Introduction

1.1 3 mục đích chính của một hệ điều hành là gì?
Trả lời: • Cung cấp một môi trường để người dùng máy tính thực thi các chương trình trên phần cứng máy tính một cách tiện lợi 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 nên là công bằng và hiệu quả nhất có thể. • Là một chương trình điều khiển, nó phục vụ hai chức năng chính: (1) giám sát thực thi các chương trình người dùng để ngăn chặn lỗi và việc sử dụng sai trái của máy tính và (2) quản lý hoạt động và điều khiển các thiết bị I / O

1.2 Những khác biệt chính giữa hệ điều hành cho máy tính lớn và máy tính cá nhân là gì?
Trả lời: Thông thường, hệ điều hành cho các hệ thống batch có yêu cầu đơn giản hơn so với cho máy tính cá nhân. Hệ thống batch không phải quan tâm nhiều đến tương tác với người dùng như một máy tính cá nhân. Kết quả là, một hệ điều hành cho máy tính cá nhân phải quan tâm đến thời gian phản hồi cho người dùng tương tác. Hệ thống batch không có các yêu cầu như vậy. Một hệ thống batch thuần túy cũng có thể không xử lý 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 tính hoàn toàn được dành riêng.
Trả lời:
1. Đặt thời gian máy.
2. Tải chương trình vào bộ nhớ bằng tay.
3. Tải địa chỉ bắt đầu và bắt đầu thực thi.
4. Giám sát và kiểm soát thực thi chương trình từ bảng điều khiển.

1.4 Chúng ta đã nhấn mạnh nhu cầu của một hệ điều hành để sử dụng hiệu quả phần cứng máy tính. Khi nào thì hệ điều hành nên bỏ qua nguyên tắc này và "lãng phí" tài nguyên? Tại sao một hệ thống như vậy không phải là lãng phí?
Trả lời: Hệ thống chỉ dành cho một người dùng nên tối đa hóa việc sử dụng hệ thống cho người dùng. Một giao diện đồ họa có thể "lãng phí" chu kỳ CPU, nhưng nó tối ưu hóa tương tác của người dùng với hệ thống.

1.5 Khó khăn chính mà một nhà lập trình phải vượt qua khi viết một hệ điều hành cho một môi trường thời gian thực là gì?
Trả lời: Khó khăn chính là giữ cho hệ điều hành trong các ràng buộc thời gian cố định của một hệ thống thời gian thực. Nếu hệ thống không hoàn thành một tác vụ trong một khung thời gian nhất định, nó có thể gây ra sự cố cho toàn bộ hệ thống đang chạy. Do đó, khi viết một hệ điều hành cho một hệ thống thời gian thực, người viết phải đảm bảo rằng các kế hoạch lên lịch của họ không cho phép thời gian phản hồi vượt quá ràng buộc thời gian

1.6 Hãy xem xét các định nghĩa khác nhau về hệ điều hành. Hãy xem xét liệu hệ điều hành có nên bao gồm các ứng dụng như trình duyệt web và chương trình thư không? Hãy bình luận về cả hai điều rằng nó nên và không nên, và hỗ trợ câu trả lời của bạn.
Trả lời:
- **Point**: Các ứng dụng như trình duyệt web và các công cụ thư điện tử đang đóng một vai trò ngày càng quan trọng trong các hệ thống máy tính để bàn hiện đại. Để thực hiện vai trò này, chúng nên được tích hợp làm phần của hệ điều hành. Bằng cách làm như vậy, 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 cùng giao diện và cảm giác với phần mềm hệ điều hành.

- **Counterpoint**: Vai trò cơ bản của hệ điều hành là quản lý tài nguyên hệ thống như CPU, bộ nhớ, các thiết bị I / O, v.v. Ngoài ra, nó còn chạy các ứng dụng phần mềm như trình duyệt web và các ứng dụng thư điện tử. Bằng cách tích hợp các ứng dụng như vậy vào hệ điều hành, chúng ta gánh nặng cho hệ điều hành với các chức năng bổ sung. Sự gánh nặng như vậy có thể dẫn đến hệ điều hành không thực hiện được công việc quản lý tài nguyên hệ thống một cách tốt nhất. Ngoài ra, chúng ta tăng kích thước của hệ điều hành, do đó tăng khả năng xảy ra sự cố hệ thống và vi phạm bảo mật.

1.7 Sự phân biệt giữa chế độ nhân (kernel mode) và chế độ người dùng (user mode) hoạt động như một hệ thống bảo vệ (bảo mật) sơ khai như thế nào?
Trả lời: Sự phân biệt giữa chế độ nhân và chế độ người dùng cung cấp 1 hệ thống bảo vệ sơ khai như sau:
- Một số chỉ thị chỉ có thể được thực hiện khi CPU ở chế độ nhân. Tương tự, các thiết bị phần cứng chỉ có thể được truy cập khi chương trình đang thực thi ở chế độ nhân.
- Kiểm soát khi nào ngắt có thể được bật hoặc tắt cũng chỉ có thể thực hiện khi CPU ở chế độ nhân. Do đó, CPU có khả năng rất hạn chế khi thực thi ở chế độ người dùng, từ đó thực thi bảo vệ cho các tài nguyên quan trọng.

1.8 Các chỉ thị nào sau đây nên được thực thi ở chế độ đặc quyền (privileged)?
a. Thiết lập giá trị của bộ định thời gian (timer)
b. Đọc đồng hồ.
c. Xóa bộ nhớ
d. Phát ra chỉ thị trap. (Issue a trap instruction)
e. Tắt ngắt (Turn off interupt)
f. Sửa đổi các mục nhập trong bảng trạng thái thiết bị (Modify entries in device-status table)
g. Chuyển từ chế độ người dùng sang chế độ nhân.
h. Truy cập thiết bị I / O.
Trả lời: Các hoạt động sau đây cần được đặc quyền: Thiết lập giá trị bộ định thời gian, xóa bộ nhớ, tắt ngắt, sửa đổi các mục nhập trong bảng trạng thái thiết bị, truy cập thiết bị I / O. Những hoạt động còn lại có thể được thực hiện ở chế độ người dùng.

1.9, Một số máy tính đời đầu bảo vệ hệ điều hành bằng cách đặt nó trong một phân vùng bộ nhớ không thể được sửa đổi bởi cả tác vụ người dùng hoặc hệ điều hành chính nó. Hãy 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, điều khiển truy cập, thông tin kế toán, v.v.) sẽ phải được **lưu trữ hoặc chuyển qua bộ nhớ không được bảo vệ** và do đó có thể **bị truy cập bởi người dùng không được ủy quyền**

1.10 Một số CPU cung cấp cho nhiều hơn 2 chế độ hoạt động. Hai mục đích sử dụng của các chế độ này là gì?
Trả lời: Mặc dù hầu hết các hệ thống chỉ phân biệt giữa chế độ người dùng và chế độ hạt nhân, một số CPU hỗ trợ nhiều chế độ. Nhiều chế độ có thể được sử dụng để cung cấp một chính sách bảo mật chi tiết hơn. Ví dụ, thay vì phân biệt giữa chỉ có chế độ người dùng và chế độ hạt nhân, bạn có thể phân biệt giữa các loại chế độ người dùng khác nhau. Có thể người dùng thuộc cùng một nhóm có thể thực thi mã của nhau. Máy tính sẽ chuyển sang chế độ được chỉ định khi một trong những người này đang chạy mã. Khi máy tính ở chế độ này, một thành viên của nhóm có thể chạy mã thuộc về bất kỳ ai khác trong nhóm.
Một khả năng khác sẽ là cung cấp các phân biệt khác nhau trong mã hạt nhân. Ví dụ, một chế độ cụ thể có thể cho phép các trình điều khiển thiết bị USB chạy. Điều này có nghĩa là các thiết bị USB có thể được phục vụ mà không cần phải chuyển sang chế độ hạt nhân, do đó cho phép các trình điều khiển thiết bị USB chạy ở chế độ giống như người dùng / hạt nhân.

1.11 Bộ đếm thời gian (Timer) có thể được sử dụng để tính thời gian hiện tại. Hãy cung cấp một mô tả ngắn gọn về cách thực hiện điều này:
Trả lời: Chương trình có thể sử dụng phương pháp sau để tính toán thời gian hiện tại bằng cách sử dụng các ngắt hẹn giờ.:
+ Chương trình đặt 1 bộ đếm thời gian cho một khoảng thời gian trong tương lai và đi vào chế độ ngủ.
+ Khi nó được đánh thức bởi ngắt, nó có thể cập nhật trạng thái cục bộ của mình, mà nó đang sử dụng để theo dõi số lượng ngắt mà nó đã nhận được cho đến nay. Sau đó, nó có thể lặp lại quá trình này của việc liên tục đặt các ngắt hẹn giờ và cập nhật trạng thái cục bộ của mình khi các ngắt được kích hoạt thực sự.

1.12 Internet là một LAN hay một WAN?
Trả lời: Internet là một WAN vì các máy tính khác nhau được đặt tại các địa điểm khác nhau địa lý và được kết nối bằng các liên kết mạng xa.

Operating-System Structures

2.1 Mục đích của các lời gọi hệ thống (system call) là gì?
Trả lời:
Các lời gọi hệ thống cho phép các tiến trình ở mức người dùng yêu cầu các dịch vụ của hệ điều hành.

2.2 Năm hoạt động chính của hệ điều hành trong việc quản lý quy trình là gì?
Trả lời:
a. Tạo và xóa các tiến trình người dùng và hệ thống
b. Tạm ngưng và tiếp tục các tiến trình
c. Cung cấp cơ chế đồng bộ hóa tiến trình
d. Cung cấp cơ chế giao tiếp tiến trình
e. Cung cấp cơ chế xử lý tình trạng khóa

2.3 Ba hoạt động chính của hệ điều hành trong việc quản lý bộ nhớ là gì?
Trả lời:
a. Theo dõi các phần bộ nhớ đang được sử dụng và được sử dụng bởi ai.
b. Quyết định quá trình nào sẽ được tải vào bộ nhớ khi không gian bộ nhớ trở nên có sẵn.
c. Cấp phát và giải phóng không gian bộ nhớ khi cần thiết.

2.4 Ba hoạt động chính của hệ điều hành trong việc quản lý lưu trữ phụ là gì?
Trả lời:
- Quản lý không gian trống.
- Cấp phát và giải phóng bộ nhớ.
- Lập lịch đĩa.

2.5 Mục đích của trình thông dịch lệnh (command interpreter) là gì? Tại sao nó thường là riêng biệt với kernel?
Trả lời:
- Nó đọc các lệnh từ người dùng hoặc từ tệp các lệnh và thực thi chúng, thường bằng cách biến chúng thành một hoặc nhiều lệnh hệ thống. Thông dịch lệnh thường không phải là một phần của kernel vì nó có thể thay đổi.

2.6 Các lệnh hệ thống nào phải được thực thi bởi 1 trình thông dịch lệnh (command interpreter) hoặc shell để bắt đầu một tiến trình mới?
Trả lời:
- Trong các hệ thống Unix, phải thực hiện 1 lệnh hệ thống fork kèm theo 1 lệnh hệ thống exec để bắt đầu 1 tiến trình mới. Lời gọi fork sao chép quá trình đang thực thi hiện tại, trong khi lời gọi exec chồng 1 tiến trình mới dựa trên 1 chương trình thực thi khác đè lên tiến trình đang 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 bản gộp các lệnh hệ thống hữu ích. Chúng cung cấp chức năng cơ bản cho người dùng để người dùng không cần phải viết chương trình riêng của họ để giải quyết các vấn đề thông thường.

2.8 Lợi ích chính của phương pháp thiết kế theo lớp cho hệ thống là gì? Nhược điểm của việc sử dụng phương pháp này là gì?
Trả lời:
- Như trong tất cả các trường hợp của thiết kế module, thiết kế 1 hệ điều hành theo cách module có một số lợi ích:
+ Hệ thống dễ gỡ lỗi và sửa đổi hơn vì các thay đổi chỉ ảnh hưởng đến các phần giới hạn của hệ thống thay vì chạm vào tất cả các phần của hệ điều hành.
+ Thông tin chỉ được giữ ở nơi cần thiết và chỉ có thể truy cập trong 1 khu vực xác định và hạn chế, vì vậy bất kỳ lỗi ảnh hưởng đến dữ liệu đó phải giới hạn trong một module hoặc lớp cụ thể.

