyển không gian địa chỉ, trạng thái của các thanh ghi CPU, và mở các tệp từ hệ thống nguồn đến đích. Tuy nhiên, điều quan trọng là các bản sao giống hệt nhau của hoạt động hệ thống đang chạy trên các hệ thống khác nhau để đảm bảo tính tương thích. Nếu hệ điều hành giống nhau, nhưng có thể các phiên bản khác nhau đang chạy trên các hệ thống riêng biệt, thì quá trình di chuyển phải đảm bảo tuân theo các nguyên tắc lập trình nhất quán giữa các phiên bản khác nhau của hệ điều hành. Các ứng dụng Java cung cấp một ví dụ hay về quá trình di chuyển giữa các hệ điều hành khác nhau. Để che giấu sự khác biệt trong hệ thống cơ bản,quá trình đã di chuyển (tức là một ứng dụng Java) chạy trên một máy ảo chứ không phải là một hệ điều hành cụ thể. Tất cả những gì được yêu cầu là cho máy ảo đang chạy trên hệ thống mà tiến trình di chuyển đến.

16.8. Để xây dựng một hệ thống phân tán mạnh mẽ, bạn phải biết những loại hỏng hóc có thể xảy ra.

a. Liệt kê ba loại lỗi có thể xảy ra trong hệ thống phân tán.

b. Chỉ định mục nào trong danh sách của bạn cũng có thể áp dụng cho hệ thống tập trung.

Trả lời: Ba lỗi phổ biến trong hệ thống phân tán bao gồm: (1) lỗi liên kết mạng, (2) lỗi máy chủ, (3) lỗi phương tiện lưu trữ. Cả (2) và (3) đều là lỗi cũng có thể xảy ra trong hệ thống tập trung, trong khi lỗi liên kết mạng chỉ có thể xảy ra trong hệ thống được phân phối theo mạng.

16.9. Có phải luôn luôn quan trọng để biết rằng tin nhắn bạn gửi đã đến không tại điểm đến của nó một cách an toàn? Nếu câu trả lời của bạn là có, hãy giải thích tại sao. Nếu là của bạn câu trả lời là không, cho ví dụ thích hợp.

Trả lời: Không. Nhiều chương trình thu thập trạng thái hoạt động dựa trên giả định rằng hệ thống đích có thể không nhận được các gói. Này các chương trình thường phát một gói tin và giả định rằng ít nhất một số các hệ thống khác trên mạng của họ sẽ nhận được thông tin. Ví dụ: một daemon trên mỗi hệ thống có thể phát tải hệ thống trung bình và số lượng người dùng. Thông tin này có thể được sử dụng để lựa chọn mục tiêu di chuyển quy trình. Một ví dụ khác là một chương trình xác định xem một trang web từ xa có đang chạy và có thể truy cập được qua mạng hay không. Nếu nó gửi một truy vấn và không nhận được trả lời, nó biết rằng hệ thống không thể hiện đang đạt được.

16.10. Trình bày một thuật toán để tái tạo lại một vòng logic sau một quá trình trong chiếc nhẫn không thành công.

Trả lời: Các hệ thống phân tán thường sử dụng quy trình điều phối viên thực hiện các chức năng cần thiết của các quá trình khác trong hệ thống. Cái này sẽ bao gồm việc thực thi loại trừ lẫn nhau và — trong trường hợp này là một chiếc nhẫn—Đặt lại mã thông báo bị mất. Một lược đồ tương tự như thuật toán vòng được trình bày trong Phần 18.6.2 có thể được dùng. Thuật toán như sau:

Thuật toán vòng giả định rằng các liên kết là đơn hướng và các quy trình gửi tin nhắn của họ đến người hàng xóm ở bên phải của họ. Chính Cấu trúc dữ liệu được sử dụng bởi thuật toán là danh sách hoạt động, một danh sách chứa số ưu tiên của tất cả các quy trình đang hoạt động trong hệ thống khi thuật toán kết thúc; mỗi quy trình duy trì danh sách hoạt động của riêng nó.

a. Nếu quá trình Pi phát hiện lỗi bộ điều phối, nó sẽ tạo ra một hoạt động mới danh sách ban đầu trống. Sau đó, nó sẽ gửi một tin nhắn chọn (i) cho nó hàng xóm bên phải và thêm số i vào danh sách hoạt động của nó

b. Nếu Pi nhận được thông báo chọn (j) từ quy trình bên trái, nó phải trả lời theo một trong ba cách:

1. Nếu đây là tin nhắn được chọn đầu tiên mà nó nhìn thấy hoặc gửi đi, Pi sẽ tạo một danh sách hoạt động mới với các số i và j. Sau đó, nó sẽ gửi thông báo chọn (i), tiếp theo là thông báo bầu chọn (j).

2. Nếu i = j, tức là tin nhắn nhận được không chứa Pi 's số, sau đó Pi thêm j vào danh sách hoạt động của nó và chuyển tiếp nhắn tin cho người hàng xóm bên phải của nó.

3. Nếu i = j, tức là Pi nhận được thông báo Elect (i), thì danh sách hoạt động cho Pi hiện chứa số lượng của tất cả các các quy trình trong hệ thống. Process Pi hiện có thể xác định số lượng lớn nhất trong danh sách hoạt động để xác định quy trình điều phối viên.

