TRƯỜNG ĐẠI HỌC THỦY LỢI
KHOA CÔNG NGHỆ THÔNG TIN

NGUYÊN LÝ HỆ ĐIỀU HÀNH

Giảng viên: TS. Đoàn Thị Quế

Bộ môn Mạng và an toàn thông tin

CHƯƠNG 2: TIẾN TRÌNH VÀ LUỒNG

2.1. Các khái niệm liên quan đến tiến trình

2.2. Luồng

2.3. Điều độ CPU

2.1 – Các khái niệm liên quan đến tiến trình

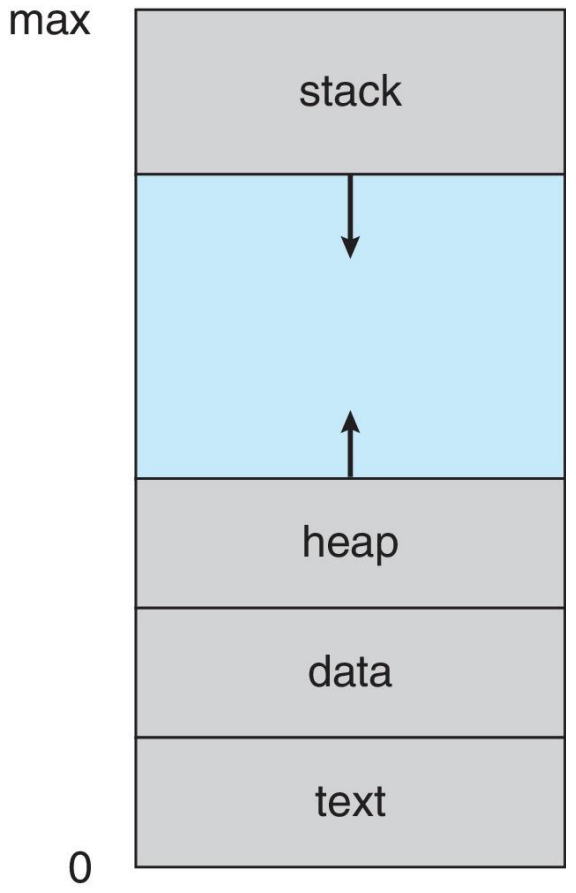
- Tiến trình là gì?
- Mô hình tiến trình
- Trạng thái của tiến trình
- Thông tin mô tả tiến trình
- Khối quản lý, bảng và danh sách tiến trình
- Các thao tác với tiến trình
- Điều độ tiến trình

1. Tiến trình là gì?

- Một tiến trình (Process) là một chương trình đang thi hành.
- Tiến trình được sinh ra khi chương trình được tải vào bộ nhớ để thực hiện.
- Một chương trình có thể bao gồm nhiều tiến trình
- Hệ thống bao gồm tập các tiến trình thực hiện đồng thời
  - Tiến trình hệ thống: Thực hiện mã lệnh hệ thống
  - Tiến trình người dùng: Thực hiện mã lệnh người dùng
- Trách nhiệm của Hệ điều hành:
  - Đảm bảo hoạt động của tiến trình
  - Tạo/xóa tiến trình (người dùng, hệ thống)
  - Điều phối tiến trình
  - Cung cấp cơ chế đồng bộ, truyền thông và ngăn ngừa tình trạng bế tắc giữa các tiến trình

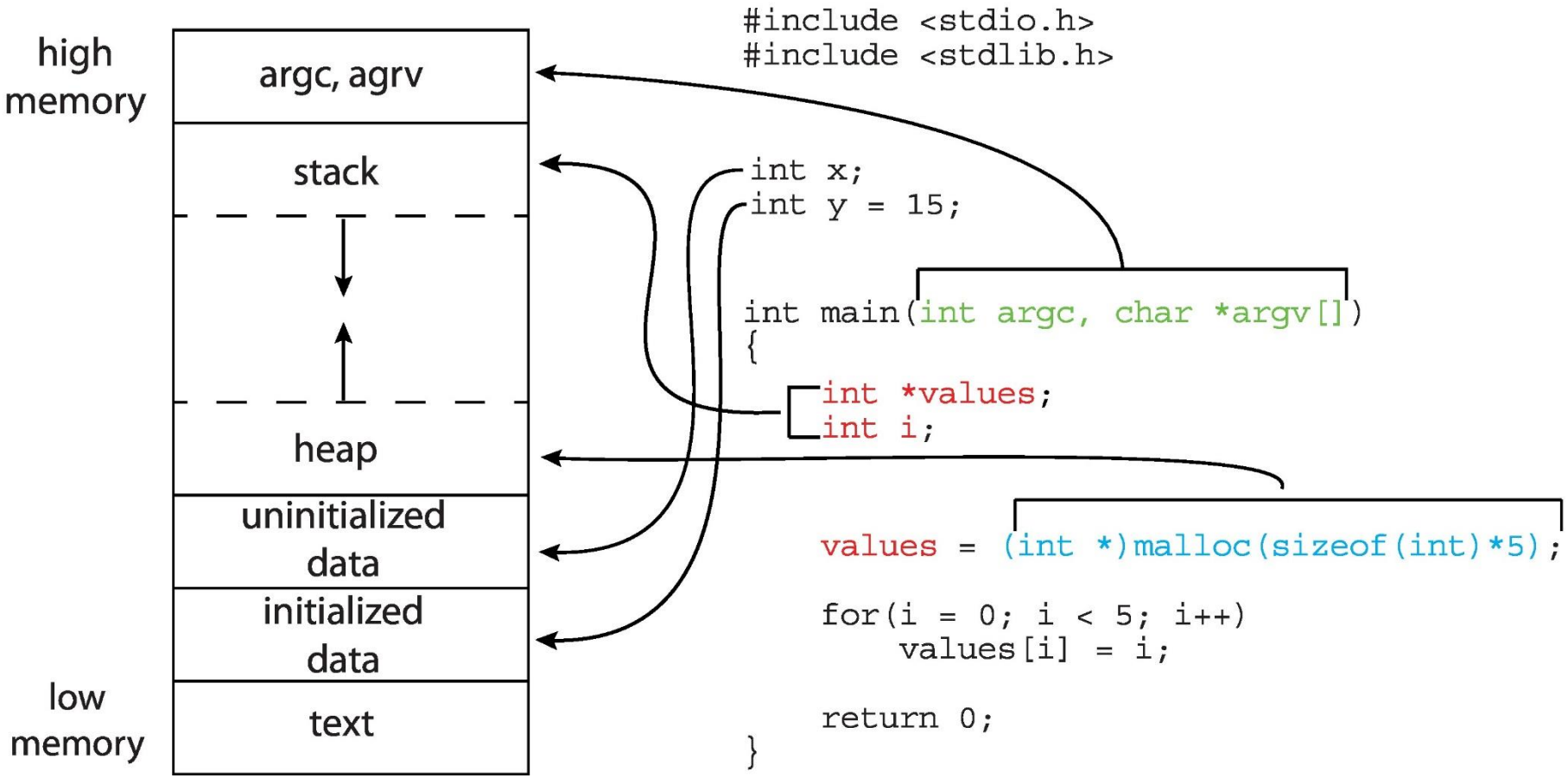
1. Tiến trình là gì?

- Mỗi tiến trình gồm:
  - ***Các lệnh***, tức là các chỉ thị cho CPU thực hiện.
  - Phần ***dữ liệu*** chứa các biến toàn cục.
  - ***Ngăn xếp*** (stack) tiến trình: chứa các dữ liệu tạm thời, ví dụ khi gọi một hàm, các tham số cần thiết để khôi phục lại trạng thái trước khi gọi hàm sẽ được lưu vào ngăn xếp, các tham số cần truyền cho hàm được gọi cũng được thêm vào ngăn xếp. Ngoài ra ngăn xếp còn chứa các biến cục bộ của hàm hoặc phương thức.
  - ***Heap***: đây là vùng bộ nhớ được cấp phát động trong quá trình thực hiện tiến trình, chẳng hạn khi tiến trình thực hiện hàm malloc() của ngôn ngữ C hay new của C++.
  - ***Thông tin về hoạt động hiện thời của tiến trình***: bao gồm nội dung con trỏ lệnh (program counter) chứa lệnh tiếp theo của tiến trình, và nội dung các thanh ghi khác của CPU.