- Nhược điểm:
+ Tăng độ phức tạp: Hệ điều hành theo lớp cần phải được thiết kế và triển khai rất cẩn thận để đảm bảo rằng các lớp tương tác với nhau đúng cách. Điều này làm tăng độ phức tạp của hệ thống và có thể dẫn đến các lỗi phức tạp.
+ Hiệu suất giảm: Khi thực hiện các hoạt động, hệ điều hành theo lớp phải chuyển từ lớp này sang lớp khác, và điều này có thể làm giảm hiệu suất của hệ thống.
+ Thiếu tính linh hoạt: Một số hệ điều hành theo lớp có cấu trúc rất cứng nhắc, điều này có thể làm giảm tính linh hoạt của hệ thống và khó khăn trong việc thực hiện các tính năng mới hoặc cập nhật.

2.9 Liệt kê năm dịch vụ được cung cấp bởi một hệ điều hành. Giải thích cách mỗi dịch vụ cung cấp tiện ích cho người dùng. Đồng thời giải thích trong trường hợp nào nó sẽ là không thể cho các chương trình cấp độ người dùng (user-level program) cung cấp các dịch vụ này?
Trả lời:
a. Thực thi chương trình. Hệ điều hành tải nội dung (hoặc các phần) của một tệp vào bộ nhớ và bắt đầu thực thi. Một chương trình cấp người dùng không đáng tin để phân bổ đúng thời gian CPU.

b. Hoạt động I/O. Đĩa, băng, đường truyền nối tiếp (serial lines) và các thiết bị khác phải được giao tiếp ở mức độ rất thấp. Người dùng chỉ cần chỉ định thiết bị và hoạt động để thực hiện trên nó, trong khi hệ thống chuyển đổi yêu cầu đó thành lệnh cụ thể cho thiết bị hoặc điều khiển. Các chương trình cấp người dùng không thể được tin tưởng để truy cập các thiết bị mà họ chỉ có quyền truy cập và truy cập chúng khi chúng không được sử dụng.

c. Thao tác hệ thống tệp. Có rất nhiều chi tiết trong việc tạo, xóa, cấp phát và đặt tên tệp mà người dùng không nên phải thực hiện. Các khối không gian đĩa được sử dụng bởi các tệp và phải được theo dõi. Xóa một tệp yêu cầu xóa thông tin tên tệp và giải phóng các khối đã được cấp phát. Phải kiểm tra các bảo vệ để đảm bảo truy cập tệp đúng. Các chương trình người dùng không thể đảm bảo tuân thủ các phương pháp bảo vệ hoặc được tin tưởng để phân bổ chỉ các khối trống và giải phóng khối khi xóa tệp.

d. Giao tiếp. Truyền thông tin giữa các hệ thống yêu cầu thông điệp được chuyển thành gói thông tin, được gửi đến bộ điều khiển mạng, truyền qua phương tiện truyền thông và được lắp ráp lại bởi hệ thống đích. Phải thực hiện sắp xếp gói và sửa lỗi dữ liệu. Một lần nữa, các chương trình người dùng có thể không đồng bộ hóa truy cập vào thiết bị mạng hoặc nhận các gói dành cho các quy trình khác.

e. Phát hiện lỗi. Phát hiện lỗi xảy ra ở cả cấp phần cứng và phần mềm. Ở cấp phần cứng, tất cả các truyền dữ liệu phải được kiểm tra để đảm bảo dữ liệu không bị hỏng trong quá trình truyền. Tất cả dữ liệu trên phương tiện lưu trữ phải được kiểm tra để đảm bảo chúng không thay đổi kể từ khi được ghi vào phương tiện đó. Ở cấp phần mềm, phải kiểm tra phương tiện lưu trữ để đảm bảo tính nhất quán của dữ liệu; ví dụ, liệu số khối được cấp phát và không được cấp phát có khớp với tổng số trên thiết bị

không. Ở đó, các lỗi thường là độc lập với quá trình (ví dụ, sự hỏng hóc của dữ liệu trên đĩa), vì vậy phải có một chương trình toàn cục (hệ điều hành) xử lý tất cả các loại lỗi. Ngoài ra, bằng cách xử lý các lỗi bằng hệ điều hành, các quy trình không cần chứa mã để bắt lỗi và sửa đổi tất cả các lỗi có thể có trên hệ thống.

2.10 Mục đích của các lệnh hệ thống là gì?
Trả lời: Các lệnh hệ thống cho phép các tiến trình cấp người dùng yêu cầu các dịch vụ của hệ điều hành.

2.11 Lợi ích chính của phương pháp nhân nhỏ trong thiết kế hệ thống là gì?
Trả lời:
Các lợi ích bao gồm
(a) việc thêm một dịch vụ mới không yêu cầu sửa đổi kernel, (b) nó an toàn hơn khi nhiều hoạt động được thực hiện ở chế độ người dùng hơn là ở chế độ kernel, và (c) một thiết kế kernel đơn giản hơn và chức năng 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 firmware, trong khi các hệ thống khác lưu trữ trên đĩa?
Trả lời: Đối với một số thiết bị, chẳng hạn như các PDA cầm tay và điện thoại di động, một đĩa với hệ thống tệp có thể không có sẵn cho thiết bị. Trong tình huống này, hệ điều hành phải được lưu trữ trong firmware.

2.13 Làm thế nào để thiết kế một hệ thống cho phép lựa chọn các hệ điều hành để khởi động? Chương trình khởi động cần làm gì?
Trả lời:
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 gọi là trình quản lý khởi động) sẽ xác định hệ điều hành nào để khởi động. Điều này có nghĩa là thay vì khởi động vào một hệ điều hành, trình quản lý khởi động sẽ chạy trước trong quá trình khởi động hệ thống. Chính trình quản lý khởi động này chịu trách nhiệm xác định hệ thống nào để khởi động. Thông thường, trình quản lý khởi động phải được lưu trữ tại các vị trí nhất định trên ổ cứng để được nhận dạng trong quá trình khởi động hệ thống. Trình quản lý khởi động thường cung cấp cho người dùng một lựa chọn các hệ thống để khởi động; Trình quản lý khởi động cũng thường được thiết kế để khởi động vào một hệ điều hành mặc định nếu người dùng không chọn.

Processes

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

a. Một phương pháp chia sẻ thời gian phải được thực hiện để cho phép mỗi quy trình có thể truy cập vào hệ thống. Phương pháp này liên quan đến việc chặn quy trình không tự nguyện nhường CPU (bằng cách sử dụng một lời gọi hệ thống, ví dụ), và kernel phải được tái nhập (để nhiều hơn một quy trình có thể thực thi mã kernel đồng thời).

b. Quy trình và tài nguyên hệ thống phải được bảo vệ khỏi nhau. Bất kỳ quy trình nào phải bị giới hạn về 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 trọng trong kernel để ngăn chặn tình trạng bế tắc giữa các quy trình, để các quy trình không phải đợi các tài nguyên được phân bổ của nhau.

3.2, Bộ xử lý Sun UltraSPARC có nhiều thanh ghi. Mô tả các hành động của một chuyển đổi ngữ cảnh nếu ngữ cảnh mới đã được tải vào một trong các bộ đăng ký. Điều gì khác phải xảy ra nếu ngữ cảnh mới nằm trong bộ nhớ thay vì trong một bộ đăng ký và tất cả các bộ đăng ký đang được sử dụng?
Trả lời: Các trỏ bộ đăng ký hiện tại của CPU được thay đổi để trỏ vào bộ đăng ký chứa bối cảnh mới, điều này mất rất ít thời gian. Nếu bối cảnh ở trong bộ nhớ, một trong các bối cảnh trong bộ đăng ký phải được chọn và được chuyển đến bộ nhớ, và bối cảnh mới phải được tải từ bộ nhớ vào bộ đăng ký. Quá trình này mất thời gian hơi nhiều hơn trên các hệ thống có một bộ đăng ký, phụ thuộc vào cách lựa chọn nạn nhân thay thế

3.3 Khi một tiến trình tạo một tiến trình mới bằng phép toán fork (), state nào được chia sẻ giữa tiến trình cha và tiến trình con?
a. Stack
b. Heap
c. Các đoạn nhớ chia sẻ

Trả lời: Chỉ có các đoạn nhớ chia sẻ được chia sẻ giữa tiến trình cha và tiến trình con mới được tạo bằng fork (). Bản sao của stack và heap được tạo cho tiến trình được tạo mới.

3.4 Một lần nữa xem xét cơ chế RPC (Remote Procedure Call), xem xét ý nghĩa "đúng một lần". Thuật toán để thực hiện ý nghĩa này có thực hiện đúng không nếu tin nhắn "ACK" trả về cho máy khách bị mất do sự cố mạng? Mô tả chuỗi các thông điệp và xem liệu "đúng một lần" vẫn được bảo toàn.

Trả lời: Ý nghĩa "đúng một lần" đảm bảo rằng một thủ tục từ xa sẽ được thực hiện chính xác một lần và chỉ một lần. Thuật toán chung để đảm bảo điều này kết hợp một phương thức xác nhận (ACK) kết hợp với timestamps (hoặc một bộ đếm tăng tuần tự khác cho phép máy chủ phân biệt giữa các thông điệp trùng lặp).

Chiến lược chung là cho máy khách gửi RPC đến máy chủ cùng với một timestamp. Máy khách cũng sẽ bắt đầu đồng hồ hết giờ. Máy khách sau đó sẽ đợi một trong hai điều kiện: (1) nó sẽ nhận được một ACK từ máy chủ cho biết thủ tục từ xa đã được thực hiện, hoặc (2) nó sẽ hết thời gian. Nếu máy khách hết thời gian, nó cho rằng máy chủ không thể thực hiện thủ tục từ xa vì vậy máy khách gọi RPC lần thứ hai, gửi một timestamp sau đó. Máy khách có thể không nhận được ACK vì một trong hai lý do sau: (1) RPC ban đầu chưa bao giờ được nhận bởi máy chủ, hoặc (2) RPC được nhận đúng - và thực hiện - bởi máy chủ nhưng ACK bị mất. Trong tình huống (1), việc sử dụng ACK cho phép máy chủ cuối cùng nhận và thực hiện RPC. Tr

Threads

4.1 Đưa tra 2 ví dụ lập trình mà trong đó đa luồng cung cấp hiệu năng tốt hơn so với giải pháp đơn luồng
Câu trả lời:
(1) Một máy chủ web phục vụ mỗi yêu cầu trong một luồng riêng biệt.
(2) Một ứng dụng song song như nhân ma trận trong đó các phần khác nhau của ma trận có thể được làm việc song song.
(3) Một chương trình GUI (Giao diện đồ hoạ người dùng) tương tác như một bộ gỡ lỗi nơi một luồng được sử dụng để giám sát đầu vào của người dùng, một luồng khác đại diện cho ứng dụng đang chạy và một luồng thứ ba giám sát hiệu suất.

4.2

CPU Scheduling

5.1, Một thuật toán lập lịch xác định 1 thứ tự cho các execution của các tiến trình được lập lịch của nó. Cho n tiến trình được lập lịch trên 1 processor (bộ xử lý), có bao nhiêu cách để lập lịch cho chúng (hoán vị)

- Trả lời: n!

5.2, Xác định sự khác biệt giữa lập lịch có ưu tiên và lập lịch không có ưu tiên.
Trả lời: Lập lịch có ưu tiên cho phép một tiến trình bị gián đoạn trong khi thực thi, lấy CPU đi và phân bổ nó cho một quy trình khác. Lập lịch không có ưu tiên đảm bảo rằng một tiến trình sẽ nhường quyền điều khiển của CPU chỉ khi nó hoàn tất với một chuỗi thời gian CPU hiện tại (with it current CPU burst)

5.4 Lợi ích của việc có kích thước khác nhau của định mức thời gian trên nhiều mức độ khác nhau của một hệ thống hàng đợi đa cấp?
Trả lời: Các tiến trình mà cần được phục vụ thường xuyên hơn, ví dụ như các tiến trình tương tác như các trình soạn thảo, có thể được đặt trong một hàng đợi với một định mức thời gian nhỏ. Các tiến trình không cần phục vụ thường xuyên có thể được đặt trong một hàng đợi với một định mức thời gian lớn, yêu cầu ít chuyển đổi bối cảnh để hoàn tất xử lý, và do đó tận dụng hiệu quả hơn máy tính.