16.11. Xem xét một hệ thống phân tán có hai địa điểm, A và B. trang web A có thể phân biệt những điều sau:

a. B đi xuống.

b. Liên kết giữa A và B đi xuống.

c. B cực kỳ quá tải và thời gian phản hồi của nó là 100 lần lâu hơn bình thường.

Câu trả lời của bạn có ý nghĩa gì đối với việc khôi phục trong phân phối hệ thống?

Trả lời: Một kỹ thuật sẽ dành cho B để gửi I-up định kỳ thông báo cho A cho biết nó vẫn còn sống. Nếu A không nhận được thông báo I-amup, nó có thể cho rằng B — hoặc liên kết mạng — bị ngắt.

Lưu ý rằng thông báo I-up không cho phép A phân biệt giữa từng loại hư hỏng. Một kỹ thuật cho phép A xác định tốt hơn nếu mạng không hoạt động, hãy gửi một tin nhắn Có sẵn cho B bằng cách sử dụng tuyến luân phiên. Nếu nó nhận được phản hồi, nó có thể xác định rằng thực sự liên kết mạng không hoạt động và mạng B vẫn hoạt động. Nếu chúng ta giả sử rằng A biết B đang hoạt động và có thể truy cập được (thông qua cơ chế I-amup) và A có một số giá trị N cho biết thời gian phản hồi, A có thể theo dõi thời gian phản hồi từ B và so sánh giá trị đến N, cho

phép A xác định xem B có bị quá tải hay không. Ý nghĩa của cả hai kỹ thuật này là A có thể chọn một máy chủ khác — giả sử C — trong hệ thống nếu B bị lỗi, không thể truy cập được, hoặc quá tải.

Chương 21 : Hệ điều hành Linux - Hệ thống

21.1. Các mô-đun nhân có thể tải động mang lại sự linh hoạt khi trình điều khiển được thêm vào một hệ thống, nhưng chúng cũng có nhược điểm? Theo những gì các trường hợp một nhân sẽ được biên dịch thành một tệp nhị phân duy nhất, và Khi nào thì tốt hơn nếu giữ nó được chia thành các mô-đun? Giải thích của bạn trả lời.

Trả lời:

Có hai hạn chế chính khi sử dụng các mô-đun. Đầu tiên là kích thước: quản lý mô-đun sử dụng nhân không thể gắn thẻ bộ nhớ và một hạt nhân cơ bản với một số mô-đun được tải sẽ ngốn nhiều bộ nhớ hơn một nhân tương đương với các trình điều khiển được biên dịch vào chính hình ảnh hạt nhân. Đây có thể là một điều rất quan trọng vấn đề trên máy có bộ nhớ vật lý hạn chế. Hạn chế thứ hai là các mô-đun có thể làm tăng độ phức tạp của quá trình bootstrap hạt nhân. Thật khó để tải một tập hợp các mô-đun từ đĩa nếu trình điều khiển cần thiết để truy cập vào đĩa đó, một mô-đun cần được tải. Do đó, quản lý kernel bootstrap bằng mô-đun có thể yêu cầu công việc bổ sung từ phía quản trị viên: các mô-đun cần thiết để bootstrap cần được đặt vào một hình ảnh đĩa ram được tải cùng với hình ảnh hạt nhân ban đầu khi hệ thống khởi tạo. Trong một số trường hợp nhất định, tốt hơn là sử dụng hạt nhân mô-đun và trong các trường hợp khác trường hợp tốt hơn là sử dụng một nhân với các trình điều khiển thiết bị của nó được liên kết trước. Ở đâu tối thiểu hóa kích thước của nhân là quan trọng, sự lựa chọn sẽ phụ thuộc về tần suất sử dụng các trình điều khiển thiết bị khác nhau. Nếu chúng được sử dụng liên tục, thì các mô-đun không phù hợp. Điều này đặc biệt đúng khi trình điều khiển cần thiết cho chính quá trình khởi động. Mặt khác, nếu một số trình điều khiển không phải lúc nào cũng cần thiết, sau đó cơ chế mô-đun al thấp những trình điều khiển đó được tải và dỡ hàng theo yêu cầu, có khả năng tiết kiệm ròng trong bộ nhớ vật lý. Nơi mà một nhân được xây dựng phải có thể sử dụng được trên nhiều loại của các máy rất khác nhau, thì việc xây dựng nó bằng các mô-đun rõ ràng là thích sử dụng một nhân đơn với hàng tá trình điều khiển không cần thiết ngốn bộ nhớ. Điều này đặc biệt xảy ra đối với các hạt nhân được phân phối thương mại, nơi hỗ trợ nhiều loại phần cứng nhất trong cách đơn giản nhất có thể là một ưu tiên. Tuy nhiên, nếu

một nhân đang được xây dựng cho một máy duy nhất có cấu hình được biết trước, sau đó biên dịch và sử dụng các mô-đun có thể đơn giản là một sự phức tạp không cần thiết. Trong những trường hợp như thế này, việc sử dụng mô-đun cũng có thể là một vấn đề của thị hiếu.

21.2. Đa luồng là một kỹ thuật lập trình thường được sử dụng. Diễn tả ba cách khác nhau mà các luồng có thể được thực hiện. Giải thích như thế nào những cách này so với cơ chế sao chép Linux. Khi nào mỗi cơ chế thay thế tốt hơn hay tệ hơn so với việc sử dụng các bản sao?