Tiến trình trong bộ nhớ

Bố cục bộ nhớ của một chương trình C



Tiến trình và chương trình

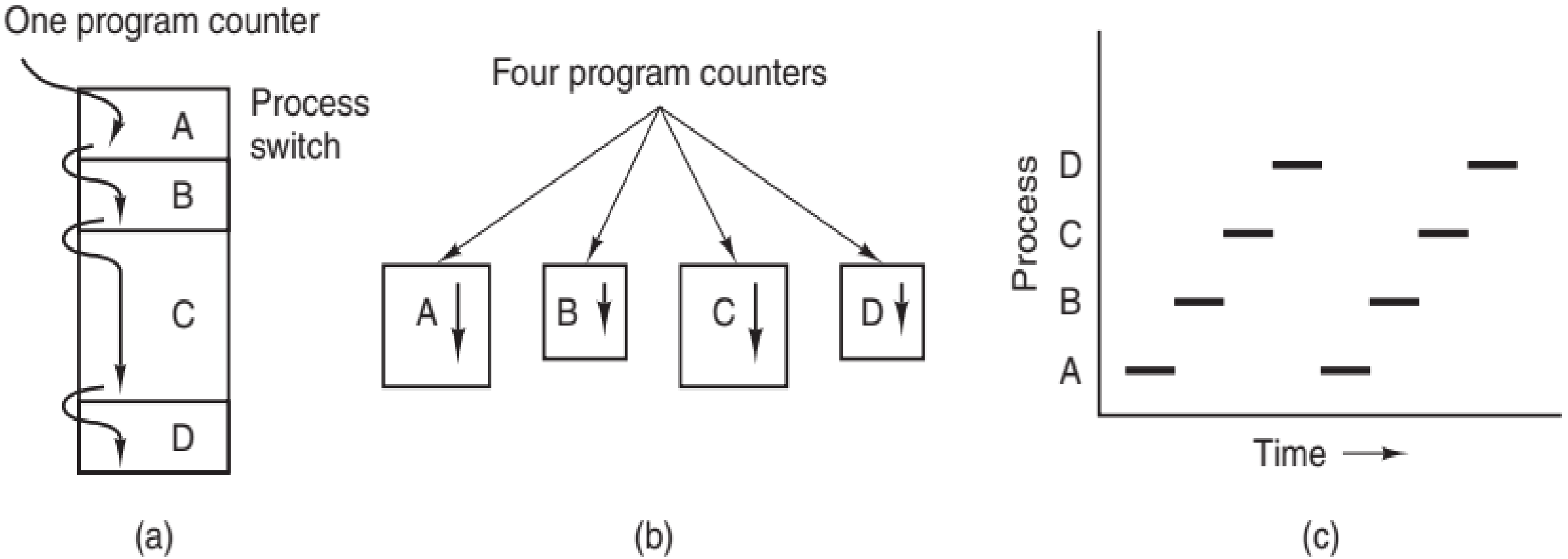
- Hai đặc điểm để phân biệt tiến trình với chương trình:
  - Chương trình là một thực thể tĩnh, tiến trình là thực thể động:
    - Chương trình bao gồm tập lệnh và dữ liệu chứa trong file, file không thay đổi theo thời gian.
    - Tiến trình bao gồm các lệnh, dữ liệu, không gian nhớ, con trỏ lệnh chỉ tới lệnh và các thanh ghi. Hầu hết các thành phần này đều thay đổi trong quá trình tiến trình tồn tại.
  - Chương trình không được cấp tài nguyên cụ thể, tiến trình được cấp tài nguyên (bộ nhớ, các thiết bị vào/ra, thời gian sử dụng CPU).

2. Mô hình tiến trình

- **Thực hiện chương trình trong máy tính:**
  - **Thực hiện tuần tự:** Tại một thời điểm chỉ có thể thực hiện một chương trình, khi chương trình đó thực hiện xong thì mới chạy được chương trình khác
  - **Thực hiện song song:** Có thể chạy nhiều chương trình cùng một lúc (hệ thống đa xử lý)

- **Thực hiện chương trình trong hệ thống đa chương trình:**
  - Tại một thời điểm CPU chỉ chạy được một tiến trình, sau đó sẽ chuyển CPU cho tiến trình khác
  - Mỗi tiến trình sẽ được chạy trong khoảng thời gian vài chục hay vài trăm mili giây. Trong một giây, CPU thực hiện được vài tiến trình → *Tạo ra cảm giác các chương trình chạy song song (giả song song)*
  - Việc quản lý các hoạt động song song có nhiều khó khăn.

➔ *Cần xây dựng mô hình khái niệm (về tiến trình) để quản lý hoạt động song song một cách dễ dàng hơn*



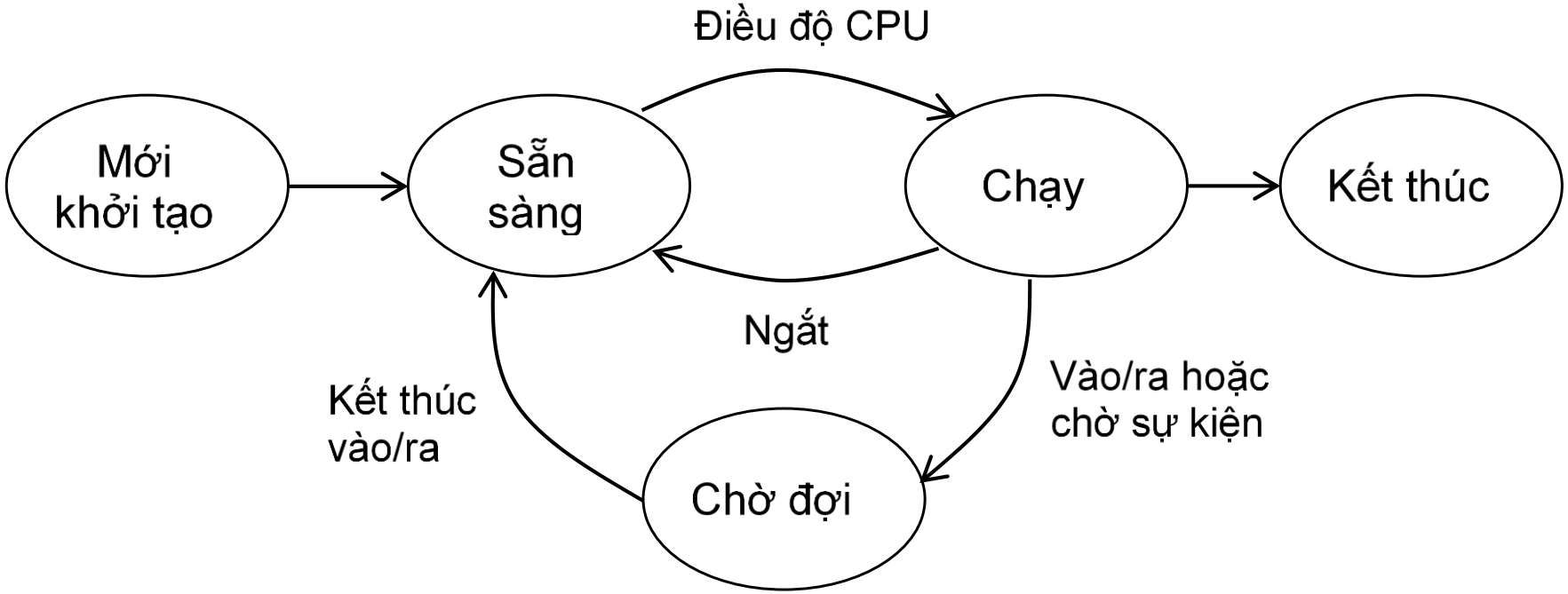
(a) Thực hiện đa chương trình với bốn tiến trình đồng thời.
(b) Mô hình khái niệm của bốn tiến trình tuần tự, độc lập.
(c) Chỉ có một tiến trình hoạt động tại một thời điểm.

3. Trạng thái của tiến trình

Khi thực hiện, tiến trình có thể ở một trong các trạng thái sau:

- **Khởi tạo** (New)**:** tiến trình đang được tạo ra
- **Sẵn sàng** (Ready)**:** Tiến trình đã được phân phối đầy đủ mọi tài nguyên, chỉ đợi CPU
- **Chạy** (Running)**:** Các câu lệnh của tiến trình đang được thực hiện
- **Chờ đợi** (Waiting): Tiến trình đang chờ đợi một sự kiện nào đó (hoàn thành thao tác vào/ra)
- **Kết thúc:** tiến trình thực hiện xong (không còn nằm trong danh sách các tiến trình được thực hiện)

Quá trình chuyển đổi trạng thái



Sơ đồ chuyển đổi giữa các trạng thái của tiền trình

4. Thông tin mô tả tiến trình

- *Làm cách nào để phân biệt được tiến trình này với tiến trình khác trong hệ thống và quản lý việc thực hiện các tiến trình đó?*

➔*Hệ điều hành cần có các thông tin mô tả tiến trình.*

4. Thông tin mô tả tiến trình

Thông tin mô tả tiến trình gồm:

- **Số định danh của tiến trình** (PID-Process Identification/ Process number)
  - Cho phép phân biệt với tiến trình khác.
- **Trạng thái tiến trình (**Process state)
  - Một trong năm trạng thái liệt kê ở phần trước.
- **Thanh ghi con trỏ lệnh (Program counter)**
  - Trỏ tới lệnh tiếp theo cần thực hiện

4. Thông tin mô tả tiến trình

Thông tin mô tả tiến trình gồm (tiếp):

- **Nội dung một số thanh ghi CPU (Registers)**
  - Con trỏ ngăn xếp, thanh ghi dữ liệu, thanh ghi trạng thái, …
- **Thông tin về bộ nhớ của tiến trình (Memory-management information)**
  - Chỉ ra nơi chứa tiến trình
- **Thông tin phục vụ việc điều độ tiến trình (CPU-scheduling information)**
  - Mức độ ưu tiên của tiến trình, vị trí tiến trình trong các hàng đợi, tài nguyên tiến trình đang sở hữu.

4. Thông tin mô tả tiến trình

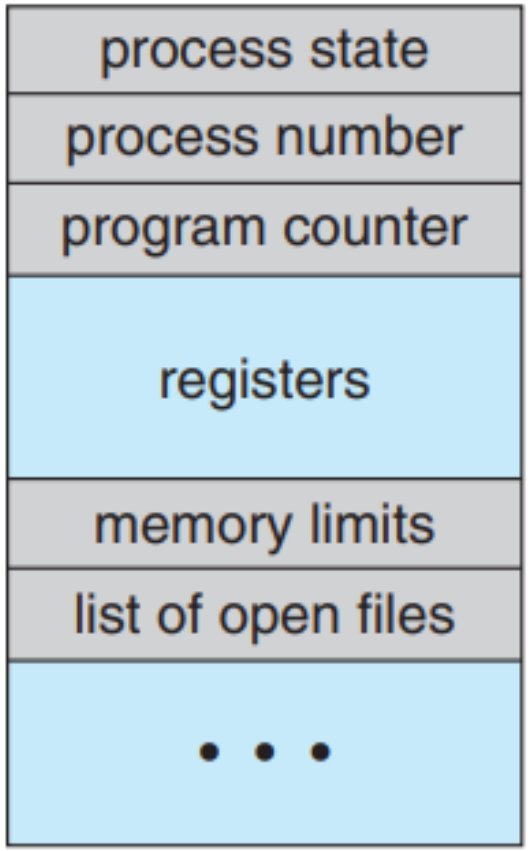
Thông tin mô tả tiến trình gồm (tiếp):

- **Danh sách các tài nguyên khác**
  - Bao gồm danh sách các file đang mở của tiến trình, các thiết bị vào ra tiến trình đang sử dụng.
- **Thông tin thống kê phục vụ quản lý (Accounting information)**
  - Thông tin loại này thường được sử dụng phục vụ thống kê hoặc tính toán chi phí đối với các hệ thống dùng chung (như khi đi thuê máy tính) và bao gồm thông tin về thời gian sử dụng CPU, giới hạn thời gian, tài khoản của người sở hữu tiến trình .v.v.