5.5, Nhiều thuật toán lập lịch CPU được tham số hóa. Ví dụ, Thuật toán điều phối yêu cầu một tham số để chỉ định phần thời gian. Hàng đợi đa cấp thông tin truyền giữa các hàng đợi và các thuật toán lập lịch khác nhau, các tiêu chí được sử dụng để di chuyển các tiến trình giữa các hàng đợi, v.v.
Do đó, các thuật toán này thực sự là các tập hợp các thuật toán (ví dụ, tập hợp các Thuật toán điều phối cho tất cả các phần thời gian, v.v.). Một tập hợp các thuật toán có thể bao gồm một tập hợp khác (ví dụ, thuật toán FCFS (FIRST COME – FIRST SERVE) là Thuật toán điều phối với một lượng thời gian vô hạn). Đâu là mối quan hệ (nếu có) giữa cáccặp thuật toán dưới đây?
1. Ưu tiên và SJF (SHORTEST JOB FIRST)
2. Hàng đợi đa cấp và FCFS (FIRST COME – FIRST SERVE)
3. Ưu tiên và FCFS (FIRST COME – FIRST SERVE)
4. RR và SJF (SHORTEST JOB FIRST)

Trả lời:
1. Công việc ngắn nhất có ưu tiên cao nhất.
2. Cấp độ thấp nhất của MLFQ (MULTILEVEL FEEDBACK QUEUE) là FCFS (FIRST COME – FIRST SERVE).
3. FCFS (FIRST COME – FIRST SERVE) ưu tiên cho công việc đã tồn tại lâu nhất.
4. Không có.

5.6, Giả sử một thuật toán lập lịch (ở cấp độ lập lịch CPU ngắn hạn) ưu tiên những tiến trình đã sử dụng ít nhất thời gian bộ xử lý gần đây. Tại sao thuật toán này ưu tiên các chương trình liên quan đến I/O và không bao giờ đói CPU các chương trình liên quan đến CPU?
Trả lời: Nó sẽ ưu tiên các chương trình liên quan đến I/O do yêu cầu CPU ngắn hơn tương đối; tuy nhiên, các chương trình liên quan đến CPU sẽ không bị thiếu bởi vì các chương trình liên quan đến I/O sẽ đưa ra CPU tương đối thường xuyên để làm I/O của nó.

5.7, Phân biệt giữa lập lịch PCS (PORT CONGESTION SURCHARGE) và SCS (SUPPLY CHAIN SUSTAINABILITY).
Trả lời: Lập lịch PCS (PORT CONGESTION SURCHARGE) được thực hiện cục bộ cho tiến trình. Đó là cách mà thư viện luồng lập lịch luồng vào các TIẾN TRÌNH NHẸ (LIGHT-WEIGHT PROCESS)s có sẵn. Lập lịch SCS (SUPPLY CHAIN SUSTAINABILITY) là trường hợp mà hệ điều hành lập lịch các luồng nhân hệ điều hành. Trên các hệ thống sử dụng mô hình nhiều-đến-một hoặc nhiều-đến-nhiều, hai mô hình lập lịch là hoàn toàn khác nhau. Trên các hệ thống sử dụng mô hình một-đến-một, PCS (PORT CONGESTION SURCHARGE) và SCS (SUPPLY CHAIN SUSTAINABILITY) là giống nhau.

5.8, Giả sử một hệ điều hành ánh xạ các luồng cấp người dùng sang hệ điều hanh nhân bằng mô hình nhiều-đến-nhiều, trong đó ánh xạ được thực hiện thông qua các TIẾN TRÌNH NHẸ (LIGHT-WEIGHT PROCESS). Hơn nữa, hệ thống cho phép các nhà phát triển tạo ra các luồng thời gian thực. Liệu có cần ràng buộc một luồng thời gian thực với một TIẾN TRÌNH NHẸ (LIGHT-WEIGHT PROCESS) không?
Trả lời: Có, nếu không một luồng người dùng có thể phải cạnh tranh để có một TIẾN TRÌNH NHẸ (LIGHT-WEIGHT PROCESS) có sẵn trước khi thực sự được lập lịch. Bằng cách ràng buộc luồng người dùng với một TIẾN TRÌNH NHẸ (LIGHT-WEIGHT PROCESS), không có độ trễ trong khi chờ đợi một TIẾN TRÌNH NHẸ (LIGHT-WEIGHT PROCESS) có

6, Process Synchronization

6.1, Trong Phần 6.4, Chúng tôi đã đề cập rằng việc vô hiệu hóa ngắt thường xuyên có thể ảnh hưởng đến đồng hồ hệ thống. Giải thích tại sao nó có thể và cách làm giảm thiểu ảnh hưởng đó

Đáp án: Đồng hồ hệ thống được cập nhật tại mỗi ngắt của đồng hồ. Nếu các ngắt bị vô hiệu hóa - đặc biệt là trong một khoảng thời gian dài - có thể dẫn đến việc đồng hồ hệ thống mất thời gian chính xác. Đồng hồ hệ thống cũng được sử dụng cho mục đích lập lịch. Ví dụ, lượng thời gian cho một tiến trình được biểu thị dưới dạng số lần đồng hồ tích tắc. Tại mỗi ngắt đồng hồ, bộ lập lịch xác định xem lượng thời gian cho tiến trình đang chạy đã hết hạn hay chưa. Nếu các ngắt đồng hồ bị vô hiệu hóa, bộ lập lịch sẽ không thể gán lượng thời gian chính xác. Hiệu ứng này có thể được giảm thiểu bằng cách vô hiệu hóa các ngắt đồng hồ trong thời gian rất ngắn.

6.2, Vấn đề Người hút thuốc: Xét một hệ thống với ba tiến trình (smoker) người hút thuốc và một tiến trình đại lý (agent). Mỗi người hút thuốc liên tục cuộn một điếu thuốc và sau đó hút nó. Nhưng để cuộn và hút một điếu thuốc, người hút thuốc cần ba nguyên liệu: thuốc lá, giấy và diêm. Một trong các tiến trình người hút thuốc có giấy, một người có thuốc lá và người thứ ba có diêm. Đại lý có cung cấp vô hạn ba vật liệu này. Đại lý đặt hai trong số các nguyên liệu trên bàn. Người hút thuốc nào có nguyên liệu còn lại thì cuộn và hút một điếu thuốc, thông báo cho đại lý khi hoàn thành. Sau đó, đại lý đặt ra hai trong số ba nguyên liệu, và chu trình này tiếp tục lặp lại. Viết một chương trình để đồng bộ hoá đại lý và người hút thuốc bằng đồng bộ hoá Java.

- Trả lời: Hãy đề cập tới trang web hỗ trợ cho mã nguồn của giải pháp
Đáp án: Vui lòng tham khảo trang web hỗ trợ để có giải pháp mã nguồn.

6.3, Đưa ra các lý do tại sao Solaris, Windows XP và Linux thực hiện nhiều cơ chế khóa. Mô tả các hoàn cảnh trong đó chúng sử dụng khóa xoay (spinlocks), bộ chuyển đổi (mutexes), đèn hiệu (semaphores), bộ chuyển đổi thích ứng (adaptive mutexes) và điều kiện biến. Trong mỗi trường hợp, giải thích tại sao cơ chế đó lại cần thiết.

- Trả lời: Những hệ điều hành này cung cấp các cơ chế khóa khác nhau tùy thuộc vào nhu cầu của các nhà phát triển ứng dụng. Khóa xoay (Spinlocks) hữu ích cho hệ thống đa bộ xử lý nơi một luồng có thể chạy trong một vòng lặp bận rộn (trong một khoảng thời gian ngắn) thay vì gây chi phí cho việc đưa nó vào hàng đợi ngủ. Bộ chuyển đổi (Mutexes) hữu ích cho việc khóa tài nguyên. Solaris 2 sử dụng các bộ chuyển đổi thích nghi, có nghĩa là mutex được thực hiện với một spin lock trên các máy đa bộ xử lý. Đèn hiệu (Semaphores) và biến điều kiện là công cụ phù hợp hơn cho đồng bộ hóa khi một tài nguyên phải được giữ trong một khoảng thời gian dài, vì việc quay vòng là không hiệu quả trong một thời gian dài.

6.4, Giải thích sự khác biệt về chi phí giữa ba loại lưu trữ volatile, nonvolatile và stable. (ổn định, không ổn định và bền)
Trả lời: Lưu trữ volatile đề cập đến bộ nhớ chính và bộ nhớ cache và rất nhanh. Tuy nhiên, lưu trữ volatile không thể tồn tại sau các sự cố của hệ thống hoặc khi tắt nguồn hệ thống. Lưu trữ nonvolatile tồn tại sau khi hệ thống bị sự cố và nguồn cung cấp được tắt. Đĩa và băng là các ví dụ về lưu trữ nonvolatile. Gần đây, các thiết bị

USB sử dụng bộ nhớ đọc duy nhất có thể xóa được (EPROM) đã xuất hiện cung cấp lưu trữ nonvolatile. Lưu trữ stable đề cập đến lưu trữ mà kỹ thuật có thể không bao giờ bị mất vì có các bản sao dự phòng dữ liệu (thường trên đĩa).

6.5, Giải thích mục đích của cơ chế kiểm tra điểm kiểm soát. Có bao nhiêu thời gian cần thực hiện kiểm tra điểm kiểm soát? Mô tả cách tần suất kiểm tra điểm kiểm soát ảnh hưởng đến:
- Hiệu suất hệ thống khi không có sự cố
- Thời gian cần để khôi phục từ sự cố hệ thống
- Thời gian cần để khôi phục từ sự cố đĩa

- Trả lời:
+ Bản ghi nhật ký điểm kiểm soát cho biết rằng đã ghi lại một bản ghi nhật ký và dữ liệu được sửa đổi đã được viết vào lưu trữ ổn định và rằng giao dịch không cần phải thực hiện lại trong trường hợp xảy ra sự cố hệ thống. Rõ ràng, càng thường xuyên thực hiện kiểm tra điểm kiểm soát, khả năng cao là không cần phải thực hiện các cập nhật dư thừa trong quá trình khôi phục.

+ Hiệu suất hệ thống khi không có sự cố - Nếu không có sự cố nào xảy ra, hệ thống phải chịu chi phí thực hiện các điểm kiểm soát không cần thiết. Trong tình huống này, thực hiện điểm kiểm soát ít thường xuyên hơn sẽ dẫn đến hiệu suất hệ thống tốt hơn.

+ Thời gian cần để khôi phục từ sự cố hệ thống - Sự tồn tại của một bản ghi nhật ký điểm kiểm soát có nghĩa là một thao tác sẽ không cần phải được thực hiện lại trong quá trình khôi phục hệ thống. Trong tình huống này, thực hiện kiểm tra điểm kiểm soát thường xuyên hơn sẽ làm cho thời gian khôi phục nhanh hơn từ sự cố hệ thống.

+ Thời gian cần để khôi phục từ sự cố đĩa - Sự tồn tại của một bản ghi nhật ký điểm kiểm soát có nghĩa là một thao tác sẽ không cần phải được thực hiện lại trong quá trình khôi phục hệ thống. Trong tình huống này, thực hiện kiểm tra điểm kiểm soát thường xuyên hơn sẽ làm cho thời gian khôi phục nhanh hơn từ sự cố đĩa.

6.6, Giải thích khái niệm về tính nguyên tử của giao dịch.
Trả lời: Một giao dịch là một chuỗi các hoạt động đọc và ghi trên một số dữ liệu, theo sau là một hoạt động cam kết. Nếu chuỗi hoạt động trong một giao dịch không thể hoàn thành, giao dịch phải bị hủy bỏ và các hoạt động đã thực hiện phải được quay trở lại. Quan trọng là chuỗi các hoạt động trong một giao dịch xuất hiện như một hoạt động không thể tách rời để đảm bảo tính toàn vẹn của dữ liệu được cập nhật. Nếu không, dữ liệu có thể bị chiếm đoạt nếu các hoạt động từ hai (hoặc nhiều) giao dịch khác nhau được xen kẽ.

6.7,

6.7 Show that some schedules are possible under the two-phase locking protocol but not possible under the timestamp protocol, and vice versa.
Answer: A schedule that is allowed in the two-phase locking protocol but not in the timestamp protocol is:

step	T_0	T_1	Precedence
1	**lock-S**(A)		
2	**read**(A)		
3		**lock-X**(B)	
4		**write**(B)	
5		**unlock**(B)	
6	**lock-S**(B)		
7	**read**(B)		$T_1 \rightarrow T_0$
8	**unlock**(A)		
9	**unlock**(B)		

This schedule is not allowed in the timestamp protocol because at step 7, the W-timestamp of B is 1.