Trả lời:

Triển khai chủ đề có thể được phân loại rộng rãi thành hai nhóm: luồng dựa trên nhân và luồng chế độ người dùng. Chủ đề chế độ người dùng các gói dựa vào một số hỗ trợ hạt nhân - ví dụ như chúng có thể yêu cầu các phương tiện ngắt bộ định thời - nhưng việc lập lịch giữa các luồng thì không được thực hiện bởi hạt nhân nhưng bởi một số thư viện mã chế độ người dùng. Nhiều luồng trong quá trình triển khai như vậy sẽ xuất hiện trên hệ điều hành như một ngữ cảnh thực thi duy nhất. Khi quá trình đa luồng đang chạy, nó tự quyết định luồng nào trong số các luồng của nó sẽ thực thi, bằng cách sử dụng nhảy để chuyển đổi giữa các chủ đề theo ưu tiên riêng của nó hoặc quy tắc lập lịch không ưu tiên. Ngoài ra, hạt nhân của hệ điều hành có thể cung cấp hỗ trợ cho chủ đề chính nó. Trong trường hợp này, các chuỗi có thể được triển khai riêng biệt các quy trình xảy ra để chia sẻ một địa chỉ chung hoàn chỉnh hoặc một phần không gian hoặc chúng có thể được triển khai dưới dạng các ngữ cảnh thực thi riêng biệt trong một quy trình duy nhất. Cho dù các chủ đề được tổ chức theo cách nào, chúng xuất hiện dưới dạng ngữ cảnh thực thi hoàn toàn độc lập với ứng dụng. Việc triển khai kết hợp cũng có thể thực hiện được, trong đó một số lượng lớn các chủ đề được cung cấp cho ứng dụng bằng cách sử dụng một số lượng nhỏ hơn của các luồng nhân. Chủ đề người dùng có thể chạy được chạy bởi người có sẵn đầu tiên

nhân chủ đề. Trong Linux, các luồng được thực thi bên trong hạt nhân bởi một bản sao cơ chế tạo một quy trình mới trong cùng một địa chỉ ảo không gian như là tiến trình mẹ. Không giống như một số gói luồng dựa trên nhân, nhân Linux không phân biệt giữa các luồng và quy trình: một luồng chỉ đơn giản là một quy trình không tạo ra một ảo mới không gian địa chỉ khi nó được khởi tạo. Ưu điểm chính của việc triển khai các luồng trong nhân thay vì hơn trong thư viện chế độ người dùng là:

• Hệ thống luồng nhân có thể tận dụng nhiều bộ xử lý nếu chúng có sẵn; và

• Nếu một luồng chặn trong quy trình dịch vụ hạt nhân (ví dụ: cuộc gọi hệ thống hoặc lỗi trang), các chủ đề khác vẫn có thể chạy.

Một lợi thế ít hơn là khả năng gán các thuộc tính bảo mật khác nhau đến từng chủ đề. Việc triển khai chế độ người dùng không có những ưu điểm này. Tại vì các triển khai như vậy chạy hoàn toàn trong một ngữ cảnh thực thi hạt nhân duy nhất, chỉ một luồng có thể chạy cùng một lúc, ngay cả khi nhiều CPU có sẵn. Vì lý do tương tự, nếu một chuỗi tham gia lệnh gọi hệ thống, không có luồng nào khác có thể chạy cho đến khi lệnh gọi hệ thống đó hoàn tất. Kết quả là, một chuỗi thực hiện đọc đĩa chặn sẽ giữ mọi chuỗi trong ứng dụng. Tuy nhiên, triển khai chế độ người dùng có thuận lợi. Rõ ràng nhất là hiệu suất: gọi hạt nhân của bộ lập lịch riêng để chuyển đổi giữa các luồng liên quan đến việc nhập miền bảo vệ mới khi CPU chuyển sang chế độ hạt nhân, trong khi chuyển giữa các luồng trong chế độ người dùng có thể đạt được đơn giản bằng cách lưu và khôi phục các thanh ghi CPU chính. Các luồng chế độ người dùng cũng có thể tiêu tốn ít bộ nhớ hệ thống hơn: hầu hết các hệ thống UNIX sẽ dự trữ ít nhất một trang đầy đủ cho ngăn xếp nhân cho mỗi luồng nhân và ngăn xếp này có thể không thể phân trang được. Phương pháp kết hợp, triển khai nhiều luồng người dùng qua một số lượng luồng nhân nhỏ hơn, cho phép đạt được sự cân bằng giữa các sự cân bằng này. Các luồng nhân sẽ cho phép nhiều luồng để chặn các cuộc gọi nhân cùng một lúc và sẽ cho phép chạy trên nhiều CPU và chuyển đổi luồng chế độ người dùng có thể xảy ra trong mỗi luồng nhân để thực hiện phân luồng nhẹ mà không có chi phí có quá nhiều luồng nhân. Nhược điểm của phương pháp này là sự phức tạp: việc trao quyền kiểm soát sự cân bằng sẽ làm phức tạp chuỗi giao diện người dùng của thư viện.

21.3. Nhân Linux không cho phép phân trang ngoài bộ nhớ nhân. Gì hạn chế này có ảnh hưởng gì đến thiết kế của nhân không? Hai là gì ưu nhược điểm của quyết định thiết kế này?