5. Khối quản lý, bảng và danh sách tiến trình

Thông tin mô tả tiến trình được lưu như thế nào?

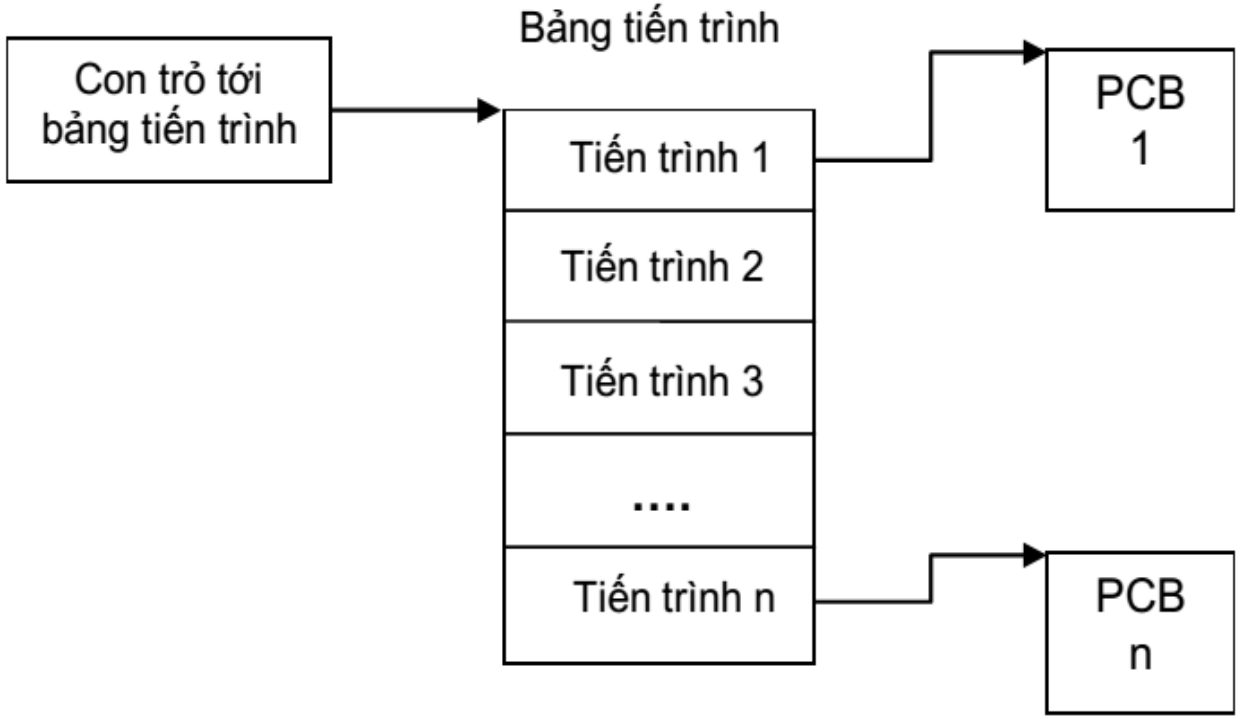
- Thông tin về mỗi tiến trình được lưu trong khối quản lý tiến trình, viết tắt là PCB (Process Control Block)
- **Khối quản lý tiến trình** (*PCB-Process Control Block*):
  - Là một **cấu trúc dữ liệu chứa các thông tin mô tả tiến trình**, được lưu trong bộ nhớ trong và có thể nằm ở các vị trí khác nhau.



Process control block (PCB)

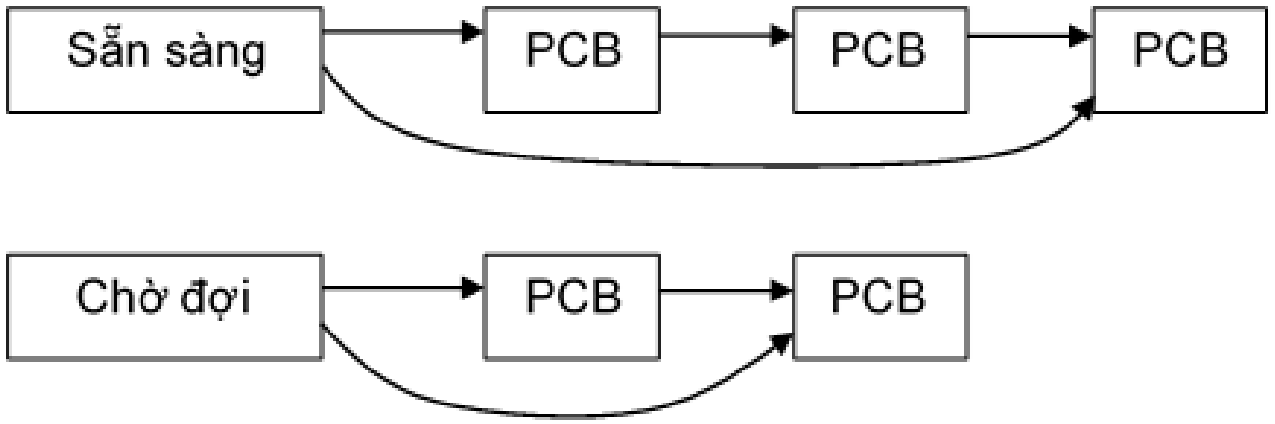
5. Khối quản lý, bảng và danh sách tiến trình

- Hệ điều hành cần lưu các PCB và có cách xác định các PCB
  - Hệ điều hành sử dụng **Bảng tiến trình (***process table***)** chứa con trỏ tới PCB của toàn bộ tiến trình có trong hệ thống.



Bảng tiến trình chứa con trỏ tới các PCB

- **Danh sách tiến trình:**
  - Bao gồm các tiến trình có cùng trạng thái
  - Ví dụ:
    - Danh sách sẵn sàng: gồm các tiến trình đang ở trạng thái sẵn sàng
    - Danh sách chờ đợi: gồm các tiến trình đang ở trạng thái chờ đợi



Danh sách liên kết PCB thuộc các trạng thái khác nhau

6. Các thao tác với tiến trình

- Tạo tiến trình
- Kết thúc tiến trình

Tạo tiến trình

- **Các tiến trình được tạo ra trong các tình huống sau:**
  - Khởi động hệ thống
  - Người dùng kích hoạt một chương trình
  - Một tiến trình đang chạy gọi hàm hệ thống để tạo tiến trình mới.
    - Unix: **fork()**
    - Windows: **CreateProcess()**
  - Khởi tạo một nhóm công việc (trên hệ thống nhóm)

Hệ điều hành thực hiện một số bước khi tạo tiến trình

- ***Tạo không gian nhớ cho tiến trình và PCB***. Kích thước không gian nhớ được tính toán dựa trên thông tin về tiến trình mà hệ điều hành có.
- ***Gán số định danh cho tiến trình*** được tạo mới.
- ***Khởi tạo PCB***. Hệ điều hành gán giá trị cho các thành phần của PCB. Đa số giá trị ban đầu được gán theo mặc định (ví dụ giá trị không), trừ số định danh tiến trình, con trỏ lệnh, con trỏ ngăn xếp và một số giá trị khác.
- ***Liên kết các PCB của các tiến trình*** vào các danh sách quản lý, ví dụ danh sách tiến trình mới khởi tạo, đặt con trỏ trong bảng tiến trình trỏ tới PCB.

Kết thúc tiến trình

Tiến trình có thể kết thúc trong một số trường hợp sau:

- Xử lý xong lệnh cuối cùng hay gọi lệnh kết thúc
  - Unix: **Exit()**
  - Windows: **ExitProcess()**
- Bị tiến trình cha kết thúc
- Do các lỗi
- Tiến trình yêu cầu nhiều bộ nhớ hơn so với dung lượng bộ nhớ hệ thống có thể cung cấp.
- Tiến trình thực hiện lâu hơn thời gian giới hạn.

7. Điều độ tiến trình

- Điều độ tiến trình là gì?
- Tại sao phải điều độ tiến trình?
- Một số khái niệm
- Yêu cầu điều độ
- Hàng đợi điều độ (Scheduling Queues)
- Chuyển đổi ngữ cảnh (Context Switch)

Điều độ tiến trình là gì?

- **Điều độ tiến trình** *(scheduling)* (theo nghĩa tổng quát) là sự tổ chức thực hiện các tiến trình theo một giải thuật điều độ (*scheduling algorithm*) nào đó, nhằm tránh sự xung đột giữa các tiến trình về mặt tài nguyên.

- **Điều độ tiến trình** (theo nghĩa hẹp) là sự tổ chức, phân phối tài nguyên CPU cho các tiến trình theo một giải thuật nào đó, nhằm đảm bảo tốc độ thực hiện của các tiến trình và hiệu quả sử dụng CPU.

Tại sao phải thực hiện điều độ tiến trình?

- Mục tiêu của đa chương trình (multiprogram) là luôn có một số tiến trình chạy để tối đa hóa việc sử dụng CPU.
- Mục tiêu của chia sẻ thời gian (time sharing) là chuyển đổi CPU giữa các tiến trình thường xuyên sao cho người dùng có thể tương tác với từng chương trình khi nó đang chạy.
- Để đáp ứng các mục tiêu này, bộ điều độ tiến trình (process scheduler) chọn một tiến trình khả dụng (từ một tập hợp các tiến trình khả dụng) để thực hiện chương trình trên CPU.