A schedule that is allowed in the timestamp protocol but not in the two-phase locking protocol is:

step	T_0	T_1	T_2
1	**write**(A)		
2		**write**(A)	
3			**write**(A)
4	**write**(B)		
5		**write**(B)	

This schedule cannot have lock instructions added to make it legal under two-phase locking protocol because T_1 must unlock (A) between steps 2 and 3, and must lock (B) between steps 4 and 5.

6.8, Câu lệnh wait() trong các ví dụ chương trình Java luôn được đặt trong một vòng lặp while. Giải thích tại sao bạn luôn cần phải sử dụng câu lệnh while khi sử dụng wait() và tại sao bạn sẽ không bao giờ sử dụng câu lệnh if.
Đáp án: 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 luồng nhất định rằng một điều kiện nhất định là đúng. Khi một luồng được thông báo, nó là trách nhiệm của nó để kiểm tra lại điều kiện mà nó đang đợi. Nếu một luồng không kiểm tra lại điều kiện, có thể xảy ra tình huống nó nhận được thông báo mà điều kiện đó chưa được đáp ứng.

7. Deadlock (Bế tắc)

7.1, Liệt kê 3 ví dụ của bế tắc mà không liên quan đến môi trường hệ thống
Đáp án:
- Hai chiếc xe đi qua một cây cầu 1 làn từ hai hướng.
- Một người đang bước xuống cầu thang trong khi một người khác đang leo lên cầu thang.
- Hai chuyến tàu di chuyển về phía nhau trên cùng một đường ray.
- Hai thợ mộc phải đóng đinh. Chỉ có một cái búa và một xô đinh. Khi một thợ có cái búa và thợ còn lại có đinh, deadlock xảy ra.

7.2, Giả sử một hệ thống đang ở trạng thái không an toàn. Chứng minh rằng quá trình có thể hoàn thành thực hiện mà không gặp kẹt cứng.
Đáp án: Một trạng thái không an toàn không nhất thiết dẫn đến kẹt cứng, nó chỉ có nghĩa là chúng ta không thể đảm bảo rằng kẹt cứng sẽ không xảy ra. Do đó, có thể xảy ra trường hợp một hệ thống ở trạng thái không an toàn vẫn cho phép tất cả các tiến trình hoàn thành mà không xảy ra kẹt cứng. Xem xét tình huống trong đó một hệ thống có 12 tài nguyên được phân bổ cho các tiến trình P0, P1 và P2. Các nguồn tài nguyên được phân bổ theo chính sách sau:
Tối đa Hiện tại Cần
P0 10 5 5
P1 4 2 2
P2 9 3 6

7.1 Liệt kê ba ví dụ về các tình huống bế tắc(bế tắc) không liên quan đến môi trường hệ thống máy tính.
Đáp án:
- Hai chiếc xe đi qua một cây cầu đơn từ hai hướng.
- Một người đang bước xuống cầu thang trong khi một người khác đang leo lên cầu thang.
- Hai chuyến tàu di chuyển về phía nhau trên cùng một đường ray.
- Hai thợ mộc phải đóng đinh. Chỉ có một cái búa và một xô đinh. Khi một thợ có cái búa và thợ còn lại có đinh, bế tắc xảy ra.

7.2 Giả sử một hệ thống đang ở trạng thái không an toàn. Chứng minh rằng tiến trình có thể hoàn thành thực hiện mà không gặp bế tắc.
Đáp án: Một trạng thái không an toàn không nhất thiết dẫn đến bế tắc, nó chỉ có nghĩa là chúng ta không thể đảm bảo rằng bế tắcsẽ không xảy ra. Do đó, có thể xảy ra trường hợp một hệ thống ở trạng thái không an toàn vẫn cho phép tất cả các tiến trình hoàn thành mà không xảy ra bế tắc. Xem xét tình huống trong đó một hệ thống có 12 tài nguyên được phân bổ cho các tiến trình P0, P1 và P2. Các nguồn tài nguyên được phân bổ theo chính sách sau:
Tối đa Hiện tại Cần
P0 10 5 5
P1 4 2 2
P2 9 3 6

```
for (int i = 0; i < n; i++) {
    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) {
                finish[j] = true;
                for (int x = 0; x < m; x++)
                    work[x] += work[j][x];
            }
        }
    }
}
```

7.3 Chứng minh rằng thuật toán an toàn được trình bày trong Mục 7.5.3 yêu cầu một thứ tự của m × n2 hoạt động.
Đáp án:
Hình 7.1 cung cấp mã Java để thực hiện thuật toán an toàn của ngân hàng viên (thực hiện đầy đủ của ngân hàng viên là có sẵn với việc tải xuống mã nguồn). Như có thể thấy, các vòng lặp nội tuyến lồng nhau - cả hai đều lặp lại n lần - cung cấp hiệu suất của n2. Trong các vòng lặp ngoại cùng này là hai vòng lặp nội tuyến tuần tự lặp lại m lần. Big-oh của thuật toán này là do đó là O(m × n2).
7.4 Xét một hệ thống máy tính chạy 5.000 công việc mỗi tháng mà không có kế hoạch ngăn bế tắchoặc tránh bế tắc. Bế tắcxảy ra khoảng hai lần mỗi tháng, và người điều hành phải kết thúc và chạy lại khoảng 10 công việc mỗi lần bế tắc. Mỗi công việc có giá trị khoảng 2 đô la (trong thời gian CPU), và các công việc bị kết thúc có xu hướng chỉ hoàn thành khoảng một nửa khi chúng bị hủy bỏ. Một lập trình viên hệ thống đã ước tính rằng một thuật toán tránh bế tắc(như thuật toán của ngân hàng viên) có thể được cài đặt vào hệ thống với một sự tăng trung bình về thời gian thực thi mỗi công việc khoảng 10 phần trăm. Vì máy hiện tại có thời gian rảnh 30 phần trăm, tất cả 5.000 công việc mỗi tháng vẫn có thể được thực hiện, mặc dù thời gian xử lý trung bình tăng khoảng 20 phần trăm.
1. Lập luận cho việc cài đặt thuật toán tránh bế tắc?
2. Lập luận chống lại việc cài đặt thuật toán tránh bế tắc?

Đáp án:
Một lập luận cho việc cài đặt tránh bế tắctrong hệ thống là chúng ta có thể đảm bảo bế tắckhông xảy ra. Ngoài ra, mặc dù thời gian xử lý tăng, tất cả 5.000 công việc vẫn có thể chạy.
Một lập luận chống lại việc cài đặt tránh bế tắclà bế tắcxảy ra không thường xuyên và chi phí khi xảy ra rất nhỏ.

7.5 Hệ thống có thể phát hiện được rằng một số tiến trình của nó bị bế tắc không? Nếu bạn trả lời "có", hãy giải thích cách thức phát hiện. Nếu bạn trả lời "không", hãy giải thích hệ thống có thể xử lý vấn đề bế tắcnhư thế nào.
Đáp án:
Bế tắclà một chủ đề khó để định nghĩa vì nó có thể có nghĩa khác nhau đối với các hệ thống khác nhau. Đối với mục đích của câu hỏi này, chúng tôi sẽ định nghĩa bế tắclà tình huống trong đó một tiến trình phải đợi quá lâu - có thể là mãi mãi - trước khi nhận được một tài nguyên được yêu cầu. Một cách để phát hiện bế tắclà xác định trước một khoảng thời gian - T - được coi là không hợp lý. Khi một tiến trình yêu cầu tài nguyên, một bộ đếm thời gian được bắt đầu. Nếu thời gian trôi qua vượt quá T, tiến trình được coi là bị bế tắc.
Một chiến lược để giải quyết tình trạng đói sẽ là áp dụng chính sách
nơi các tài nguyên chỉ được giao cho tiến trình đã đợi lâu nhất. Ví dụ, nếu tiến trình Pa đã đợi lâu hơn cho tài nguyên X so với tiến trình Pb, yêu cầu từ tiến trình Pb sẽ được trì hoãn
cho đến khi yêu cầu của tiến trình Pa được đáp ứng.
Một chiến lược khác sẽ không nghiêm ngặt hơn những gì vừa được đề cập. Trong
kịch bản này, một tài nguyên có thể được cấp cho một tiến trình đã đợi ít hơn một tiến trình khác, miễn là tiến trình khác không bị đói. Tuy nhiên, nếu một tiến trình khác được coi là đói, yêu cầu của nó sẽ được đáp ứng trước.

7.6 Hãy xem xét chính sách phân bổ tài nguyên sau đây. Yêu cầu và giải phóng
cho tài nguyên được cho phép vào bất kỳ thời điểm nào. Nếu yêu cầu
cho tài nguyên không thể được đáp ứng vì tài nguyên không có sẵn, thì chúng tôi kiểm tra bất kỳ tiến trình nào đang bị chặn, đang chờ đợi tài nguyên. Nếu họ có tài nguyên mong muốn, thì tài nguyên này sẽ được lấy từ họ và được cung cấp cho tiến trình yêu cầu. Vector của tài nguyên mà tiến trình đang chờ đợi được tăng lên để bao gồm các tài nguyên đã bị lấy đi.
Ví dụ, xem xét một hệ thống với ba loại tài nguyên và vector Available được khởi tạo là (4,2,2). Nếu tiến trình P0 yêu cầu (2,2,1), nó sẽ nhận được chúng. Nếu P1 yêu cầu (1,0,1), nó sẽ nhận được chúng. Sau đó, nếu P0 yêu cầu (0,0,1), nó bị chặn (không có tài nguyên). Nếu P2 yêu cầu (2,0,0), nó sẽ nhận được tài nguyên có sẵn (1,0,0) và một tài nguyên đã được cấp cho P0 (do P0 bị chặn). Vector Allocation của P0 giảm xuống còn (1,2,1) và vector Need tăng lên còn (1,0,1).
1. Có thể xảy ra tình trạng bế tắc? Nếu bạn trả lời “có”, hãy cho một ví dụ. Nếu bạn
trả lời “không”, chỉ ra điều kiện cần thiết nào không thể xảy ra.
1. Có thể xảy ra tình trạng chặn vô thời hạn? Giải thích câu trả lời của bạn.

Đáp án:
1. Không thể xảy ra tình trạng bế tắc vì có thể giải phóng tài nguyên.
2. Có. Một tiến trình có thể không bao giờ có thể yêu cầu được tất cả các tài nguyên nó cần nếu chúng liên tục bị giải phóng bởi một loạt các yêu cầu như các yêu cầu của tiến trình C.

7.7 Giả sử bạn đã viết mã 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 định nghĩa lại Maxi = Waitingi + Alloca tioni, trong đó Waitingi là vector xác định các tài nguyên tiến trình i đang chờ đợi và Alloca tioni được định nghĩa trong Phần 7.5 không? Giải thích câu trả lời của bạn.
Đáp án:
Có. Vector Max đại diện cho yêu cầu tối đa mà một tiến trình có thể thực hiện. Khi tính toán thuật toán an toàn, chúng ta sử dụng ma trận Need,
đại diện cho Max - Allocation. Một cách khác để nghĩ về điều này là Max
= Need + Allocation. Theo câu hỏi, ma trận Waiting thực hiện một vai trò tương tự như ma trận Need, do đó Max = Waiting + Allocation.
7.8 Có thể xảy ra tình trạng bế tắc liên quan đến chỉ một tiến trình duy nhất không?
Giải thích câu trả lời của bạn.
Đáp án: Không. Điều này trực tiếp xuất phát từ điều kiện giữ và chờ.

8, Memory Management

8.1, Kể tên hai điểm khác biệt giữa địa chỉ luận lý và địa chỉ vật lý.

- Trả lời: Một địa chỉ luận lý không đề cập đến một địa chỉ thực tế hiện có; đúng hơn là, nó đề cập đến một địa chỉ trừu tượng trong một không gian địa chỉ trừu tượng. Tương phản điều này với một địa chỉ vật lý đề cập đến một địa chỉ vật lý thực tế trong bộ nhớ. Một địa chỉ luận lý được tạo bởi CPU và được đơn vị quản lý bộ nhớ (MMU) dịch thành địa chỉ vật lý. Do đó, các 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 (lệnh tìm nạp) hay dữ liệu (tìm nạp hoặc lưu trữ dữ liệu). Do đó, hai cặp thanh ghi giới hạn cơ sở được cung cấp:
+ Một cho hướng dẫn và một cho dữ liệu.