Trả lời:

Tác động chính của việc không cho phép phân trang bộ nhớ hạt nhân trong Linux là tính không ưu tiên của hạt nhân được bảo toàn. Bất cứ xử lý lỗi trang, cho dù là trong nhân hay ở chế độ người dùng, rủi ro được lên lịch lại trong khi dữ liệu cần thiết được phân trang từ đĩa. Bởi vì hạt nhân có thể dựa vào việc không được lên lịch lại trong quá trình truy cập vào cấu trúc dữ liệu chính, yêu cầu khóa để bảo vệ tính toàn vẹn của những cấu trúc dữ liệu đó được đơn giản hóa rất nhiều. Mặc dù

thiết kế bản thân sự đơn giản là một lợi ích, nó cũng cung cấp một lợi thế hiệu suất quan trọng trên các máy đơn xử lý do thực tế là không cần thiết phải thực hiện khóa bổ sung trên hầu hết các cấu trúc dữ liệu nội bộ. Có một số nhược điểm đối với việc thiếu nhân có thể phân trang bộ nhớ, tuy nhiên. Trước hết, nó áp đặt các ràng buộc về số lượng bộ nhớ mà hạt nhân có thể sử dụng. Không hợp lý để giữ rất lớn cấu trúc dữ liệu trong bộ nhớ không thể phân trang, vì nó đại diện cho bộ nhớ hoàn toàn không thể được sử dụng cho bất cứ điều gì khác. Cái này có hai tác động: trước hết, hạt nhân phải cắt bỏ nhiều dữ liệu bên trong của nó cấu trúc theo cách thủ công, thay vì có thể dựa vào một cơ chế bộ nhớ ảo duy nhất để kiểm soát việc sử dụng bộ nhớ vật lý. Thứ hai, nó làm cho việc triển khai một số tính năng yêu cầu một lượng lớn bộ nhớ ảo trong nhân, chẳng hạn như / tmp-hệ thống tệp (một hệ thống tệp dựa trên bộ nhớ ảo nhanh được tìm thấy trên một số Hệ thống UNIX). Lưu ý rằng sự phức tạp của việc quản lý các lỗi trang trong khi chạy mã nhân không phải là một vấn đề ở đây. Mã nhân Linux đã có thể để đối phó với các lỗi của trang: nó cần có khả năng đối phó với các lệnh gọi hệ thống có các đối số tham chiếu đến bộ nhớ người dùng có thể được phân trang tới đĩa.

21.4. Ba ưu điểm của liên kết động (dùng chung) của các thư viện là gì so với liên kết tĩnh? Hai trường hợp liên kết tĩnh là gì thích hơn?

Trả lời:

Ưu điểm chính của thư viện chia sẻ là chúng giảm bộ nhớ và không gian đĩa được sử dụng bởi một hệ thống và chúng tăng cường khả năng bảo trì. Khi các thư viện được chia sẻ đang được sử dụng bởi tất cả các chương trình đang chạy, chỉ có một phiên bản của mỗi quy trình thư viện hệ thống trên đĩa và nhiều nhất một trường hợp trong bộ nhớ vật lý. Khi thư viện được đề cập là một trong những ứng dụng và chương trình được sử dụng, sau đó là đĩa và tiết kiệm bộ nhớ có thể khá đáng kể. Ngoài ra, thời gian khởi động để chạy các chương trình mới có thể được giảm bớt, vì nhiều các chức năng cần thiết của chương trình đó có thể đã được tải vào bộ nhớ vật lý. Khả năng duy trì cũng là một lợi thế chính của liên kết động tĩnh. Nếu tất cả các chương trình đang chạy sử dụng thư viện được chia sẻ để truy cập các quy trình thư viện hệ thống của chúng, thì hãy nâng cấp các quy trình đó, hoặc để thêm mới chức năng hoặc để sửa lỗi, có thể được thực hiện đơn giản bằng cách thay thế thư viện. Không cần biên dịch lại hoặc liên kết lại bất kỳ ứng dụng nào; bất cứ các chương trình được tải sau khi nâng cấp hoàn tất sẽ tự động chọn lên các phiên bản mới của thư viện. Có những lợi thế khác nữa. Một chương trình sử dụng các thư