Một số khái niệm

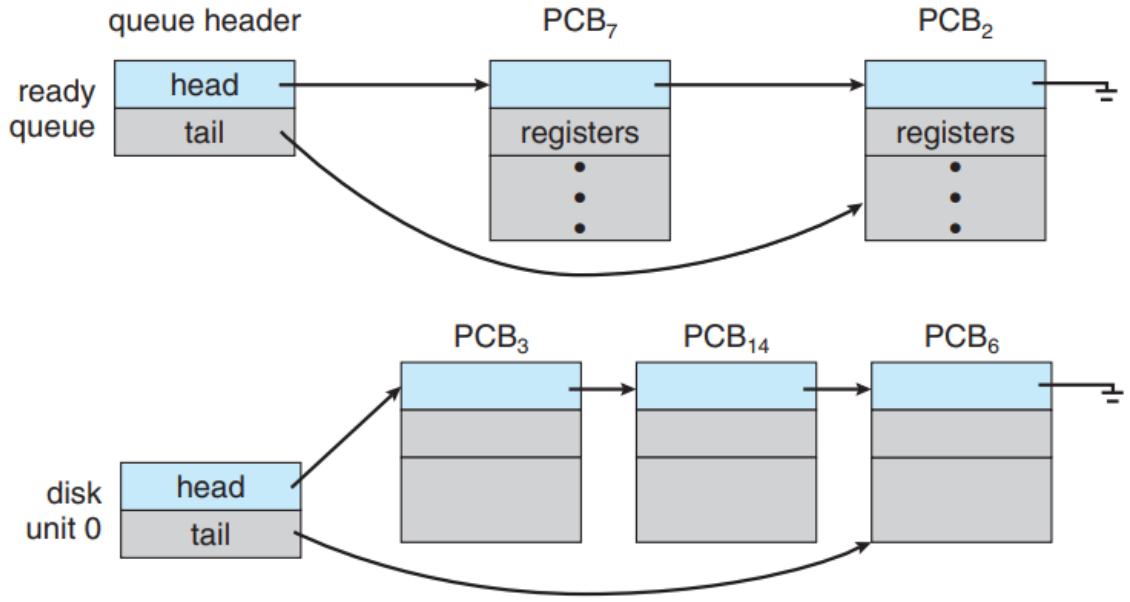
- Việc lựa chọn một trong số các tiến trình đang sẵn sàng và cung cấp CPU cho nó được gọi là điều độ tiến trình (**scheduling**), gọi tắt là điều độ.
- Bộ phận thực hiện việc lựa chọn tiến trình gọi là bộ điều độ tiến trình (**process scheduler**).
- Thuật toán để lựa chọn tiến trình được gọi là thuật toán điều độ (**scheduling algorithm**)

Yêu cầu điều độ

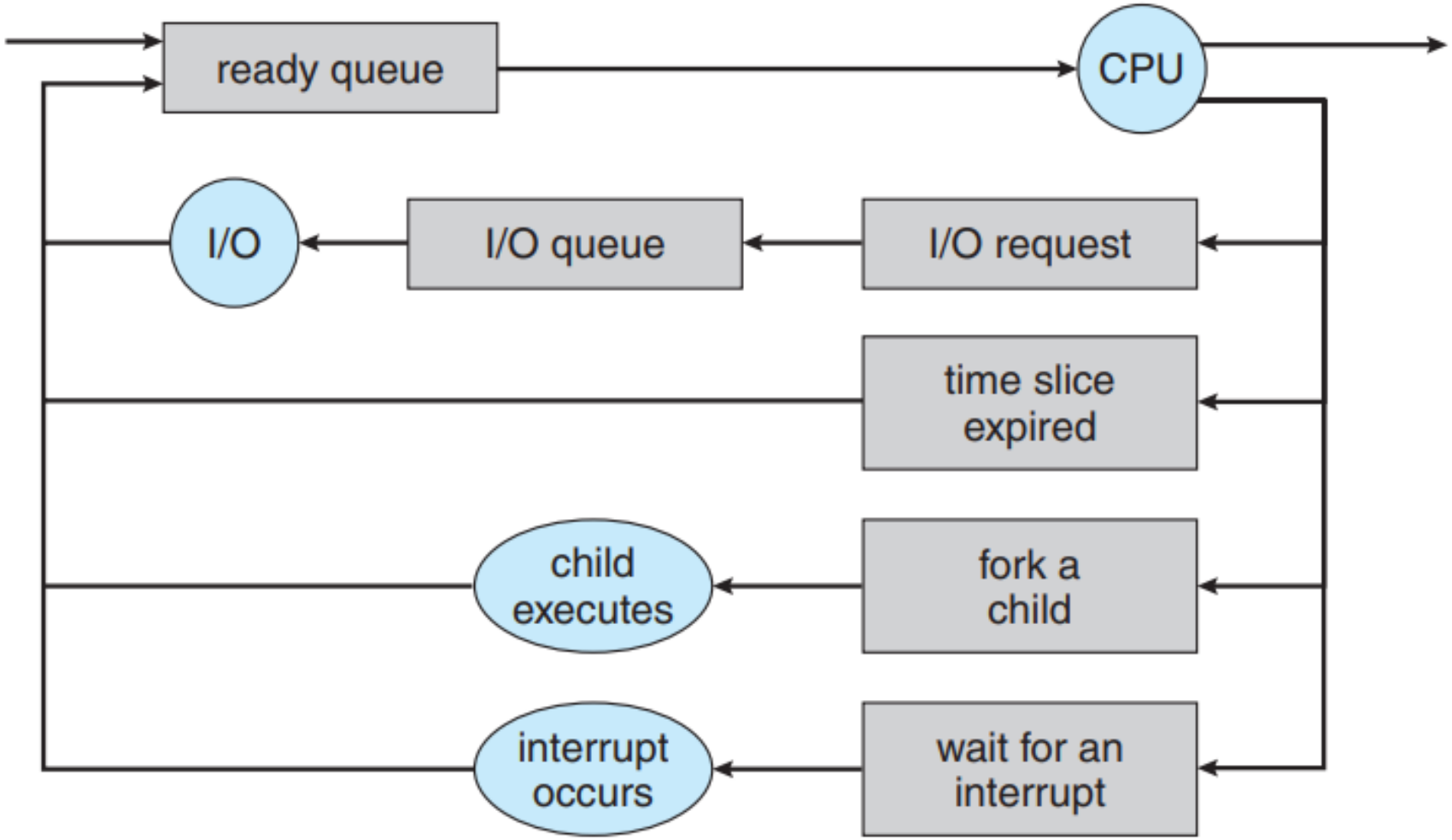
- Tài nguyên găng
  - Tài nguyên găng là loại tài nguyên mà tại một thời điểm chỉ có thể phục vụ cho một đối tượng
    - Ví dụ: CPU, một ô nhớ, máy in, ...
- Đoạn găng: là đoạn chương trình sử dụng tài nguyên găng
- Công việc điều độ phải thoả mãn các yêu cầu sau:
  1) Mỗi thời điểm chỉ có một tiến trình nằm trong đoạn găng
  2) Không tiến trình nào được phép ở lâu vô hạn trong đoạn găng
  3) Không tiến trình nào phải chờ vô hạn trước đoạn găng

Hàng đợi điều độ (Scheduling Queues)

- **Hàng đợi công việc (job queue):** Khi các tiến trình được đưa vào hệ thống, chúng được đặt vào hàng đợi công việc. Hàng đợi công việc chứa tất cả tiến trình trong hệ thống.
- **Hàng đợi sẵn sàng (ready queue):** Các tiến trình đang nằm trong bộ nhớ chính và sẵn sàng chờ để thực thi được giữ trên một danh sách được gọi là hàng đợi sẵn sàng.
- **Hàng đợi thiết bị (device queue)**