Cặp thanh ghi giới hạn cơ sở hướng dẫn tự động 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 sơ đồ này là nó là một cơ chế hiệu quả để chia sẻ mã và dữ liệu. Ví dụ: chỉ cần lưu một bản sao của trình soạn thảo hoặc trình biên dịch trong bộ nhớ và mã này có thể được chia sẻ bởi tất cả các tiến trình cần quyền truy cập vào trình soạn thảo mã hoặc trình biên dịch mã, Một ưu điểm khác là việc bảo vệ mã chống sửa đổi sai.

+ Nhược điểm duy nhất là mã và dữ liệu phải được tách biệt, điều này thường được tuân thủ trong mã do trình biên dịch tạo.

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 địa chỉ thành một trang và độ lệch. Cách hiệu quả nhất là chia địa chỉ thành các
X bit trang và Y bit độ lệch, thay vì thực hiện phép tính số học trên địa chỉ để tính số trang và độ lệch. Bởi vì vị trí mỗi bit đại diện cho lũy thừa 2, việc chia địa chỉ giữa các bit dẫn đến kích thước trang là lũy thừa 2.

8.4, Xét một không gian địa chỉ luận lý gồm 8 trang, mỗi trang 1024 từ, được ánh xạ vào bộ nhớ vật lý gồm 32 khung.
a, Có bao nhiêu bit trong địa chỉ luận lý?
b, Có bao nhiêu bit trong địa chỉ vật lý?
Trả lời:
a. Địa chỉ luận lý: 13 bit
b. Địa chỉ vật lý: 15 bit

8,5, Tác dụng của việc cho phép hai mục trong bảng trang trỏ đến cùng một khung trang trong bộ nhớ là gì? 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 sang nơi khác. Việc cập nhật một số byte trên một trang sẽ có tác dụng gì trên trang kia?

- Trả lời: Bằng cách cho phép hai mục trong 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 lớn 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 để các bảng trang khác nhau trỏ đến cùng một vị trí bộ nhớ. Tuy nhiên, việc chia sẻ mã hoặc dữ liệu không liên quan có nghĩa là bất kỳ người dùng nào có quyền truy cập vào mã đều có thể sửa đổi mã đó và những sửa đổi này sẽ được phản ánh trong “bản sao” của người dùng khác.

8.6, Mô tả một cơ chế theo đó một phân đoạn có thể thuộc về không gian địa chỉ của hai tiến trình khác nhau.

Trả lời: Vì các bảng phân đoạn là một tập hợp các thanh ghi giới hạn cơ sở, các phân đoạn có thể được chia sẻ khi các mục trong bảng phân đoạn của hai công việc khác nhau trỏ đến cùng một vị trí thực tế. Hai bảng phân đoạn phải có các con trỏ cơ sở giống hệt nhau và số phân đoạn được chia sẻ phải giống
nhau trong hai tiến trình.

8.7, Có thể chia sẻ các phân đoạn giữa các tiến trình mà không yêu cầu cùng một số phân đoạn trong 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 sơ đồ phân trang cho phép chia sẻ các trang mà không yêu cầu số trang phải giống nhau.

- Trả lời: Cả hai vấn đề này dẫn đến việc một chương trình có thể tham chiếu cả mã và dữ liệu của chính nó mà không cần biết phân đoạn hoặc số trang được liên kết với địa chỉ. MULTICS đã giải quyết vấn đề này bằng cách liên kết bốn thanh ghi với mỗi tiến trình. Một thanh ghi có địa chỉ của đoạn chương trình hiện tại, một thanh ghi khác có địa chỉ cơ sở cho ngăn xếp, một thanh ghi khác có địa chỉ cơ sở cho dữ liệu chung, v.v. Ý tưởng là tất cả các tham chiếu phải gián tiếp thông qua một thanh ghi ánh xạ tới 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 số trang hoặc số phân đoạn.

8,8. Trong IBM/370, bảo vệ bộ nhớ được cung cấp thông qua việc sử dụng các khóa. Khóa là một đại lượng 4 bit. Mỗi khối bộ nhớ 2K có một khóa (khóa lưu trữ) được liên kết với nó. CPU cũng có một khóa (khóa bảo vệ) được liên kết với nó. Thao tác lưu trữ chỉ được phép nếu cả hai khóa bằng nhau hoặc nếu một trong hai khóa bằng không. Sơ đồ quản lý bộ nhớ nào sau đây 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 số lượng tiến trình cố định
d. Đa chương trình với số lượng tiến trình thay đổi
đ. Phân trang
f. Phân đoạn

- Trả lời:
a. Bảo vệ không cần thiết, đặt khóa hệ thống thành 0.
b. Đặt phím 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 bằng các khối bộ nhớ.
d. Giống như trên.
đ. Kích thước khung hình phải tăng dần 2k byte, phân bổ khóa theo trang.
f. Kích thước phân đoạn phải tăng dần 2k byte, phân bổ khóa với các phân đoạn.

9. Virtual Memory

9.1, Trong trường hợp nào thì lỗi trang xảy ra? 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 chưa được đưa vào bộ nhớ chính. Hệ điều hành xác minh quyền truy cập bộ nhớ, hủy bỏ chương trình nếu nó không hợp lệ. Nếu nó hợp lệ, một khung trống (free-frame) được định vị và I/O được yêu cầu để đọc trang cần thiết vào khung trống. Sau khi hoàn thành I/O, bảng quy trình và bảng trang được cập nhật và lệnh 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 dàp ***p*** ; ***n*** số trang riêng biệt xảy ra trong đó. Trả lời những câu hỏi dưới đây cho bất kỳ thuật toán thay thế trang nào:
a. Giới hạn dưới của số lỗi trang là bao nhiêu?
b. Giới hạn trên của 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. Một ngăn xếp (Stack)
b. Bảng ký hiệu băm
c. tìm kiếm tuần tự
d. Tìm kiếm nhị phân
đ. mã thuần túy
f. Phép toán vectơ
g. gián tiếp

Trả lời:
a. Ngăn xếp — tốt.
b. Bảng ký hiệu băm—không tốt. YÊU NGÀI NAM NHẤT
c. Tìm kiếm tuần tự—tốt.
d. Tìm kiếm nhị phân—không tốt.
đ. Mã thuần túy — tốt.
f. Hoạt động véc tơ — tốt.
g. Định hướng—không tốt.

9.4, Hãy xem xét các thuật toán thay thế trang sau đây. Xếp hạng các thuật toán này theo thang điểm năm từ “kém” đến “hoàn hảo” theo tỷ lệ lỗi trang của chúng. Tách biệt những thuật toán bị dị thường Belady với những thuật toán không mắc phải.
a. thay thế LRU
b. FIFO thay thế
c. thay thế tối ưu
d. Thay thế cơ hội thứ hai

Trả lời:
Thứ hạng thuật toán: (Từ 1 đến 4)
- Tối ưu (Không chịu sự ảnh hưởng của nghịch lý Belady)
- LRU (Không chịu sự ảnh hưởng của nghịch lý Belady)
- Cơ hội thứ hai (Có chịu sự ảnh hưởng của nghịch lý Belady)
- FIFO (Có chịu sự ảnh hưởng của nghịch lý Belady)

9,5, Khi bộ nhớ ảo được triển khai trong một hệ thống máy tính, sẽ có một số chi phí nhất định liên quan đến kỹ thuật và một số lợi ích nhất định. Liệt kê chi phí và lợi ích. Có thể cho các chi phí vượt quá lợi ích? 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. Lợi ích là tận dụng tốt bộ nhớ và không gian địa chỉ luận lý lớn hơn không gian địa chỉ vật lý.

9,6, Một hệ điều hành hỗ trợ bộ nhớ ảo được phân trang, sử dụng bộ xử lý trung tâm với thời gian chu kỳ là 1 micro giây. Mất thêm 1 micro giây để truy cập một trang khác với trang hiện tại. Trang có 1000 từ và thiết bị phân trang là một cái trống quay với tốc độ 3000 vòng quay mỗi phút và truyền 1 triệu từ mỗi giây. Các phép đo thống kê sau đây được lấy từ hệ thống:
• 1 phần trăm của tất cả các hướng dẫn được thực thi đã truy cập một trang khác với trang hiện tại.
• Trong số các hướng dẫn đã truy cập trang khác, 80 phần trăm đã truy cập trang đã có trong bộ nhớ.
• Khi một trang mới được yêu cầu, trang được thay thế đã được sửa đổi 50 phần trăm thời gian.
Tính toán thời gian hướng dẫn hiệu quả trên hệ thống này, giả sử rằng hệ thống chỉ chạy một quy trình và bộ xử lý không hoạt động trong quá trình truyền trống.

Trả lời:

Thời gian truy cập hiệu quả

$$
\begin{aligned}
&= 0.99 \times (1\ \mu\text{sec} + 0.008 \times (2\ \mu\text{sec}) \\
&\quad + 0.002 \times (10{,}000\ \mu\text{sec} + 1{,}000\ \mu\text{sec}) \\
&\quad + 0.001 \times (10{,}000\ \mu\text{sec} + 1{,}000\ \mu\text{sec}) \\
&= (0.99 + 0.016 + 22.0 + 11.0)\ \mu\text{sec} \\
&= 34.0\ \mu\text{sec}
\end{aligned}
$$

9.7, Xét mảng hai chiều A:
int A[][] = new int[100][100];
A[0][0 nằm ở đâu trong vị trí 200], trong một hệ thống bộ nhớ được phân trang với các trang có kích thước 200. Một quy trình nhỏ nằm trong trang 0 (Vị trí từ 0 đến 199) để thao tác ma trận; do đó, mọi lệnh tìm nạp sẽ từ trang 0. Đối với ba khung trang, có bao nhiêu lỗi trang được tạo bởi các vòng lặp khởi tạo mảng sau, sử dụng thay thế LRU và giả sử khung trang 1 có quy 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 sẽ xảy ra đối với các thuật toán thay thế sau đây, giả sử một, hai, ba, bốn, năm, sáu hoặc bảy khung hình? Hãy nhớ rằng tất cả các khung ban đầu đều trống, vì vậy các trang duy nhất đầu tiên của bạn sẽ có một lỗi trên mỗi trang.
• Thay thế LRU
• FIFO thay thế
• Thay thế tối ưu
Trả lời:

Number of frames	LRU	FIFO	Optimal
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 bit tham chiếu (chẳng hạn như thay thế cơ hội thứ hai hoặc mô hình bộ làm việc), nhưng phần cứng không cung cấp bit này. Phác thảo cách bạn có thể mô phỏng bit tham chiếu ngay cả khi phần cứng không cung cấp bit đó hoặc giải thích lý do tại sao không thể làm như vậy. Nếu có thể, hãy tính xem chi phí sẽ là bao nhiêu.