viện được chia sẻ thường có thể được điều chỉnh cho các mục đích cụ thể chỉ đơn giản bằng cách thay thế một hoặc nhiều thư viện hơn hoặc thậm chí (nếu hệ thống cho phép và hầu hết các UNIX kể cả Linux do) thêm một cái mới vào lúc chạy. Ví dụ, một thư viện gỡ lỗi có thể được thay thế cho thư viện bình thường để theo dõi sự cố trong một ứng dụng. Thư viện được chia sẻ cũng cho phép các chương trình nhị phân được liên kết với mã thư viện thương mại, độc quyền mà không thực sự bao gồm bất kỳ mã nào trong số đó trong tệp thực thi cuối cùng của chương trình. Đây là quan trọng vì trên hầu hết các hệ thống UNIX, nhiều tiêu chuẩn được chia sẻ thư viện là sở hữu độc quyền và các vấn đề cấp phép có thể ngăn cản bao gồm mã đó trong các tệp thực thi để được phân phối cho các bên thứ ba. Tuy nhiên, ở một số nơi, liên kết tĩnh là thích hợp. Một ví dụ là trong môi trường cứu hộ dành cho quản trị viên hệ thống. Nếu quản trị viên hệ thống mắc lỗi khi cài đặt bất kỳ thư viện mới nào hoặc nếu phần cứng phát triển các vấn đề, hoàn toàn có thể xảy ra đối với các thư viện chia sẻ hiện có trở nên đồi bại. Do đó, thường một tập hợp các tiện ích cứu hộ cơ bản là được liên kết tĩnh để có cơ hội sửa lỗi mà không cần phải dựa vào các thư viện được chia sẻ hoạt động chính xác. Ngoài ra còn có những lợi thế về hiệu suất mà đôi khi làm cho tĩnh liên kết ưu tiên trong các trường hợp đặc biệt. Để bắt đầu, liên kết động không tăng thời gian khởi động cho một chương trình, vì liên kết bây giờ phải được thực hiện tại thời điểm chạy hơn là tại thời điểm biên dịch. Liên kết động cũng có thể đôi khi tăng kích thước bộ làm việc tối đa của một chương trình (tổng số trang vật lý của bộ nhớ cần thiết để chạy chương trình). Trong thư viện được chia sẻ, người dùng không có quyền kiểm soát vị trí trong thư viện tệp nhị phân chứa các chức năng khác nhau. Vì hầu hết các chức năng không điền chính xác vào một trang đầy đủ hoặc các trang của thư viện, việc tải một hàm sẽ thường dẫn đến việc tải các phần của các chức năng xung quanh. liên kết tĩnh, hoàn toàn không có chức năng nào không được tham chiếu (trực tiếp hoặc gián tiếp) bởi ứng dụng cần được tải vào bộ nhớ. Các vấn đề khác xung quanh liên kết tĩnh bao gồm tính dễ phân phối: việc phân phối tệp thực thi có liên kết tĩnh dễ dàng hơn so với liên kết động nếu nhà phân phối không chắc chắn liệu người nhận sẽ có các thư viện chính xác được cài đặt trước. Cũng có thể có các hạn chế thương mại đối với việc phân phối lại một số tệp nhị phân như được chia sẻ các thư viện. Ví dụ: giấy phép cho môi trường đồ họa UNIX “Motif” cho phép các tệp nhị phân sử dụng Motif được phân phối tự do miễn là vì chúng được liên kết tĩnh, nhưng các thư viện được chia sẻ có thể không được sử dụng mà không có giấy phép.

21.5. So sánh việc sử dụng các ổ cắm mạng với việc sử dụng bộ nhớ dùng chung như một cơ chế để giao tiếp dữ liệu giữa các quy trình trên một máy tính. Ưu điểm của từng phương pháp là gì? Khi nào mỗi được ưa thích?

Trả lời:

Sử dụng ổ cắm mạng thay vì bộ nhớ dùng chung cho cục bộ giao tiếp có một số lợi thế. Ưu điểm chính là rằng giao diện lập trình socket có một tập hợp các tính năng đồng bộ hóa phong phú. Một quá trình có thể dễ dàng xác định khi nào dữ liệu mới có đã đến trên kết nối socket, lượng dữ liệu hiện có và ai đã gửi nó. Các quy trình có thể chặn cho đến khi dữ liệu mới đến trên một ổ cắm hoặc chúng có thể yêu cầu gửi tín hiệu khi dữ liệu đến. Một ổ cắm cũng quản lý các kết nối riêng biệt. Một quy trình với một ổ cắm mở để nhận có thể chấp nhận nhiều kết nối đến ổ cắm đó và sẽ được thông báo khi các quy trình mới cố gắng kết nối hoặc khi các quy trình cũ giảm kết nối. Bộ nhớ dùng chung không cung cấp các tính năng này. Không có cách nào cho một quy trình để xác định xem một quy trình khác đã phân phối hoặc đã thay đổi dữ liệu trong bộ nhớ dùng chung ngoài việc xem xét nội dung của bộ nhớ đó. Không thể có quy trình nào chặn và yêu cầu đánh thức khi bộ nhớ dùng chung được phân phối và không có cơ chế tiêu chuẩn cho các quy trình khác để thiết lập bộ nhớ dùng chung liên kết đến một quy trình hiện có. Tuy nhiên, bộ nhớ dùng chung có ưu điểm là nó rất nhiều nhanh hơn so với giao tiếp socket trong nhiều trường hợp. Khi dữ liệu được gửi đi qua một socket, nó thường được sao chép từ bộ nhớ sang nhiều bộ nhớ lần. Cập nhật bộ nhớ dùng chung không yêu cầu bản sao dữ liệu: nếu một quá trình cập nhật cấu trúc dữ liệu trong bộ nhớ dùng chung, cập nhật đó ngay lập tức hiển thị cho tất cả các tiến trình khác chia sẻ bộ nhớ đó. Gửi hoặc nhận dữ liệu qua ổ cắm yêu cầu thực hiện lệnh gọi dịch vụ hệ thống hạt nhân để bắt đầu chuyển giao, nhưng giao tiếp bộ nhớ dùng chung có thể được thực hiện hoàn toàn ở chế độ người dùng mà không cần chuyển quyền điều khiển. Giao tiếp ổ cắm thường được ưu tiên khi quản lý kết nối là quan trọng hoặc khi có yêu cầu đồng bộ hóa người gửi và người nhận. Ví dụ: các quy trình máy chủ thường sẽ thiết lập một ổ cắm lắng nghe mà máy khách có thể kết nối khi họ muốn để sử dụng dịch vụ đó. Sau khi ổ cắm được thiết lập, các yêu cầu riêng lẻ cũng được gửi bằng cách sử dụng socket để máy chủ có thể dễ dàng xác định khi nào một yêu cầu mới đến và nó đến từ ai. Tuy nhiên, trong một số trường hợp, bộ nhớ dùng chung được ưu tiên hơn. Bộ nhớ dùng chung thường là một giải pháp tốt hơn khi một trong hai lượng lớn dữ liệu phải được chuyển giao hoặc khi hai quy trình cần truy cập ngẫu nhiên vào một tập dữ liệu chung. Tuy nhiên, trong trường hợp này, các quá trình giao tiếp vẫn có thể cần một cơ chế bổ sung ngoài bộ nhớ dùng

