ÔN TẬP

KIỂM TRA GIỮA KỲ

HỌC PHẦN: ***NGUYÊN LÝ HỆ ĐIỀU HÀNH***

GV: ***ThS. Nguyễn Danh Minh Trí***

HỌC KỲ 2, NĂM HỌC 2022-2023

LỚP: D20CNTT03, D20CNTT04

Tiến trình là một thực thể (instance) của một chương trình máy tính đang được thực thi bởi một hoặc nhiều luồng (thread). Một tiến trình có riêng một không gian địa chỉ, có ngăn xếp (stack) riêng rẽ, có bảng chứa các file descriptor được mở cùng tiến trình và đặc biệt là có một định danh PID (process identifier) duy nhất trong toàn bộ hệ thống vào thời điểm tiến trình đang chạy. Khi cho nhiều tiến trình hoạt động giả lập song hành, mục tiêu chính là làm tăng mức độ đa chương, chứ không phải tăng tốc độ xử lý hoặc tăng hiệu suất dùng CPU.

Tiến trình có thể ở trạng thái blocked của một tiến trình khi nó đang chờ một sự kiện hoặc chờ nhập xuất (có thể từ người dùng). Tiến trình thường không bị khóa khi đã được xử lý xong hoặc bị ngắt bởi tiến trình khác và dĩ nhiên sẽ không bị block khi đang được xử lý hoặc ở trong hàng đợi chờ điều phối.

Hệ điều hành có chức năng chính là hỗ trợ việc quản lí tài nguyên, qua đó giúp cho người sử dụng khai thác chức năng của phần cứng máy tính dễ dàng hơn, hiệu quả hơn. Hệ điều hành cũng có chức năng khác như: quản lý bộ nhớ, quản lí tập tin, quản lí tiến trình, điều hành hệ thống và giúp cho người sử dụng khai thác chức năng của phần cứng máy tính dễ dàng hơn, hiệu quả hơn. Ngoài ra, hệ điều hành còn giúp khai thác chức năng của phần cứng máy tính.

Một tiến trình chuyển từ trạng thái Running sang trạng thái Ready khi hết thời gian sử dụng CPU. Tiến trình sẽ không chuyển Running sang Ready khi nó bị treo hoặc chờ nhập xuất, chờ một sự kiện nào đó. Ví dụ: tiến trình có đoạn code : `void main() { printf("Hello world\n"); printf("Hi! Man"); exit(0); }` sẽ trải qua ít nhất 3 lần trạng thái Ready. Chúng ta gọi môi trường giao tiếp giữa chương trình của người sử dụng và hệ điều hành là các system calls (lời gọi hệ thống). System call không thể bị nhầm lẫn với môi trường giao tiếp giữa chương trình và phần cứng cũng như là môi trường giao tiếp giữa phần cứng và hệ điều hành.

Có nhiều cách tạm dừng một tiến trình. Tuy nhiên, thao tác Suspend() thường dùng để tạm dừng một tiến trình đang dừng. Thao tác này khác với Pause() cũng như Abort() hay Kill(). Chương trình điều khiển trực tiếp các thiết bị thường gọi là chương trình vi điều khiển. Cần phân biệt khái niệm này với chương trình điều khiển hệ thống/máy tính, cũng như các chương trình điều khiển gián tiếp các thiết bị.

Best-fit, không phải là Worst-fit, cũng không giống như First-fit, là thuật toán chọn vùng trống đầu tự do nhỏ nhất nhưng đủ lớn để nạp tiến trình. Các chương trình được cài thêm (như chương trình diệt virus, phần mềm chat, MSTeam, Zalo) thường không được xem là các chương trình hệ thống. Trong khi đó, hệ điều hành, trình biên dịch và thông dịch chính là các chương trình hệ thống.

Đặt ra khái niệm độ ưu tiên của các tiến trình nhằm hỗ trợ cho việc điều phối được hiệu quả hơn. Điều này khác với mục đích phân biệt các tiến trình có độ ưu tiên thấp cần thực hiện trước hay để dò xem tiến trình sử dụng CPU nhiều hay ít hoặc tiến trình chiếm nhiều vùng nhớ hay ít vùng nhớ. Đồng bộ hóa các tiến trình

là đảm bảo các tiến trình xử lý song song không tác động sai lệch đến nhau, đảm bảo tại một thời điểm, chỉ có một tiến trình được quyền truy xuất một tài nguyên không thể chia sẻ và các tiến trình hợp tác với nhau để hoàn thành công việc.

Hệ điều hành là phần mềm điều khiển thiết bị phần cứng, luôn luôn phải có để máy tính hoạt động, hỗ trợ quản lý và phân phối tài nguyên máy tính phục vụ cho các ứng dụng. Với địa chỉ logic <s,d> và thanh ghi nền STBR, thanh ghi giới hạn stlr. Địa chỉ vật lí được tính tương ứng với địa chỉ logic là stbr+s+d. Trong Hệ điều hành UNIX, Sử dụng cơ chế liên lạc signal, người dùng nhấn phím Ctrl-C để ngắt xử lý tiến trình. Luồng (thread) là các phần nhỏ của tiến trình, có trạng thái như tiến trình, sở hữu con trỏ lệnh, tập thanh ghi, và stack nhưng không có không gian địa chỉ. Thread còn sở hữu các biến cục bộ riêng của nó.