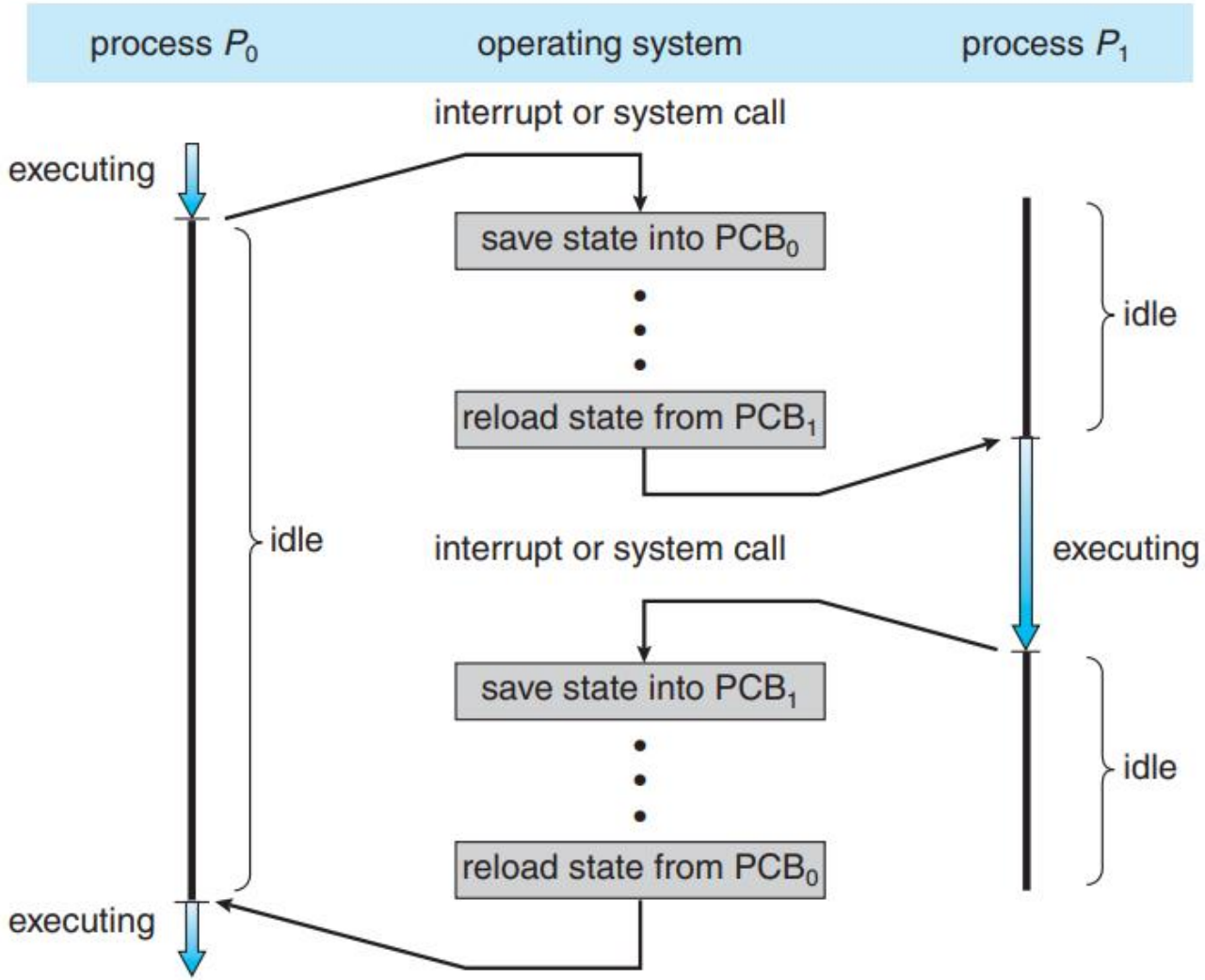
Sơ đồ hàng đợi của điều độ tiến trình



Chuyển đổi ngữ cảnh (Context Switch)

- Quá trình chuyển CPU tới một tiến trình khác gọi là chuyển ngữ cảnh (context switch).
- Ngữ cảnh của tiến trình được mô tả trong trong PCB của tiến trình
- Quá trình chuyển ngữ cảnh cần:
  - Ghi lại trạng thái của tiến trình hiện thời vào PCB của tiến trình hiện thời
  - Nạp trạng thái từ PCB của tiến trình chuyển đến

CPU chuyển từ tiến trình này sang tiến trình khác



CHƯƠNG 2: TIẾN TRÌNH VÀ LUỒNG

2.1. Các khái niệm liên quan đến tiến trình

2.2. Luồng

2.3. Điều độ CPU

2.2. Luồng

1. Luồng là gì?

2. Tài nguyên của tiến trình và luồng

3. Ưu điểm của mô hình đa luồng

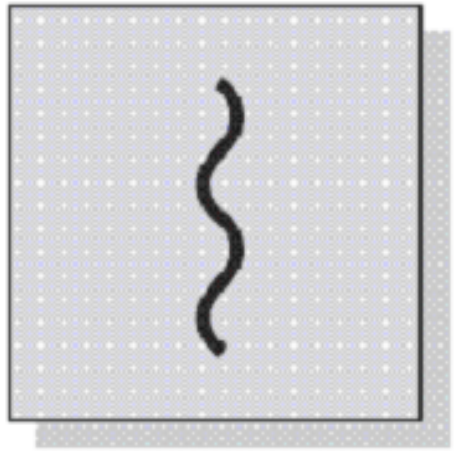
1. Luồng là gì?

- **Tiến trình có thể được xem xét từ hai khía cạnh:**
  - **Tiến trình là một đơn vị sở hữu tài nguyên:**
    - Tiến trình được cấp bộ nhớ để chứa mã lệnh và dữ liệu, cũng như các tài nguyên khác.
  - **Tiến trình là một đơn vị thực hiện công việc tính toán xử lý:**
    - Tiến trình được cấp CPU để thực hiện các lệnh của mình

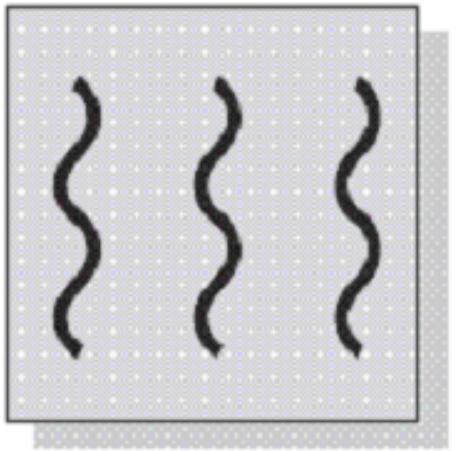
1. Luồng là gì?

- Hệ điều hành trước đây, mỗi tiến trình chỉ tương ứng với một đơn vị xử lý duy nhất:
  - các lệnh được thực hiện một cách tuần tự
  - mỗi tiến trình chỉ thực hiện được một công việc ở một thời điểm
- Hệ điều hành hiện đại, mỗi tiến trình có nhiều đơn vị xử lý:
  - Mỗi đơn vị thực hiện của tiến trình là một **chuỗi các lệnh** được cấp phát CPU để **thực hiện độc lập** (song song), được gọi là **luồng**

Minh họa khái niệm luồng trong tiến trình



tiến trình gồm một luồng



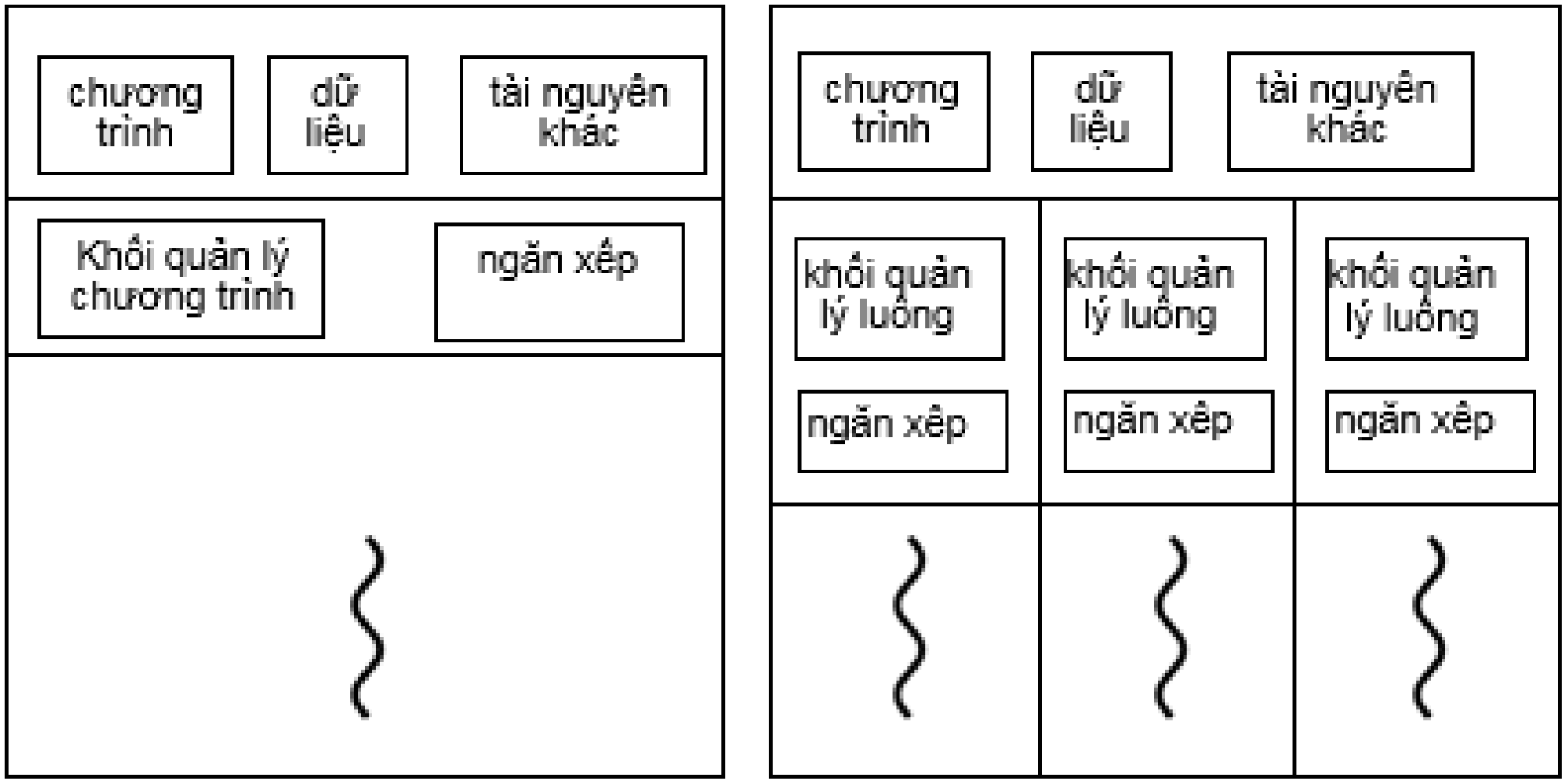
tiến trình gồm nhiều luồng

 = chuỗi lệnh

Tiến trình và luồng

2. Tài nguyên của tiến trình và luồng

- Trong hệ thống cho phép đa luồng, tiến trình vẫn là đơn vị được hệ điều hành sử dụng để phân phối tài nguyên.
- Mỗi tiến trình và tất cả các luồng thuộc tiến trình đó sẽ sở hữu chung một số số tài nguyên bao gồm:
  - Không gian nhớ của tiến trình. Đây là không gian nhớ lôgic, có thể là không gian nhớ ảo, được sử dụng để chứa phần chương trình (các lệnh), phần dữ liệu của tiến trình.
  - Các tài nguyên khác như file do tiến trình mở, thiết bị hoặc cổng vào/ra.