chung để đạt được sự đồng bộ giữa chúng. Hệ thống cửa sổ X, a môi trường hiển thị đồ họa cho UNIX, là một ví dụ điển hình về điều này: hầu hết các yêu cầu đồ họa được gửi qua các ổ cắm, nhưng bộ nhớ dùng chung được cung cấp như một phương tiện vận chuyển bổ sung trong các trường hợp đặc biệt, nơi các bitmap lớn phải được hiển thị trên màn hình. Trong trường hợp này, một yêu cầu hiển thị bitmap sẽ vẫn được gửi qua socket, nhưng dữ liệu lớn của chính bitmap sẽ được gửi qua bộ nhớ dùng chung.

21.6. Hệ thống UNIX được sử dụng để sử dụng tối ưu hóa bố cục đĩa dựa trên vị trí xoay của dữ liệu đĩa, nhưng các triển khai hiện đại, bao gồm Linux, chỉ cần tối ưu hóa để truy cập dữ liệu tuần tự. Tại sao họ làm như vậy? Truy cập tuần tự tận dụng những đặc điểm phần cứng nào? Tại sao tối ưu hóa xoay vòng không còn hữu ích nữa?

Trả lời:

Các đặc tính hiệu suất của phần cứng đĩa có đã thay đổi đáng kể trong những năm gần đây. Đặc biệt, nhiều cải tiến đã được giới thiệu để tăng băng thông tối đa giúp có thể đạt được trên đĩa. Trong một hệ thống hiện đại, có thể có một đường dẫn giữa hệ điều hành và đầu đọc-ghi của đĩa. Yêu cầu I / O đĩa phải chuyển qua bộ điều khiển đĩa cục bộ của máy tính, qua buslogic đến chính ổ đĩa và sau đó là nội bộ đĩa, nơi có khả năng có một bộ điều khiển phức tạp có thể lưu dữ liệu vào bộ nhớ cache truy cập và có khả năng tối ưu hóa thứ tự của các yêu cầu I / O. Do sự phức tạp này, thời gian cần thiết cho một yêu cầu I / O là thừa nhận và yêu cầu tiếp theo được tạo và nhận bởi đĩa có thể vượt xa khoảng thời gian giữa một khu vực đĩa đi qua phần đầu đọc-ghi và phần đầu khu vực tiếp theo đến Để có thể đọc nhiều sector cùng một lúc một cách hiệu quả, các đĩa sẽ sử dụng một bộ nhớ cache trên đầu đọc. Trong khi một khu vực đang được chuyển trở lại đến máy tính chủ, đĩa sẽ bận đọc các thành phần tiếp theo trong dự đoán yêu cầu đọc chúng. Nếu yêu cầu đọc bắt đầu đến một đơn đặt hàng phá vỡ đường ống dẫn đầu đọc này, hiệu suất sẽ giảm Do đó, hiệu suất mang lại lợi ích đáng kể nếu hệ điều hành cố gắng giữ cho các yêu cầu I / O theo thứ tự tuần tự nghiêm ngặt. Đặc điểm thứ hai của đĩa hiện đại là hình học của chúng có thể rất phức tạp. Số lượng các cung trên mỗi xi lanh có thể thay đổi tùy theo vị trí của hình trụ: càng nhiều dữ liệu có thể được ép vào càng lâu các rãnh gần mép đĩa hơn ở tâm đĩa. Cho một hệ điều hành để tối ưu hóa vị trí luân chuyển của dữ liệu trên đĩa, nó sẽ phải có hiểu biết đầy đủ về hình học này, cũng như đặc điểm thời gian của đĩa và bộ điều khiển của nó. Nói chung, chỉ

logic bên trong của đĩa mới có thể xác định lập lịch tối ưu I / Os và hình dạng của đĩa có khả năng đánh bại bất kỳ nỗ lực nào của hệ điều hành để thực hiện tối ưu hóa luân phiên.

Chương 22: Window XP

22.1. Windows XP là loại hệ điều hành nào? Mô tả hai trong số nó các tính năng chính.

Trả lời: Hệ điều hành đa nhiệm 32/64 bit hỗ trợ nhiều người dùng. (1) Khả năng tự động sửa chữa các sự cố ứng dụng và hệ điều hành. (2) Kết nối mạng và thiết bị tốt hơn kinh nghiệm (bao gồm cả nhiếp ảnh kỹ thuật số và video).