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. Ở lần tham chiếu đầu tiên, một cái bẫy đối với hệ điều hành được tạo ra. Hệ điều hành sẽ đặt 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 cho là tối ưu. Trong một số trường hợp thử nghiệm méo mó, sự bất thường của Belady xảy ra. Thuật toán mới có tối ưu không? 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 sẽ thay thế trang sẽ không được sử dụng trong thời gian dài nhất. Sự bất thường của Belady xảy ra khi một thuật toán thay thế trang loại bỏ một trang sẽ cần thiết trong tương lai gần. Một thuật toán tối ưu sẽ không chọn một trang như vậy.
9.11 Phân đoạn tương tự như phân trang nhưng sử dụng các “trang” có kích thước thay đổi. Định nghĩa hai thuật toán thay thế đoạn dựa trên lược đồ thay thế trang FIFO và LRU. Hãy nhớ rằng do các phân đoạn không có cùng kích thước nên phân đoạn được chọn để thay thế có thể không đủ lớn để chừa đủ vị trí liên tiếp cho phân đoạn cần thiết. Xem xét các chiến lược dành cho các hệ thống không thể di chuyển các phân đoạn và chiến lược dành cho các hệ thống có thể.
Trả lời:
a.FIFO. Tìm phân khúc đầu tiên đủ lớn để chứa phân khúc sắp tới. Nếu không thể di chuyển vị trí và không có phân đoạn nào đủ lớn, hãy chọn tổ hợp các phân đoạn có bộ nhớ liền kề, “gần với đầu tiên nhất trong danh sách” và có thể chứa phân đoạn mới. Nếu có thể di chuyển, hãy sắp xếp lại bộ nhớ để bộ nhớ đầu tiênNcác phân đoạn đủ lớn cho phân đoạn đến nằm liền kề trong bộ nhớ. Thêm bất kỳ không gian còn lại nào vào danh sách không gian trống trong cả hai trường hợp.
b.LRU. Chọn phân đoạn không được sử dụng trong khoảng thời gian dài nhất và đủ lớn, thêm mọi dung lượng còn lại vào danh sách dung lượng trống. Nếu không có phân đoạn nào đủ lớn, hãy chọn tổ hợp các phân đoạn "cũ nhất" liền kề trong bộ nhớ (nếu không có khả năng di chuyển) và đủ lớn.
Nếu có thể di dời, hãy sắp xếp lại cái cũ nhấtNphân đoạn liền kề trong bộ nhớ và thay thế chúng bằng phân đoạn mới.
9.12Hãy xem xét một hệ thống máy tính được phân trang theo yêu cầu trong đó mức độ

đa chương trình hiện được cố định ở mức bốn. Hệ thống gần đây đã được đo để xác định việc sử dụng CPU và đĩa hoán 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 mức sử dụng CPU không? Là phân trang giúp đỡ?
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:
Một. Đập đang xảy ra.
b. Mức sử dụng CPU đủ cao để bỏ qua mọi thứ và tăng mức độ đa chương trình.
c. Tăng mức độ đa chương trình.
9.13Chúng tôi có một hệ điều hành cho máy sử dụng thanh ghi cơ sở và giới hạn, nhưng chúng tôi đã sửa đổi máy để cung cấp bảng trang.
Các bảng trang có thể được thiết lập để mô phỏng các thanh ghi cơ sở và giới hạn không? Làm thế nào họ có thể được, hoặc tại sao họ không thể được?
Trả lời:Bảng trang có thể được thiết lập để mô phỏng các thanh ghi cơ sở và giới hạn miễn là bộ nhớ được phân bổ trong các phân đoạn có kích thước cố định. 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 phần đó của phân đoạn nằm trong bộ nhớ. Sẽ có một số vấn đề với sự phân mảnh nội bộ.

10, File system giao diện

10.1. Một số hệ thống tự động xóa tất cả các tệp người dùng khi người dùng đăng xuất hoặc công việc kết thúc, ngoại trừ khi người dùng yêu cầu rõ ràng rằng họ đã được lưu giữ; các hệ thống khác giữ tất cả các tệp được loại trừ khi người dùng xóa chúng theo một cách rõ ràng. Thảo luận về giá trị tương đối của từng phương pháp.
Trả lời:Việc xóa tất cả các tệp không phải do người dùng cụ thể lưu có thể có ưu điểm là 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 bị xóa cụ thể sẽ an toàn hơn cho người dùng tại chỗ không thể vô tình làm mất tệp do quên lưu chúng.

10.2. Tại sao một số hệ thống theo dõi loại tệp, trong khi các hệ thống khác để nó cho người dùng hoặc đơn giản là không khai thác nhiều loại tệp? Hệ thống nào "tốt hơn?"

- Trả lời:Một số hệ thống cho phép các thao tác tệp khác nhau dựa trên loại tệp (ví dụ: tệp ascii có thể được đọc dưới định dạng luồng trong khi tệp cơ sở dữ liệu có thể được đọc qua chỉ mục cho một khối). Các hệ thống khác nhau để khôi phục lại việc giải thích dữ liệu của tệp như vậy cho quy trình và không cung cấp hỗ trợ nào trong quá trình truy cập dữ liệu. Phương pháp “tốt hơn” phụ thuộc vào nhu cầu của các quy trình trên hệ thống và nhu cầu của người dùng đối với hệ điều hành. Nếu một hệ thống chủ yếu chạy các ứng dụng cơ sở dữ liệu, thì hệ thống điều hành có thể phát triển khai thác một loại tệp cơ sở dữ liệu và cung cấp các thao tác sẽ hiệu quả hơn vì làm cho mỗi chương trình phát triển cùng khai thác một thứ (có thể theo những cách khác nhau).Đối với các hệ thống có mục đích chung, có thể tốt hơn là chỉ phát triển các loại tệp cơ bản để duy trì kích thước hệ thống điều hành nhỏ hơn và cho phép tự động làm tối đa đối với các quy trình trên hệ thống.

10.3. Tương tự, một số hệ thống hỗ trợ nhiều loại cấu trúc cho dữ liệu của tệp, trong khi các hệ thống khác chỉ hỗ trợ một byte luồng. Những lợi thế và bất lợi là gì?

Trả lời:Một ưu điểm của công việc hệ thống hỗ trợ các cấu trúc tệp khác nhau 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, thì nó có thể phát triển hỗ trợ khai thác có hiệu quả hơn một ứng dụng. Nhược điểm của việc hệ thống cung cấp hỗ trợ cho các loại tệp được xác định 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 hệ thống cung cấp không thể chạy trên các hệ thống đó. Một chiến lược thay thế là để xác định hệ thống điều hành không hỗ trợ tệp cấu trúc cấu trúc và thay vào đó coi tất cả các tệp là một byte chuỗi. Đây là cách tiếp cận được thực hiện bởi các hệ thống UNIX. Ưu điểm của phương pháp này là nó đơn giản hóa công việc hỗ trợ điều hành hệ thống cho các tệp hệ thống, 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 ứng dụng, do đó làm giảm tình trạng hệ thống không thể cung cấp định nghĩa tệp cần thiết cho một ứng dụng cụ thể.

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

Trả lời:Nếu có thể sử dụng tùy chọn tên dài, thì có thể mô phỏng cấu trúc thư mục đa cấp cấu trúc. Điều này có thể được thực hiện, ví dụ, bằng cách sử dụng các ký tự “.” để chỉ ra khỏi cuối của 1 thu mục con. Vì vậy, ví dụ, tênjim.java.F1xác nhận rằngF1is một tệp trong thư mục con java giống với nó nằm trong thư mục gốc gim.

Nếu tên tệp bị giới hạn trong bảy ký tự, thì Lược đồ trên không thể sử dụng được và do đó, nói chung, câu trả lời làKHÔNG. Cách tiếp cận tốt nhất tiếp theo trong vấn đề này là sử dụng một tệp công cụ có thể làm bảng ký hiệu (thư mục) để ánh xạ các tên tùy chọn dài (chẳng hạn như thời hạn)jim.java.F1) sang tên ngắn hơn (ví dụ như XX00743), sau đó được sử dụng để truy cập tệp thực tế.

10.6. Ví dụ về ứng dụng trong đó dữ liệu trong tệp sẽ được truy cập theo thứ tự sau:
a. tuần tự
b. tự nhiên
Trả lời:
a. Trong nội dung của tập tin.
b. Trong nội dung bản ghi của tôi. 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 phòng vệ có thể phát sinh.
b. Đề xuất một kế hoạch để giải quyết từng vấn đề phòng vệ mà bạn nêu tên 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 nhau đánh bại sơ đồ phòng vệ vệ quyền truy cập.

b. Không cho phép người dùng ghi trực tiếp vào thư mục con. Thay vào đó, hãy cung cấp các hệ thống hoạt động để làm như vậy.

11, File-System Implementation

Câu 1 : Hãy xem xét một tệp hiện bao gồm 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 hoạt động I/O đĩa
được yêu cầu cho các chiến lược phân bổ liền kề, được liên kết và được lập chỉ mục (một cấp), nếu đối với một khối, các điều kiện sau được đáp ứng. bên trong
trường hợp phân bổ liền kề, giả sử rằng không có chỗ để phát triển trong
bắt đầu, nhưng có chỗ để phát triển cuối cùng. Giả sử rằng khối
thông tin được thêm vào được lưu trữ trong bộ nhớ.

	Contiguous	Linked	Indexed
A	210	1	1
B	101	52	1
C	1	3	1
d	198	1	0
E	98	52	0
f	0	100	0

Câu 2 : Những vấn đề gì có thể xảy ra nếu một hệ thống cho phép một hệ thống tập tin được gắn đồng thời tại nhiều vị trí?
Việc có đồng thời nhiều đường dẫn cho cùng 1 tệp, điều này có thể gây nhầm lẫn cho người dung hoặc dẫn đến sai lầm (xóa tệp bằng đường 1 đường dẫn có thể xóa tệp trong tất cả đường dẫn khác)
Câu 3 : Tại sao bản đồ bit để phân bổ tệp phải được lưu trữ trên bộ nhớ chung, thay vì hơn trong bộ nhớ chính?
Trong trường hợp hệ thống gặp sự cố (lỗi bộ nhớ), danh sách dung lượng trống sẽ không bị mất nếu như bản đồ bit đã lưu dữ liệu vào bộ nhớ chính

Câu 4 Hãy xem xét một hệ thống hỗ trợ các chiến lược tiếp giáp, liên kết, và phân bổ được lập chỉ mục. Những tiêu chí nào nên được sử dụng để quyết định lược được sử dụng tốt nhất cho một tệp cụ thể?
- Contiguous : nếu tệp thường xuyên được truy cập tuần tự và tệp tương đối nhỏ
- Linked nều tệp lớn và thường xuyê truy cập tuần tự
- Indexed neu tệp lớn và thường xuyên truy cập ngẫu nhiên

Câu 5 Một vấn đề với phân bổ liên tục là người dùng phải cấp phát trước đủ không gian cho mỗi tập tin. Nếu tập tin phát triển lớn hơn không gian được cấp phát cho nó, các hành động đặc biệt phải được thực hiện. Một giải pháp cho vấn đề này là định nghĩa một cấu trúc tập tin bao gồm một khu vực liên tục ban đầu (có kích thước được chỉ định). Nếu khu vực này đã đầy, hệ điều hành sẽ tự động định nghĩa một khu vực tràn đầy được liên kết với khu vực liên tục ban đầu. Nếu khu vực tràn đầy cũng đã đầy, một khu vực tràn đầy khác được cấp phát. So sánh cài đặt tập tin này với các cài đặt liên tục và liên kết tiêu chuẩn.
- Phương pháp này yêu cầu nhiều chi phí hơn so với phân bổ liền kề tiêu chuẩn. Nó đòi hỏi ít chi phí hơn so với liên kết tiêu chuẩn phân bổ

Câu 6 : Bộ nhớ đệm giúp cải thiện hiệu xuất như thế nào ? tại sao hệ thống không sử dụng nhiều hoặc bộ nhớ đệm lớn hơn nếu chúng hữu ích như vậy?
Bộ nhớ cache cho phép các thành phần có tốc độ khác nhau giao tiếp hiệu quả hơn bằng cách tạm thời lưu trữ dữ liệu từ thiết bị chậm hơn, trong một thiết bị nhanh hơn (bộ đệm). Bộ nhớ cache, gần như theo định nghĩa, nhiều hơn đắt hơn thiết bị mà chúng đang lưu vào bộ nhớ đệm, vì vậy hãy tăng số lượng hoặc kích thước của bộ đệm sẽ làm tăng chi phí hệ thống
Câu 7 : Tại sao nó lại thuận lợi cho người dùng đối với một hệ điều hành để phân bổ tự động các bảng nội bộ của nó? các hình phạt cho các hoạt động là gì hệ thống để làm như vậy?