Mô hình đơn luồng và đa luồng

Ưu điểm của mô hình đa luồng

- Tăng hiệu năng và tiết kiệm thời gian.
- Dễ dàng chia sẻ tài nguyên và thông tin.
- Tăng tính đáp ứng.
- Tận dụng được kiến trúc xử lý với nhiều CPU.
- Thuận lợi cho việc tổ chức chương trình.

CHƯƠNG 2: TIẾN TRÌNH VÀ LUỒNG

2.1. Các khái niệm liên quan đến tiến trình

2.2. Luồng

2.3. Điều độ CPU

2.3. Điều độ CPU

- Các khái niệm cơ bản
- Các tiêu chuẩn điều độ
- Một số thuật toán điều độ

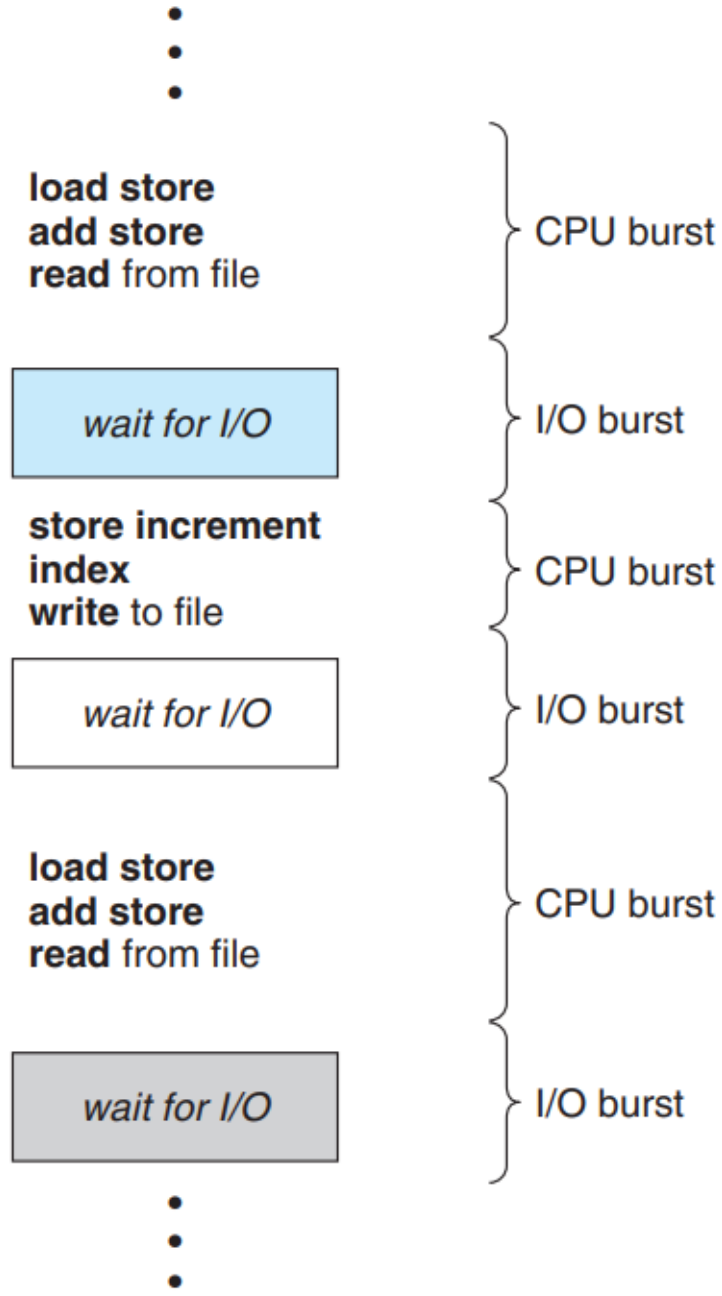
1. Các khái niệm cơ bản

- Chu kỳ CPU-I/O
- Bộ điều độ CPU
- Điều độ độc quyền và điều độ không độc quyền
- Bộ điều phối (Dispatcher)

Chu kỳ CPU-I/O

- Sự thành công của việc lập lịch CPU phụ thuộc vào thuộc tính sau đây của tiến trình: Việc thực thi tiến trình chứa một chu kỳ thực thi CPU và chờ đợi nhập/xuất. Các tiến trình chuyển đổi giữa hai trạng thái này.
- Sự thực thi tiến trình bắt đầu với một chu kỳ CPU (CPU burst), theo sau bởi một chu kỳ nhập/xuất (I/O burst), sau đó một chu kỳ CPU khác, sau đó lại tới một chu kỳ nhập/xuất khác khác,..Sau cùng, chu kỳ CPU cuối cùng sẽ kết thúc với một yêu cầu hệ thống để kết thúc việc thực thi.
- Một chương trình hướng nhập/xuất (I/O-bound) thường có nhiều chu kỳ CPU ngắn. Một chương trình hướng xử lý (CPU- bound) có thể có một hoặc nhiều chu kỳ CPU dài. Sự phân bổ này có thể giúp chúng ta chọn giải thuật lập lịch CPU hợp lý.

Chu kỳ CPU-I/O

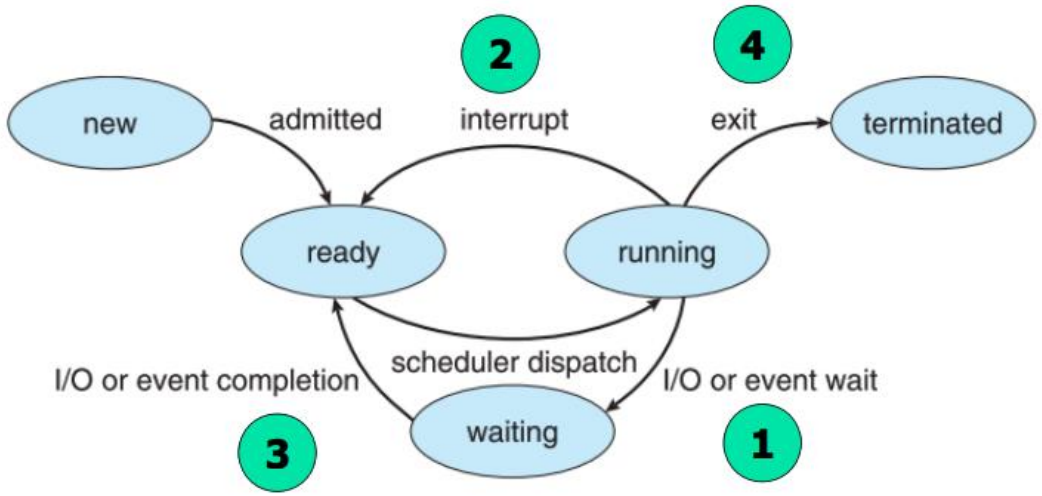

load store
add store
read from file
wait for I/O
store increment
index
write to file
wait for I/O
load store
add store
read from file
wait for I/O
CPU burst
I/O burst
CPU burst
I/O burst
CPU burst
I/O burst

Bộ điều độ CPU

- Điều độ CPU là chọn một tiến trình từ các tiến trình trong hàng đợi sẵn sàng để cấp phát CPU cho tiến trình đó.
- Quá trình lựa chọn tiến trình được thực hiện bởi bộ điều độ ngắn hạn (short-term scheduler) hoặc bộ điều độ CPU (CPU scheduler).

Các trường hợp điều độ CPU có thể xảy ra

- Các quyết định điều độ CPU có thể diễn ra trong bốn trường hợp sau:
  1. Khi một tiến trình chuyển từ trạng thái đang chạy sang trạng thái chờ (ví dụ: do yêu cầu I/O hoặc lệnh gọi wait() để chấm dứt một tiến trình con)
  2. Khi một tiến trình chuyển từ trạng thái đang chạy sang trạng thái sẵn sàng (ví dụ: khi xảy ra ngắt)
  3. Khi một tiến trình chuyển từ trạng thái chờ sang trạng thái sẵn sàng (ví dụ: khi hoàn thành I/O)
  4. Khi một tiến trình kết thúc

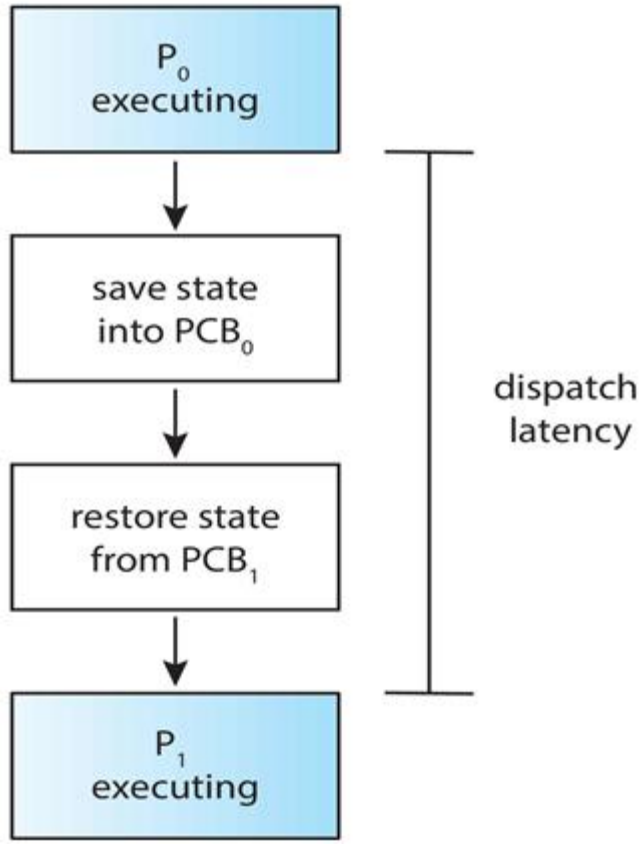


Điều độ độc quyền và điều độ không độc quyền

- Điều độ độc quyền (nonpreemptive scheduling): Khi CPU đã được phân bổ cho một tiến trình, tiến trình đó sẽ giữ CPU cho đến khi nó giải phóng CPU bằng cách kết thúc (trường hợp 1) hoặc bằng cách chuyển sang trạng thái chờ (trường hợp 4).
- Điều độ không độc quyền (preemptive scheduling): Khi một tiến trình được trao CPU thì CPU có thể bị lấy đi khỏi tiến trình đó.