22.2. Liệt kê các mục tiêu thiết kế của Windows XP. Mô tả hai chi tiết.

Trả lời: Mục tiêu thiết kế bao gồm bảo mật, độ tin cậy, Windows và POSIX khả năng tương thích ứng dụng, hiệu suất cao, khả năng mở rộng, tính di động và hỗ trợ quốc tế. (1) Độ tin cậy được coi là nghiêm ngặt yêu cầu và bao gồm xác minh trình điều khiển rộng rãi, cơ sở vật chất cho bắt lỗi lập trình trong mã cấp người dùng và quy trình chứng nhận nghiêm ngặt cho trình điều khiển, ứng dụng và thiết bị của bên thứ ba. (2) Để đạt được hiệu suất cao yêu cầu kiểm tra các khu vực vấn đề trước đây như hiệu suất I / O, tắc nghẽn CPU máy chủ và khả năng mở rộng của môi trường đa luồng và đa xử lý.

22.3. Mô tả quá trình khởi động hệ thống Windows XP.

Trả lời: (1) Khi phần cứng bật nguồn, BIOS bắt đầu thực thi từ ROM và tải và thực thi trình tải bootstrap từ đĩa. (2) Chương trình NTLDR được tải từ thư mục gốc của thiết bị hệ thống và xác định thiết bị khởi động nào chứa hệ điều hành hệ thống. (3) NTLDR tải thư viện HAL, hạt nhân và tổ chức hệ thống. Các hệ thống hive chỉ ra các trình điều khiển khởi động cần thiết và tải chúng. (4) Quá trình thực thi hạt nhân bắt đầu bằng cách khởi tạo hệ thống và tạo hai quy trình: quy trình hệ thống chứa tất cả các luồng nhân viên nội bộ, và quy trình khởi tạo chế độ người dùng đầu tiên: SMSS. (5) SMSS hơn nữa khởi tạo hệ thống bằng cách thiết lập tệp hoán trang và tải thiết bị trình điều khiển. (6) SMSS tạo ra hai quy trình: WINLOGON, đưa lên phần còn lại của hệ thống và CSRSS (quy trình hệ thống con Win32).

22.4. Mô tả ba lớp kiến trúc chính của Windows XP.

Trả lời: (1) HAL (Lớp trừu tượng phần cứng) tạo ra tính di động của hệ điều hành bằng cách ẩn đi sự khác biệt phần cứng so với phần trên các lớp của hệ điều hành. Các chi tiết quản trị của các cơ sở cấp thấp được cung cấp bởi các giao diện HAL. HAL giới thiệu một máy ảo giao diện được sử dụng bởi người điều phối hạt nhân, người điều hành và thiết bị trình điều khiển. (2) Lớp nhân cung cấp nền tảng cho việc điều hành chức năng và hệ thống con chế độ người dùng. Kernel vẫn còn trong bộ nhớ và không bao giờ được đánh trước. Các trách nhiệm của nó là lập lịch luồng, xử lý ngắt và ngoại lệ, đồng bộ hóa bộ xử lý cấp thấp, và khôi phục sự cố mất điện. (3) Lớp điều hành cung cấp một tập hợp các dịch vụ được sử dụng bởi tất cả các hệ thống con: trình quản lý đối tượng, trình quản lý bộ nhớ ảo, quản lý quy trình, cơ sở gọi thủ tục cục bộ, quản lý I / O, bảo mật giám sát, trình quản lý plug-and-play, đăng ký và khởi động.

22.5. Công việc của người quản lý đối tượng là gì?

Trả lời: Các đối tượng trình bày một tập hợp chung các giao diện chế độ hạt nhân cho các chương trình mã hóa. Các đối tượng được thao tác bởi đối tượng lớp điều hành người quản lý. Công việc của người quản lý đối tượng là giám sát việc phân bổ và sử dụng tất cả các đối tượng được quản lý.

22.6. Người quản lý quy trình cung cấp những loại dịch vụ nào? Một là gì cuộc gọi thủ tục địa phương?

Trả lời: Trình quản lý quy trình cung cấp các dịch vụ để tạo, xóa và sử dụng các quy trình, luồng và công việc. Người quản lý quy trình cũng triển khai hàng đợi và phân phối các cuộc gọi thủ tục không đồng bộ tới chủ đề. Cuộc gọi thủ tục cục bộ (LPC) là một hệ thống truyền thông điệp. Hệ điều hành sử dụng LPC để chuyển các yêu cầu và kết quả giữa các quy trình máy khách và máy chủ trong một máy duy nhất, cụ thể là giữa các hệ thống con của Windows XP.

22.7. Người quản lý I / O có những trách nhiệm gì?

Trả lời: Trình quản lý I / O chịu trách nhiệm về hệ thống tệp, thiết bị trình điều khiển và trình điều khiển mạng. Người quản lý I / O theo dõi trình điều khiển thiết

bị, trình điều khiển bộ lọc và hệ thống tệp được tải và quản lý bộ đệm cho các yêu cầu I / O. Nó cũng hỗ trợ cung cấp I / O tệp được ánh xạ bộ nhớ và kiểm soát trình quản lý bộ nhớ cache cho toàn bộ I / O hệ thống.