Bảng động cho phép linh hoạt hơn trong tăng trưởng sử dụng hệ thống bảng không bao giờ bị vượt quá, tránh giới hạn sử dụng giả tạo. Không may thay, cấu trúc kernel và mã phức tạp hơn, vì vậy có nhiều tiềm năng cho các lỗi. Việc sử dụng một tài nguyên có thể lấy đi nhiều tài nguyên hệ thống hơn (bằng cách phát triển để đáp ứng các yêu cầu) so với bảng tĩnh
Câu 8 Hãy giải thích cách lớp VFS cho phép hệ điều hành dễ dàng hỗ trợ nhiều loại hệ thống tệp khác nhau?
VFS giới thiệu một lớp định hướng trong việc triển khai hệ thống tệp. Theo nhiều cách, nó tương tự như lập trình hướng đối tVFS giới thiệu một lớp định hướng trong việc triển khai hệ thống tệp. Theo nhiều cách, nó tương tự như lập trình hướng đối tượng kỹ xảo. Các cuộc gọi hệ thống có thể được thực hiện một cách tổng quát (không phụ thuộc vào tệp loại hệ thống). Mỗi loại hệ thống tệp cung cấp các cuộc gọi chức năng và dữ liệu của nócấu trúc cho lớp VFS. Một cuộc gọi hệ thống được dịch sang đúng 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 tương tự như vậy là độc lập với hệ thống tệp. Bản dịch ở lớp VFS biến các lệnh gọi chung này thành các lệnh gọi dành riêng cho hệ thống tệp hoạt động ượng kỹ xảo.
Các cuộc gọi hệ thống có thể được thực hiện một cách tổng quát (không phụ thuộc vào tệp loại hệ thống). Mỗi loại hệ thống tệp cung cấp các cuộc gọi chức năng và dữ liệu của nó cấu trúc cho lớp VFS. Một cuộc gọi hệ thống được dịch sang đúng 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 ấu trúc cuộc gọi hệ thống tương tự như vậy là độc lập với hệ thống tệp. Bản dịch ở lớp VFS biến các lệnh gọi chung này thành các lệnh gọi dành riêng cho hệ thống tệp hoạt động

12,

Cau 1
a. t = 0.95 + 0.05L
b. FCFS 362.60; SSTF 95.80; SCAN 497.95; LOOK 174.50; C-SCAN 500.15; (and 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 của SSTF hơn FCFS 74% đối với thời gian tìm kiếm. Nếu chúng ta bao gồm chi phí cho độ trễ quay và truyền dữ liệu, tỷ lệ phần trăm tăng tốc sẽ ít hơn
Câu 2 :Lập lịch đĩa, ngoài lập lịch FCFS, có hữu ích trong môi trường một người dùng? Giải thich câu trả lơi của bạn.

Trong môi trường sử dụng đơn người dùng, hàng đợi I/O thường là trống rỗng. Các yêu cầu thường đến từ một quá trình duy nhất cho một khối hoặc một chuỗi các khối liên tiếp. Trong những trường hợp này, FCFS là một phương pháp lập lịch đĩa kinh tế. Nhưng LOOK gần như dễ lập trình và sẽ cung cấp hiệu suất tốt hơn khi nhiều quá trình thực hiện I/O đồng thời, chẳng hạn như khi trình duyệt web truy xuất dữ liệu phía sau khi hệ điều hành đang phân trang và một ứng dụng khác đang hoạt động phía trước.
Câu 3 Giải thích tại sao lập biểu SSTF có xu hướng ưu tiên các trụ ở giữa hơn xi lanh trong cùng và ngoài cùng
Trung tâm của đĩa là vị trí có khoảng cách trung bình nhỏ nhất đến tất cả các track khác. Do đó, đầu đĩa sẽ di chuyển khỏi các cạnh của đĩa. Đây là cách khác để nghĩ về vấn đề này. Vị trí hiện tại của đầu đĩa chia các cylinder thành hai nhóm. Nếu đầu đĩa không ở giữa đĩa và một yêu cầu mới đến, yêu cầu mới đó có khả năng cao hơn nằm trong nhóm bao gồm trung tâm của đĩa; do đó, đầu đĩa có khả năng di chuyển theo hướng đó.
Câu 4 : Tại sao độ trễ quay thường không được 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?
Hầu hết các đĩa cứng không xuất thông tin về vị trí quay của chúng đến cho máy chủ. Ngay cả khi chúng làm được điều đó, thời gian để thông tin này đến bộ lập lịch sẽ bị không chính xác và thời gian tiêu thụ bởi bộ lập lịch là biến đổi, vì vậy thông tin về vị trí quay sẽ trở nên không chính xác. Hơn nữa, các yêu cầu đĩa thường được đưa ra dưới dạng các số khối logic và bản đồ giữa các khối logic và vị trí vật lý là rất phức tạp.

Câu 5 : Lập lịch đĩa cố gắng giảm thời gian trên đầu đĩa để định vị vị trí. Vì đĩa RAM có thời gian truy cập đồng đều, nên lập lịch gần như là không cần thiết. Sự so sánh giữa đĩa RAM và bộ nhớ đệm đĩa chính không có ảnh hưởng đến lập lịch đĩa cứng, vì chúng ta chỉ lập lịch cho các trường hợp không trúng bộ đệm, không phải các yêu cầu tìm thấy dữ liệu của chúng trong bộ nhớ chính.
Câu 6 : tại sao cân bằng vào ra hệ thống tệp giữa các đĩa và bộ điều khiển trên một hệ thống trong môi trường đa nhiệm là quan trọng

Một hệ thống chỉ có thể hoạt động ở tốc độ của đường ống chậm nhất. Ổ đĩa hoặc bộ điều khiển đĩa thường là điểm chặn chính trong các hệ thống hiện đại vì hiệu suất của chúng không thể theo kịp hiệu suất của CPU và bus hệ thống. Bằng cách cân bằng I/O giữa các đĩa và bộ điều khiển, không có đĩa hoặc bộ điều khiển cá nhân nào bị quá tải, để tránh điểm chặn này.
Câu 7 : Sự đánh đổi liên quan đến việc đọc lại các trang mã từ tệp là gì hệ thống so với việc sử dụng không gian hoán đổi để lưu trữ chúng?
Nếu các trang mã được lưu trữ trong không gian trao đổi, chúng có thể được chuyển đổi nhanh hơn sang bộ nhớ chính (vì phân bổ không gian trao đổi được điều chỉnh cho hiệu suất nhanh hơn so với phân bổ hệ thống tệp tổng quát). Sử dụng không gian trao đổi có thể yêu cầu thời gian khởi động nếu các trang được sao chép đến đó khi quá trình được gọi ra thay vì chỉ được phân trang đến không gian trao đổi khi cần thiết. Ngoài ra, phải phân bổ thêm không gian trao đổi nếu nó được sử dụng cho cả trang mã và trang dữ liệu.
Câu 8 Có cách nào để triển khai lưu trữ thực sự ổn định không?
Ổ đĩa lưu trữ thực sự ổn định sẽ không bao giờ mất dữ liệu. Kỹ thuật cơ bản cho lưu trữ ổn định là duy trì nhiều bản sao của dữ liệu, để nếu một bản sao bị hỏng, thì vẫn có bản sao khác sẵn sàng để sử dụng. Nhưng với bất kỳ kế hoạch nào, chúng ta có thể tưởng tượng được một thảm họa đủ lớn để tất cả các bản sao đều bị hỏng.
Câu 9 :
a. Ổ cứng quay 120 lần mỗi giây, và mỗi lần quay chuyển một track 80 KB. Do đó, tốc độ chuyển dữ liệu liên tục có thể được xấp xỉ là 9600 KB/s.
b. Giả sử 100 cylinder là một lượng dữ liệu lớn. Tốc độ chuyển là tổng số byte chia tổng thời gian. Tổng số byte là: 100 cyl * 20 trk/cyl * 80 KB/trk, tức là 160.000 KB. Thời gian: thời gian quay + thời gian chuyển track + thời gian chuyển cylinder. Thời gian quay là 2000 trks / 120 trks mỗi giây, tức là 16,667 giây. Thời gian chuyển track là 19 lần chuyển mỗi cylinder * 100 cylinder * 0,5 ms, tức là 950 ms. Thời gian chuyển cylinder là 99 * 2 ms, tức là 198 ms. Do đó, tổng thời gian là 16,667 + 0,950 + 0,198, tức là 17,815 giây. (Chúng ta bỏ qua bất kỳ thời gian tìm kiếm ban đầu và độ trễ quay, có thể thêm khoảng 12 ms vào lịch trình, tức là 0,1%.) Do đó, tốc độ chuyển là 8981,2 KB/s. Tổng thời gian chuyển track và cylinder khoảng 6,5%.
c. Thời gian cho mỗi lần chuyển dữ liệu là 8 ms cho tìm kiếm + 4,167 ms trễ quay trung bình + 0,052 ms (tính từ 1 / (120 trk mỗi giây * 160 sector mỗi trk)) để quay một sector qua đầu đĩa khi đọc. Chúng ta tính số lần chuyển dữ liệu mỗi giây là 1 / (0,012219), tức là 81,8. Vì mỗi lần chuyển dữ liệu là 0,5 KB, nên tốc độ chuyển dữ liệu là 40,9 KB/s.
d. Chúng ta bỏ qua việc chuyển track và cylinder để đơn giản hóa. Đối với các lần đọc kích thước 4 KB, 8 KB và 64 KB, số lần chuyển dữ liệu tương ứng được tính từ tìm kiếm, trễ quay và thời gian chuyển tròn như trong mục trước, cho (lần lượt) 1 / (0,0126), 1 / (0,013) và 1 / (0,019). Do đó, chúng ta có được 79,4, 76,9 và 52,6 l
Câu 10 :
Đối với I/O ngẫu nhiên 8 KB trên đĩa ít tải, với thời gian truy cập ngẫu nhiên tính được khoảng 13 ms (xem Bài tập 12.9), tỷ lệ truyền dữ liệu hiệu quả là khoảng 615 MB/s. Trong trường hợp này, 15 đĩa cứng sẽ có tỷ lệ truyền dữ liệu tổng hợp nhỏ hơn 10 MB/s, điều này không nên làm cho bus quá tải. Đối với việc đọc ngẫu nhiên 64 KB từ một ổ đĩa ít tải, tỷ lệ truyền dữ liệu là khoảng 3,4 MB/s, do đó năm ổ đĩa hoặc ít hơn sẽ làm cho bus quá tải. Đối với việc đọc 8 KB với hàng đợi đủ lớn để giảm thời gian tìm kiếm trung bình xuống còn 3 ms, tỷ lệ truyền dữ liệu là khoảng 1 MB/s, vì vậy băng thông bus có thể đủ để chứa 15 ổ đĩa.
Câu 11
Vì đĩa chứa 22.400.000 sector, xác suất yêu cầu một trong 100 sector được định vị lại là rất nhỏ. Một ví dụ về trường hợp tồi nhất là khi chúng ta cố gắng đọc một trang 8 KB, nhưng một sector ở giữa bị lỗi và đã được định vị lại ở vị trí tệ nhất trên một track khác trong cylinder đó. Trong trường hợp này, thời gian để lấy dữ liệu có thể là 8 ms để tìm kiếm, cộng với hai lần chuyển track và hai lần chờ quay đầy đủ. Có khả năng rằng bộ điều khiển hiện đại sẽ đọc tất cả các sector tốt được yêu cầu từ track ban đầu trước khi chuyển sang track dự phòng để lấy sector đã được định vị lại và do đó chỉ gây ra một lần chuyển track và chờ quay đầy đủ. Vì vậy, thời gian sẽ là 8 ms để tìm kiếm + 4.17 ms trễ xoay trung bình + 0.05 ms chuyển track + 8.3 ms trễ xoay đầy đủ + 0.83 ms thời gian đọc (8 KB là 16 sector, 1/10 quay track). Do đó, thời gian để phục vụ yêu cầu này sẽ là 21,8 ms, cung cấp tốc độ I/O là 45,9 yêu cầu mỗi giây và băng thông hiệu quả là 367 KB/s. Đối với một ứng dụng bị giới hạn thời gian nghiêm trọng, điều này có thể quan trọng, nhưng ảnh hưởng tổng thể trong trung bình có trọng số của 100 sector được định vị lại và 22,4 triệu sector tốt là không đáng kể.
Câu 12 Trong một hộp đĩa, tác động của việc có nhiều tệp mở hơn số ổ đĩa trong hộp đĩa sẽ như thế nào?
Có thể xảy ra hai kết quả xấu. Một khả năng là đói đường ứng dụng mà phát sinh các I/O chặn đến băng từ chưa được gắn trên đầu đọc. Một khả năng khác là xảy ra tình trạng thrashing, khi jukebox được yêu cầu chuyển đổi băng sau mỗi hoạt động I/O.
Câu 13 Nếu ổ đĩa cứng từ tính cuối cùng có chi phí trên mỗi gigabyte tương tự như băng từ, liệu băng từ sẽ trở nên lỗi thời hay vẫn cần thiết? Hãy giải thích câu trả lời của bạn.
Băng từ dễ dàng tháo rời nên chúng rất hữu ích cho việc sao lưu ngoài địa điểm và cho việc truyền tải số lượng lớn dữ liệu (bằng cách gửi các đĩa). Như một quy tắc chung, đĩa cứng từ không phải là phương tiện có thể tháo rời.
Câu 14
Một. Đối với 512 byte, tốc độ truyền hiệu quả được tính như sau.
ETR = quy mô chuyển/thời gian chuyển.
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 hiệu quả là 512B/15,0097ms = 33,12 KB/giây.
ETR cho 8KB = .47MB/giây.
ETR cho 1 MB = 4,65 MB/giây.
ETR cho 16 MB = 4,98 MB/giây.
b. Mức 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 16 MB = 99,6%.
c. Tính toán 0,25 = ETR/STR, tính kích thước truyền X.
STR = 5 MB, vì vậy 1,25 MB/S = ETR.
1,25 MB/giây * ((X/5) + .015) = X.
.25X + .01875 = X .
X = 0,025MB.
d. Đĩa là thiết bị truy cập ngẫu nhiên để truyền dữ liệu lớn hơn K byte (trong đó K > kích thước khối đĩa) và là thiết bị truy cập tuần tự cho chuyển khoản nhỏ hơn.