Bộ điều phối (Dispatcher)

- Một thành phần khác liên quan đến chức năng điều độ CPU là bộ điều phối (dispatcher).
- Bộ điều phối là một mô-đun cung cấp quyền điều khiển CPU cho tiến trình trình được chọn bởi bộ điều độ CPU. Chức năng này liên quan:
  - Chuyển ngữ cảnh
  - Chuyển sang chế độ người dùng
  - Nhảy đến vị trí thích hợp trong chương trình người dùng để khởi động lại chương trình đó
- Bộ điều phối phải nhanh nhất có thể vì nó được gọi trong mỗi lần chuyển đổi tiến trình. Thời gian cần thiết để bộ điều phối dừng một tiến trình này và bắt đầu một tiến trình khác được gọi là độ trễ điều phối (dispatch latency).



2. Các tiêu chuẩn điều độ

- Độ sử dụng CPU (lớn nhất)
- Thông lượng (lớn nhất)
- Thời gian hoàn thành (nhỏ nhất)
- Thời gian chờ (nhỏ nhất)
- Thời gian đáp ứng (nhỏ nhất)

2. Các tiêu chuẩn điều độ

- Độ sử dụng CPU (lớn nhất)
  - Mục đích của điều độ là làm CPU hoạt động nhiều nhất có thể
  - Độ sử dụng CPU thay đổi từ 40% (hệ thống tải nhẹ) đến 90% (hệ thống tải nặng).
- Thông lượng (lớn nhất)
  - Số lượng tiến trình hoàn thành trong một đơn vị thời gian
    - Các tiến trình dài: 1 tiến trình/giờ
    - Các tiến trình ngắn: 10 tiến trình/giây
- Thời gian hoàn thành (nhỏ nhất)
  - Khoảng thời gian từ thời điểm gửi đến hệ thống tới khi quá trình hoàn thành
    - Thời gian chờ đợi để đưa tiến trình vào bộ nhớ
    - Thời gian chờ đợi trong hàng đợi sẵn sàng
    - Thời gian chờ đợi trong hàng đợi thiết bị
    - Thời gian thực hiện thực tế

2. Các tiêu chuẩn điều độ

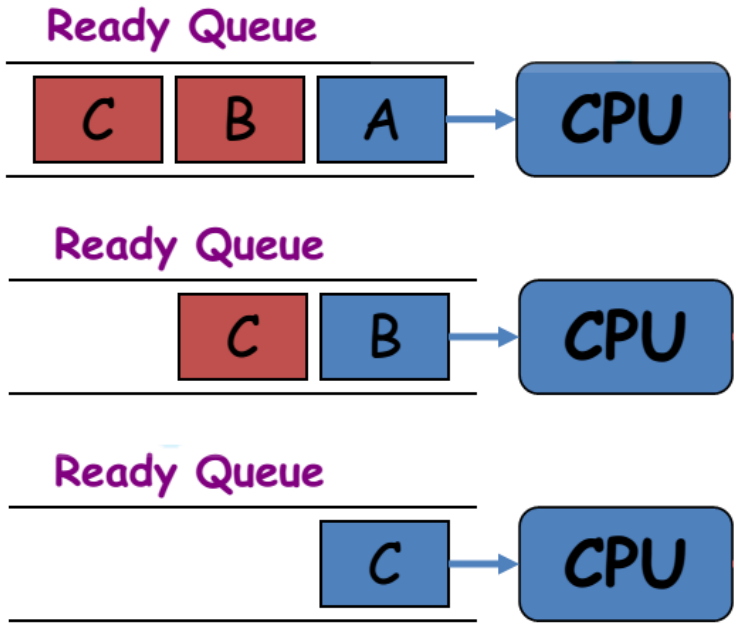
- Thời gian chờ (nhỏ nhất)
  - Thời gian chờ là tổng thời gian chờ trong hàng đợi sẵn sàng
  - Lưu ý: Thuật toán điều độ CPU không ảnh hưởng đến lượng thời gian trong đó một quá trình thực thi trên CPU hoặc thực hiện I/O.
- Thời gian đáp ứng (nhỏ nhất)
  - Khoảng thời gian cần thiết kể từ khi yêu cầu được gửi cho đến khi nhận được phản hồi đầu tiên.

3. Một số thuật toán điều độ

- **FCFS (First-Come First-Served)**
  Đến trước phục vụ trước
- **RR (Round-Robin)**
  Quay vòng
- **SJF (Shortest Job First)**
  Ưu tiên tiến trình ngắn nhất
- **SRTN (Shortest Remaining Time Next)**
  Ưu tiên tiến trình có thời gian còn lại ngắn nhất

FCFS (First-Come First-Served)
Đến trước phục vụ trước

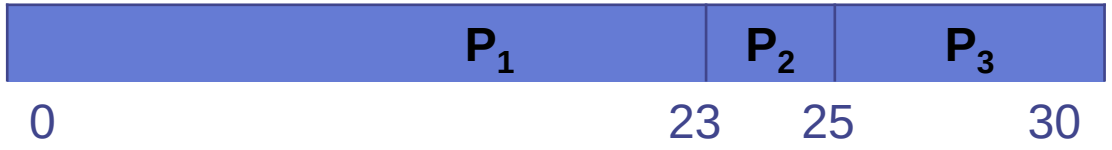
- Tiến trình yêu cầu CPU trước sẽ được cấp CPU trước. Tiến trình độc quyền chiếm dụng CPU.
- Hệ điều hành xếp tiến trình sẵn sàng vào hàng đợi FIFO. Tiến trình mới được xếp vào cuối hàng đợi.
- Hệ điều hành sẽ lấy tiến trình từ đầu hàng đợi và cấp CPU cho tiến trình đó thực hiện.



Ví dụ: FCFS

Tiến trình (Process)	Thời điểm xuất hiện (Arrival Time)	Thời gian sử dụng CPU (Burst Time)
P_1	0	23
P_2	1	2
P_3	2	5

Sơ đồ Gantt cho điều độ:



Thời gian lưu lại hệ thống của một tiến trình = Thời điểm tiến trình kết thúc sử dụng CPU – Thời điểm xuất hiện

Thời gian chờ của tiến trình = Thời gian lưu lại hệ thống của tiến trình – Thời gian tiến trình sử dụng CPU

Tiến trình	Thời gian lưu lại hệ thống	Thời gian chờ (Waiting time)
P_1	23 – 0 = 23	23 – 23 = 0
P_2	25 – 1 = 24	24 – 2 = 22
P_3	30 – 2 = 28	28 – 5 = 23

Thời gian chờ trung bình (Average waiting time)

(0+22+23)/3 =15

Bài tập áp dụng

- Thực hiện điều phối các tiến trình sau theo thuật toán FCFS
  - Vẽ sơ đồ Gantt
  - Tính thời gian chờ trung bình của mỗi tiến trình

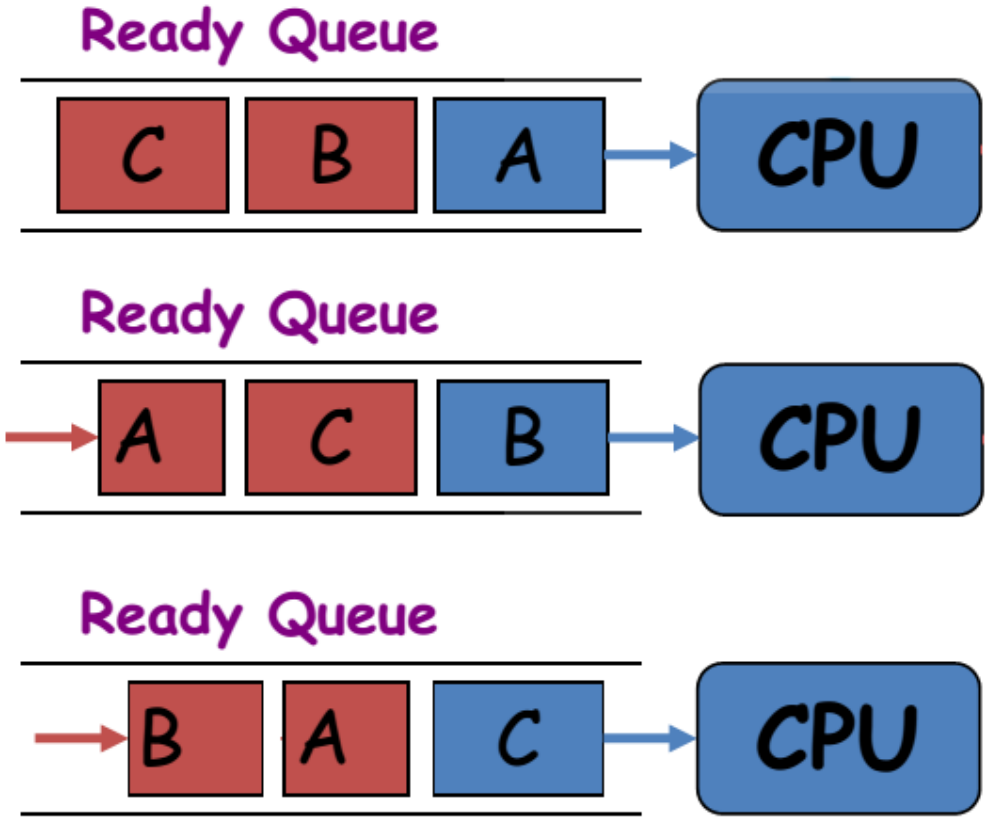
Tiến trình	Thời điểm xuất hiện	Thời gian sử dụng CPU
P_1	0	10
P_2	1	1
P_3	2	2
P_4	3	1
P_5	4	5