22.8. Windows XP có cung cấp bất kỳ quy trình chế độ người dùng nào cho phép nó chạy không chương trình được phát triển cho các hệ điều hành khác? Mô tả hai trong số này hệ thống con.

Trả lời: Hệ thống con môi trường là các quy trình chế độ người dùng được phân lớp qua các dịch vụ thực thi gốc để cho phép Windows XP chạy các chương trình được phát triển cho các hệ điều hành khác. (1) Một ứng dụng Win 3 được gọi là máy DOS ảo (VDM) được cung cấp như một quy trình chế độ người dùng để chạy các ứng dụng MS-DOS. VDM có thể thực thi hoặc mô phỏng Intel 486 hướng dẫn và cũng cung cấp các thủ tục để mô phỏng MS-DOS BIOS dịch vụ và cung cấp trình điều khiển ảo cho màn hình, bàn phím và các cổng giao tiếp. (2) Windows-on-windows (WOW32) cung cấp nhân và các quy trình sơ khai cho các chức năng của Windows 3.1. Các thói quen sơ khai gọi các chương trình con Win32 thích hợp, chuyển đổi các địa chỉ 16 bit thành Địa chỉ 32-bit.

22.9. Windows XP hỗ trợ những loại mạng nào? Windows XP thực hiện các giao thức truyền tải như thế nào? Mô tả hai giao thức mạng.

Trả lời: Hỗ trợ được cung cấp cho cả máy chủ ngang hàng và máy khách kết nối mạng. Các giao thức vận tải được thực hiện dưới dạng trình điều khiển. (1) Các Gói TCP / IP bao gồm hỗ trợ SNMP, DHCP, WINS và NetBIOS. (2) Giao thức đường hầm điểm-điểm được cung cấp để giao tiếp giữa mô-đun truy cập từ xa chạy trên máy chủ Windows XP và ứng dụng khách khác hệ thống kết nối qua internet. Sử dụng lược đồ này, đa giao thức mạng riêng ảo (VPN) được hỗ trợ qua internet.

22.10. Không gian tên NTFS được tổ chức như thế nào? Diễn tả.

Trả lời: Không gian tên NTFS được tổ chức như một hệ thống phân cấp của các thư mục trong đó mỗi thư mục sử dụng cấu trúc dữ liệu cây B + để lưu trữ một chỉ mục của tên tệp trong thư mục đó. Gốc chỉ mục của một thư mục chứa cấp cao nhất của cây B +. Mỗi mục trong thư mục chứa tên và tệp tham chiếu của tệp cũng như dấu thời gian cập nhật và Kích thước tập tin.

22.11 NTFS xử lý cấu trúc dữ liệu như thế nào? NTFS phục hồi như thế nào một sự cố hệ thống? Điều gì được đảm bảo sau khi quá trình khôi phục diễn ra?

Trả lời: Trong NTFS, tất cả các cập nhật cấu trúc dữ liệu hệ thống tệp đều được thực hiện giao dịch bên trong. Trước khi cấu trúc dữ liệu bị thay đổi, giao dịch ghi một bản ghi nhật ký có chứa thông tin làm lại và hoàn tác. Một cam kết bản ghi được ghi vào nhật ký sau khi giao dịch thành công. Sau một sự cố hệ thống tệp có thể được khôi phục về trạng thái nhất quán bằng cách xử lý các bản ghi nhật ký, các hoạt động thực hiện lại đầu tiên cho các giao dịch đã cam kết và hoàn tác hoạt động cho các giao dịch không được cam kết thành công. Lược đồ này không đảm bảo rằng nội dung tệp người dùng là chính xác sau khi khôi phục, mà thay vào đó là cấu trúc dữ liệu hệ thống tệp (siêu dữ liệu tệp) không bị hư hại và phản ánh một số trạng thái nhất quán tồn tại trước tai nạn.

22.12. Windows XP phân bổ bộ nhớ người dùng như thế nào?

Trả lời: Bộ nhớ người dùng có thể được cấp phát theo một số sơ đồ: bộ nhớ ảo, tệp ánh xạ bộ nhớ, đống và bộ nhớ cục bộ chuỗi.

22.13. Mô tả một số cách ứng dụng có thể sử dụng bộ nhớ thông qua API Win32.

Trả lời: (1) Bộ nhớ ảo cung cấp một số chức năng cho phép một ứng dụng để dự trữ và giải phóng bộ nhớ, chỉ định địa chỉ mà bộ nhớ được cấp phát. (2) Một tệp có thể được ánh xạ bộ nhớ vào không gian địa chỉ, cung cấp phương tiện cho hai quá trình chia sẻ bộ nhớ. (3) Khi một quy trình Win32 được khởi tạo, nó sẽ được tạo với một đống mặc định. Có thể tạo đống riêng để cung cấp cho các khu vực không gian địa chỉ dành riêng cho các ứng dụng. Chức năng quản lý luồng được cung cấp để phân bổ và kiểm soát quyền truy cập luồng vào các heap riêng tư. (4) Cơ chế lưu trữ cục bộ luồng cung cấp một cách thức cho toàn cục và tĩnh dữ liệu để hoạt động bình thường trong môi trường đa luồng. Vặn ốc bộ nhớ phân bổ bộ nhớ toàn cầu trên cơ sở mỗi luồng.