Nguyên lý phân phối độc quyền thường thích hợp với hệ thống xử lí theo lô, nhưng không thích hợp với hệ thống đa chương cũng như hệ thống chia sẻ tương tác và hệ thống xử lí theo thời gian thực. Tiến trình xử lí tín hiệu theo cách riêng của nó, bằng cách gọi hàm xử lí tín hiệu và tiến trình có thể trao đổi dữ liệu, nhưng tiến trình không thể thông báo cho nhau về một sự kiện. Pipe là cơ chế liên lạc 1 chiều, do vậy tiến trình đọc pipe sẽ bị khóa nếu pipe trống, tiến trình ghi pipe sẽ bị khóa nếu pipe đầy và cho phép truyền dữ liệu không có cấu trúc

Đảm bảo sự công bằng là một trong những mục tiêu quan trọng của điều phối tiến trình. Tuy nhiên, còn các mục tiêu khác như cực tiểu hoá thời gian phản hồi, cực tiểu hóa thời gian chờ… Các tiến trình trao đổi thông tin với nhau nhằm phối hợp hoạt động cũng như chia sẻ các tài nguyên dùng chung. Dispatcher có nhiệm vụ là lưu ngữ cảnh tiến trình hiện hành và nạp ngữ cảnh tiến trình được chọn kế tiếp. Dispatcher không chọn một tiến trình trong Ready Queue để cấp phát CPU sau đó nạp ngữ cảnh tiến trình được chọn kế tiếp. Preemptive scheduling là cơ chế cho phép một tiến trình mới vào chiếm quyền sử dụng CPU của tiến trình đang xử lý. Điều này làm ngắt tạm thời hoạt động của tiến trình đang xử lý. Trong khi đó, Non-preemptive scheduling, Medium Term Scheduling và Job scheduling có cơ chế hoạt động khác.

Cấu trúc của hệ thống máy tính gồm các phần như hệ điều hành, người sử dụng, phần cứng, các chương trình ứng dụng/chương trình hệ thống. Trong Unix không dùng system call new hay create để tạo mới process. Multi-tasking operating system là hệ thống điều phối bộ vi xử lý theo kiểu time – sharing, chứ không phải là hệ thống quản lý tiến trình theo lô hay hệ thống quản lý làm việc phân tán, hoặc hệ thống quản lý nhiều người dùng (multi user).

So với các chiến lược điều phối FIFO, Lottery và RR thì chiến lược SJF cho thời gian chờ trung bình đạt cực tiểu. Mô hình hệ thống phân tán tương thích dễ dàng với cấu trúc hệ điều hành client – server, chứ không tương thích dễ dàng với các cấu trúc hệ điều hành đơn giản/theo lớp hoặc máy ảo. Một tiến trình ở trạng thái ready khi tiến trình hết lượt sử dụng CPU (hết quantum) chứ không phải khi tiến trình kết thúc hoặc khi đang thực hiện Input/Ouput dữ liệu hoặc khi tiến trình được tạo mới.

Cấu trúc HĐH trong mô hình máy ảo thì bộ nhớ máy tính gồm nhiều tiến trình và nhiều hệ điều hành. Khi hệ điều hành hủy bỏ một tiến trình đã kết thúc xử lý, nó cũng sẽ hủy bỏ khối quản lý của tiến trình và thu hồi các tài nguyên cấp phát cho tiến trình. Bên cạnh đó, nó cũng hủy tiến trình ra khỏi tất cả các danh sách quản lý của hệ thống. Để khắc phục sự "đói CPU" thì sử dụng chiến lược ưu tiên nhiều mức, chứ không dùng SJF không độc quyền hoặc thay đổi độ ưu tiên hoặc RR với quantum nhỏ.

Khi sử dụng máy ảo thì chúng ta cũng sẽ dùng thiết bị ảo, bộ nhớ ảo, CPU ảo. Một tiến trình mới sinh ra thì hệ điều hành sẽ tạo khối quản lý tiến trình (PCB) tương ứng cho nó. Khi chuyển trạng thái tiến trình Running -> Terminated, Running-> Ready, Ready -> Running đều hợp lý. Tuy nhiên, không thể chuyển Ready -> Blocked. Hệ điều hành sẽ thực hiện việc điều phối tiến trình khi tiến trình: Running->Waiting; Running->Ready; Waiting->Ready; tiến trình kết thúc; tiến trình có độ ưu cao hơn xuất hiện**.** Dịch vụ

quản lý tiến trình của Hệ Điều Hành không thực hiện cấp phát và thu hồi một vùng nhớ cho tiến trình. Tiến trình yêu cầu một tài nguyên nhưng chưa được đáp ứng vì tài nguyên chưa sẵn sàng, hoặc chờ một sự kiện hay thao tác nhập/xuất thì tiến trình đó chuyển trạng thái Running->Waiting. Tiến trình không chuyển Running->Ready hay Ready->Waiting. Hệ điều hành MS-DOS thuộc kiến trúc đơn giản trong khi hệ điều hành Linux và OS-X theo kiến trúc client-server.