Nhận xét FCFS

- Đơn giản, dễ cài đặt
- Chịu đựng hiện tượng tích lũy thời gian chờ
  - Tiến trình có thời gian xử lý ngắn đợi tiến trình có thời gian xử lý dài
- Có tình trạng độc chiếm CPU
- Không phù hợp với hệ thống tương tác

RR (Round-Robin)
Quay vòng

- RR tương tự FCFS nhưng có thể cơ chế phân phối lại bằng cách sử dụng ngắt của đồng hồ.
- Tiến trình sẽ lần lượt được cấp CPU trong những khoảng thời gian q (time quantum/time slice)



A chỉ được chiếm CPU trong khoảng thời gian *q*

B được giao quyền sử dụng CPU trong khoảng thời gian *q* kế tiếp

C được giao quyền sử dụng CPU trong khoảng thời gian *q* kế tiếp

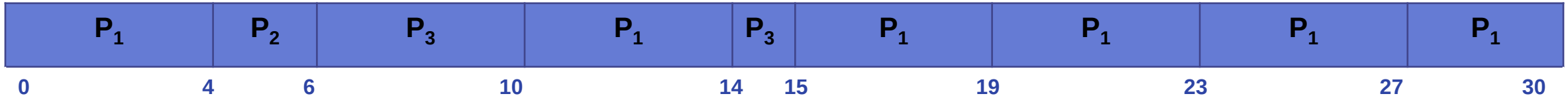
Ví dụ: RR

Tiến trình	Thời điểm xuất hiện	Thời gian sử dụng CPU
P_1	0	23
P_2	1	2
P_3	2	5

Tiến trình	Thời gian lưu lại hệ thống	Thời gian chờ
P_1	30 – 0 = 30	30 – 23 = 7
P_2	6 – 1 = 5	5 – 2 = 3
P_3	15 – 2 = 13	13 – 5 = 8

q = 4

Thời gian chờ trung bình=(7+3+8)/3 = 6



0: P_1 vào RQ
P_1 dùng CPU

1: P_2 vào RQ

2: P_3 vào RQ

4: P_1 hết lượt, xếp vào RQ (sau P_3)
P_2 dùng CPU

6: P_2 kết thúc
P_3 dùng CPU

10: P_3 hết lượt, xếp vào RQ (sau P_1)
P_1 dùng CPU

14: P_1 hết lượt, xếp vào RQ (sau P_3)
P_3 dùng CPU

15: P_3 kết thúc
P_1 dùng CPU

19: P_1 dùng CPU

23: P_1 dùng CPU

27: P_1 dùng CPU

30: P_1 kết thúc

Ví dụ: RR

Tiến trình	Thời điểm xuất hiện	Thời gian sử dụng CPU
P_1	0	23
P_2	1	2
P_3	2	5

q = 2

Thời gian chờ trung bình ≈ 4.66

Bài tập áp dụng

- Vẽ sơ đồ Gantt theo thuật toán RR với q = 2
- Tính thời gian chờ trung bình của mỗi tiến trình

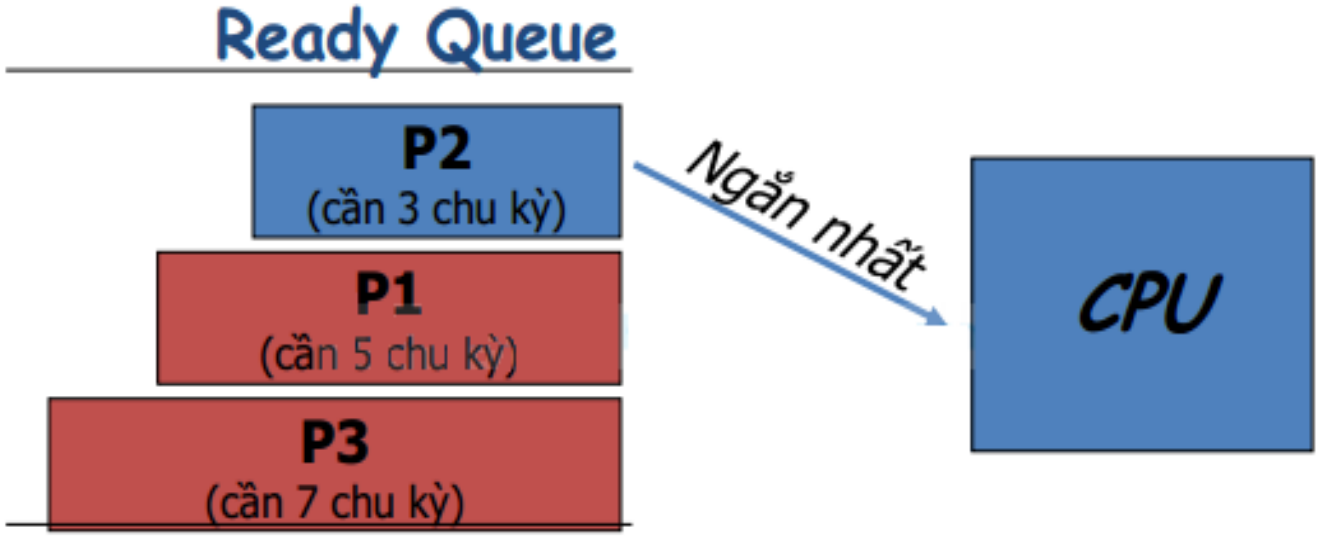
Tiến trình	Thời điểm xuất hiện	Thời gian sử dụng CPU
P_1	0	10
P_2	1	1
P_3	2	2
P_4	3	1
P_5	4	5

Nhận xét RR

- Giảm thời gian chờ đợi trung bình
- Loại bỏ hiện tượng độc chiếm CPU
- Phù hợp với hệ thống tương tác người dùng

SJF (Shortest Job First)
Ưu tiên tiến trình ngắn nhất

- Lựa chọn trong hàng đợi tiến trình có chu kỳ sử dụng CPU ngắn nhất để phân phối CPU.
- Nếu nhiều tiến trình tiếp theo với chu kỳ CPU bằng nhau, tiến trình đứng trước sẽ được chọn.



Ví dụ: SJF

Tiến trình	Thời điểm xuất hiện	Thời gian sử dụng CPU
P_1	0	10
P_2	1	1
P_3	2	2
P_4	3	1
P_5	4	5

Tiến trình	Thời gian lưu lại hệ thống	Thời gian chờ
P_1	10 – 0 = 10	10 – 10 = 0
P_2	11 – 1 = 10	10 – 1 = 9
P_3	14 – 2 = 12	12 – 2 = 10
P_4	12 – 3 = 9	9 - 1 = 8
P_5	19 – 4 = 15	15 – 5 = 10

0: P_1 vào RQ
P_1 dùng CPU

1: P_2 vào RQ
2: P_3 vào RQ
3: P_4 vào RQ
4: P_5 vào RQ

10: P_1 kết thúc
P_2 dùng CPU

11: P_2 kết thúc
P_4 dùng CPU

12: P_4 kết thúc
P_3 dùng CPU

14: P_3 kết thúc
P_5 dùng CPU

19: P_5 kết thúc

Thời gian chờ trung bình:
(0+9+10+8+10)/5 = 7.4

Bài tập áp dụng

- Vẽ sơ đồ Gantt theo thuật toán SJF
- Tính thời gian chờ trung bình

Tiến trình	Thời điểm xuất hiện	Thời gian sử dụng CPU
P_1	0	6
P_2	1	3
P_3	2	4
P_4	3	2

Nhận xét SJF

- Phải biết trước độ dài chu kỳ sử dụng CPU của tiến trình tiếp theo (rất khó)
  - ⇨ Không khả thi

SRTN (Shortest Remaining Time Next)
Ưu tiên tiến trình có thời gian còn lại ngắn nhất

- Ưu tiên tiến trình có thời gian còn lại ngắn nhất
- Khi một tiến trình mới xuất hiện, thời gian thực hiện của nó được so sánh với thời gian thực hiện còn lại của tiến trình đang chạy. Nếu tiến trình mới có thời gian thực hiện ngắn hơn, nó sẽ được chọn để chạy, còn tiến trình đang chạy sẽ bị treo.

Ví dụ: SRTN

Tiến trình	Thời điểm xuất hiện	Thời gian sử dụng CPU
P_1	0	10
P_2	1	1
P_3	2	2
P_4	3	1
P_5	4	5

Tiến trình	Thời gian lưu lại hệ thống	Thời gian chờ
P_1	19 – 0 = 19	19 – 10 = 9
P_2	2 – 1 = 1	1 – 1 = 0
P_3	4 – 2 = 2	2 – 2 = 0
P_4	5 – 3 = 2	2 – 1 = 1
P_5	10 – 4 = 6	6 – 5 = 1

0: P_1 vào RQ
P_1 dùng CPU
1: P_2 vào RQ
P_2 dùng CPU
2: P_3 vào RQ
P_2 kết thúc
P_3 dùng CPU

3: P_4 vào RQ
P_3 vẫn được dùng CPU
4: P_5 vào RQ
P_3 kết thúc
P_4 dùng CPU

5: P_4 kết thúc
P_5 dùng CPU
10: P_5 kết thúc
P_1 dùng CPU
19: P_1 kết thúc

Thời gian chờ trung bình:
(9+0+0+1+1)/5 = 2.2

Bài tập áp dụng

- Vẽ sơ đồ Gantt cho thuật toán SRTN
- Tính thời gian trễ trung bình

Tiến trình	Thời điểm xuất hiện	Thời gian sử dụng CPU
P_1	0	6
P_2	1	3
P_3	2	4
P_4	3	2

Nhận xét SRTN

- Thời gian chờ đợi trung bình nhỏ
- Phải dự đoán được độ dài chu kỳ sử dụng CPU của tiến trình. So với điều độ quay vòng, việc chuyển đổi tiến trình diễn ra ít hơn và do vậy thời gian chuyển đổi ngữ cảnh giảm.

Hết Chương 2