đ. Tính toán kích thước truyền tối thiểu để sử dụng chấp nhận được bộ nhớ đệm:
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 = 80 MB, 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 = .5, L = 60 giây.
.5 (XMB/2 + 60) = XMB.
.25XMB + 30 = XMB.

X = 40MB.
f. Nó phụ thuộc vào cách nó đang được sử dụng. Giả sử chúng ta đang sử dụng
băng để khôi phục bản sao lưu. Trong trường hợp này, băng hoạt động như một
thiết bị truy cập tuần tự nơi chúng tôi đang đọc tuần tự nội dung của băng. Một ví dụ khác, giả sử chúng ta đang sử dụng băng để truy cập nhiều loại bản ghi được lưu trữ trên băng. trong này dụ, truy cập vào băng là tùy ý và do đó được coi là
ngẫu nhiên

câu 15
a. Giả sử một ổ đĩa cứng chứa 10 GB. Sau đó 100 ổ đĩa có dung lượng 1 TB, 100.000 ổ đĩa có dung lượng 1 PB, và 100.000.000 ổ đĩa có dung lượng 1 EB. Để lưu trữ 4 EB sẽ cần khoảng 400 triệu ổ đĩa. Nếu một băng từ từ có dung lượng 40 GB, chỉ cần 100 triệu băng từ. Nếu một băng quang có dung lượng lớn hơn gấp 50 lần so với một băng từ, chỉ cần 2 triệu băng quang để đủ. Nếu một hộp chiếu khảm có thể lưu trữ 180 GB, sẽ cần khoảng 22,2 triệu hộp chiếu khảm.

b. Một ổ đĩa cứng loại 3,5 inch có kích thước khoảng 1 inch cao, 4 inch rộng và 6 inch sâu. Chuyển sang đơn vị feet, đó là 1/12 feet chiều cao, 1/3 feet chiều rộng và 1/2 feet chiều sâu, tương đương với 1/72 feet khối. Nếu đóng gói chặt chẽ, 400 triệu ổ đĩa sẽ chiếm khoảng 5,6 triệu feet khối. Nếu chúng ta cho phép một tỷ lệ hai cho khoảng trống không khí và không gian cho nguồn cung cấp điện, dung tích yêu cầu là khoảng 11 triệu feet khối.

c. Một hộp băng từ loại 1/2 inch có chiều cao khoảng 1 inch và chiều dài rộng 4,5 inch. Thể tích là khoảng 1/85 feet khối. Với 100 triệu băng từ từ được đóng gói chặt chẽ, thể tích là khoảng 1,2 triệu feet khối. Với 2 triệu băng quang, thể tích là 23.400 feet khối.

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

13,

13.1 Nêu ba ưu điểm của việc đặt chức năng trong bộ điều khiển thiết bị, chứ không phải trong hạt nhân. Nêu ba nhược điểm.
Trả lời: Ba ưu điểm:
Lỗi ít có khả năng gây ra sự cố hoạt động của hệ thống.
Hiệu suất có thể được cải thiện bằng cách sử dụng phần cứng chuyên dụng và thuật toán mã hóa cứng
Nhân được đơn giản hóa bằng cách di chuyển các thuật toán ra khỏi nó
Ba nhược điểm: Lỗi khó sửa hơn—một phiên bản phần sụn mới hoặc phần cứng mới là cần thiết
Việc cải thiện các thuật toán cũng yêu cầu cập nhật phần cứng thay vì chỉ là bản cập nhật kernel hoặc trình điều khiển thiết bị
Các thuật toán nhúng có thể xung đột với việc sử dụng thiết bị của ứng dụng, gây giảm hiệu suất.

13.2 Ví dụ về bắt tay trong Phần 13.2 đã sử dụng 2 bit: một bit bận và một bit sẵn sàng cho lệnh. Có thể thực hiện bắt tay này với chỉ 1 chút? Nếu có, hãy mô tả giao thức. Nếu không, giải thích tại sao 1 bit là không đủ.
Trả lời: Có thể, sử dụng thuật toán sau. Hãy giả sử
chúng tôi chỉ cần sử dụng bit bận (hoặc bit sẵn sàng cho lệnh; câu trả lời này là luôn như nhau). Khi bit tắt, bộ điều khiển không hoạt động. Các máy chủ ghi 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 bắt đầu quá trình hoạt động tiếp theo. Giải pháp này yêu cầu cả máy chủ và bộ điều khiển đã đọc và ghi quyền truy cập vào cùng một bit, điều này có thể làm phức tạp mạch và tăng chi phí của bộ điều khiển.

13.3 Tại sao một hệ thống có thể sử dụng I/O điều khiển ngắt để quản lý một sê-ri cổng, nhưng bỏ phiếu I/O để quản lý bộ xử lý ngoại vi, chẳng hạn như thiết bị đầu cuối tập trung?
Trả lời: Bỏ phiếu có thể hiệu quả hơn so với I/O điều khiển ngắt. Cái này là trường hợp khi 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, hãy sử dụng các ngắt, một tập hợp các cổng nối tiếp, chẳng hạn như các cổng trong thiết bị đầu cuối bộ tập trung có thể tạo ra nhiều thao tác I/O ngắn và làm gián đoạn đối với mỗi người có thể tạo ra một tải nặng trên hệ thống. đúng lúc vòng bỏ phiếu có thể giảm tải mà không lãng phí nhiều tài nguyên thông qua vòng lặp mà không cần I/O.

13.4 Bỏ phiếu để hoàn thành I/O có thể lãng phí một lượng lớn chu kỳ CPU nếu bộ xử lý lặp lại vòng lặp bận chờ nhiều lần trước I/O hoàn thành. Nhưng nếu thiết bị I/O đã sẵn sàng hoạt động, việc bỏ phiếu có thể hiệu quả hơn là bắt và gửi một ngắt. Mô tả một chiến lược kết hợp kết hợp bỏ phiếu, ngủ và gián đoạn cho dịch vụ thiết bị I/O. Đối với mỗi trong số ba chiến lược này (bỏ phiếu thuần túy, thuần túy ngắt, kết hợp), mô tả một môi trường điện toán trong đó chiến lược hiệu quả hơn một trong hai chiến lược còn lại.
Trả lời: Một cách tiếp cận 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 đang bận ở mức N+1, chúng ta có thể thiết lập một ngắt và ngủ. Cách tiếp cận này sẽ tránh được thời gian dài chu kỳ chờ bận. Phương pháp này sẽ là tốt nhất trong thời gian rất 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 lượng nhỏ các chu kỳ) 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 là tốt nhất với khi biết thời gian chờ đợi lâu.

13.5 DMA tăng tính đồng thời của hệ thống như thế nào? Nó phức tạp như thế nào trong 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 bus 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 trở thành chủ xe buýt. Ăn cắp chu kỳ cũng có thể cần thiết để cho phép bộ điều khiển CPU và DMA chia sẻ việc sử dụng xe buýt bộ nhớ.

13.6 Tại sao việc mở rộng quy mô tốc độ thiết bị và bus hệ thống lại quan trọng như 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 tăng tổng hiệu suất hệ thống chỉ bằng 50%. Nhân đôi cả hai khía cạnh hệ thống sẽ tăng hiệu suất lên 100%. Nói chung, điều quan trọng là phải loại bỏ
tắc nghẽn hệ thống hiện tại và để tăng hiệu suất toàn bộ hệ thống, thay vì tăng hiệu suất của từng thành phần hệ thống 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 một hoạt động STREAMS. Mô-đun STREAMS sửa đổi luồng dữ liệu giữa đầu (giao diện người dùng) và trình điều khiển.

14

Câu 1 : Điểm khác biệt chính giữa danh sách khả năng và danh sách truy cập ?
- Answer: Danh sách truy cập là danh sách cho từng đối tượng bao gồm các miền với một tập quyền truy cập không trống cho đối tượng đó. Một danh sách khả năng là một danh sách các đối tượng và các hoạt động được phép trên các đối tượng đó

Câu 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 bit ngẫu nhiên. Vì mục đích gì thì một kế hoạch như vậy sẽ hữu ích?

- Answer: 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, cố ý hoặc bởi tai nạn, bởi một chương trình khác. Điều này đặc biệt hữu ích cho bất kỳ thông tin mật.

Câu 3 :Trong hệ thống bảo vệ vòng, cấp độ 0 có quyền truy cập lớn nhất vào các đối tượng và cấp độ n (lớn hơn 0) có ít quyền truy cập hơn. Quyền truy cập của một chương trình tại một cấp độ cụ thể trong cấu trúc vòng được coi như một tập các khả năng. Quan hệ giữa khả năng của một miền ở cấp độ j và một miền ở cấp độ i đối với một đối tượng là gì (đối với j> i)?
- Answer: Dj là tập con của Di

Câu 4 : Hệ thống RC 4000 (và các hệ thống khác) đã định nghĩa một cây các tiến trình (gọi là cây tiến trình) sao cho tất cả các hậu duệ của một tiến trình chỉ nhận được các tài nguyên (đối tượng) và quyền truy cập từ các tổ tiên của chúng. Do đó, một hậu duệ sẽ không bao giờ có khả năng làm bất cứ điều gì mà tổ tiên của nó không thể làm được. Gốc của cây là hệ điều hành, có khả năng làm bất cứ điều gì. Giả sử tập quyền truy cập được biểu diễn bằng ma trận truy cập, A. A (x, y) xác định quyền truy cập của tiến trình x đối với đối tượng y. Nếu x là một hậu duệ của z, mối quan hệ giữa A (x, y) và A (z, y) đối với một đối tượng y tùy ý là gì?

Đối với một đối tượng bất kỳ y, nếu x là một hậu duệ của z, thì A(x,y) phụ thuộc vào A(z,y). Cụ thể, quyền truy cập của x đối với đối tượng y là một tập con của quyền truy cập của z đối với đối tượng y. Tức là, x không được cấp quyền truy cập đối với đối tượng y nếu z không được cấp quyền truy cập đối với đối tượng y hoặc quyền truy cập của z đối với đối tượng y là một tập con của quyền truy cập của x đối với đối tượng y.
- Answer: A(x,y) tập con của A(z,y)

Câu 5: Nếu một ngăn xếp được chia sẻ được sử dụng cho truyền tham số, thì có thể xảy ra các vấn đề về bảo vệ như thế nào?
- Answer: Nội dung của ngăn xếp có thể bị tổn hại bởi tiến trình khác chia sẻ ngăn xếp

Câu 6 : "Xét một môi trường tính toán trong đó mỗi quy trình và mỗi đối tượng trong hệ thống được gán một số duy nhất. Giả sử chúng ta cho phép một quy trình có số n truy cập vào một đối tượng có số m chỉ khi n > m. Chúng ta có loại cấu trúc bảo vệ nào?"
- Answer: Cấu trúc phân cấp

Câu 7: Xem xét môi trường tính toán trong đó một tiến trình được cấp quyền truy cập vào một đối tượng chỉ n lần. Đề xuất một kế hoạch để triển khai chính sách này.
- Answer: Thêm một số nguyên đếm khả năng

Câu 8 :
- Answer: Bộ đếm tham chiếu

Câu 9 :

Câu 10 :
Danh sách khả năng được xem xét là “ đối tượng bảo vệ” và là lối truy cập duy nhất bởi người dùng. Hệ điều hành đảm bảo người dùng không thể truy cập vào danh sách khả năng truy cập.