HỆ ĐIỀU HÀNH
(OPERATING SYSTEM)

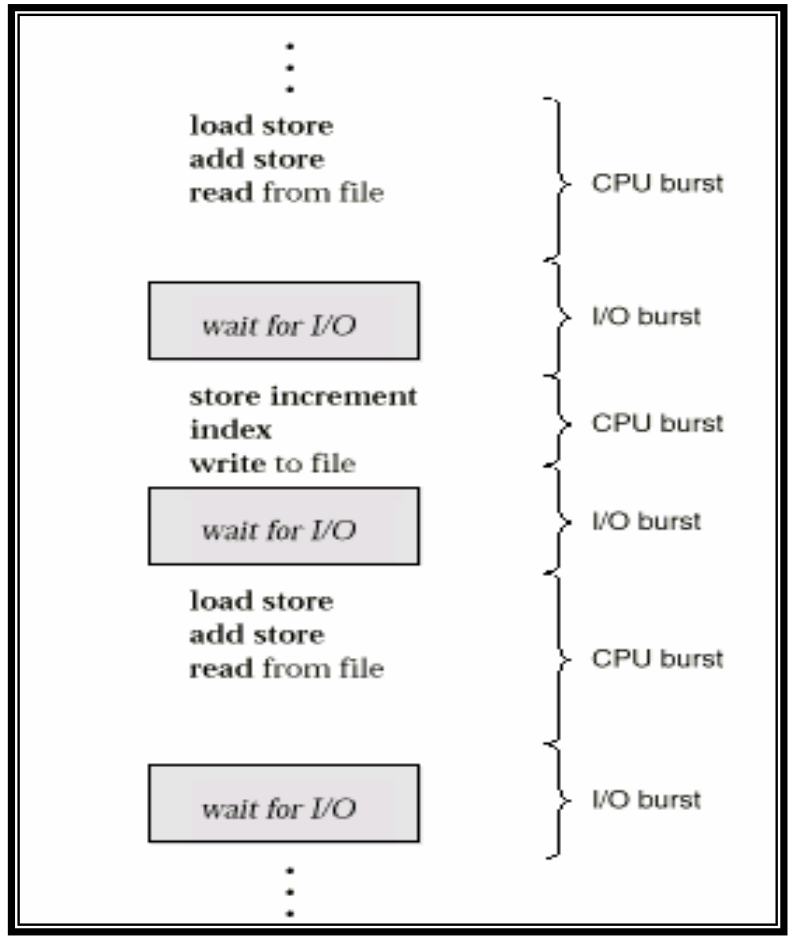
Trình bày:Nguyễn Hoàng Việt
Khoa Công Nghệ Thông Tin
Đại Học Cần Thơ

Chương 4: Định thời CPU
(CPU Scheduling)

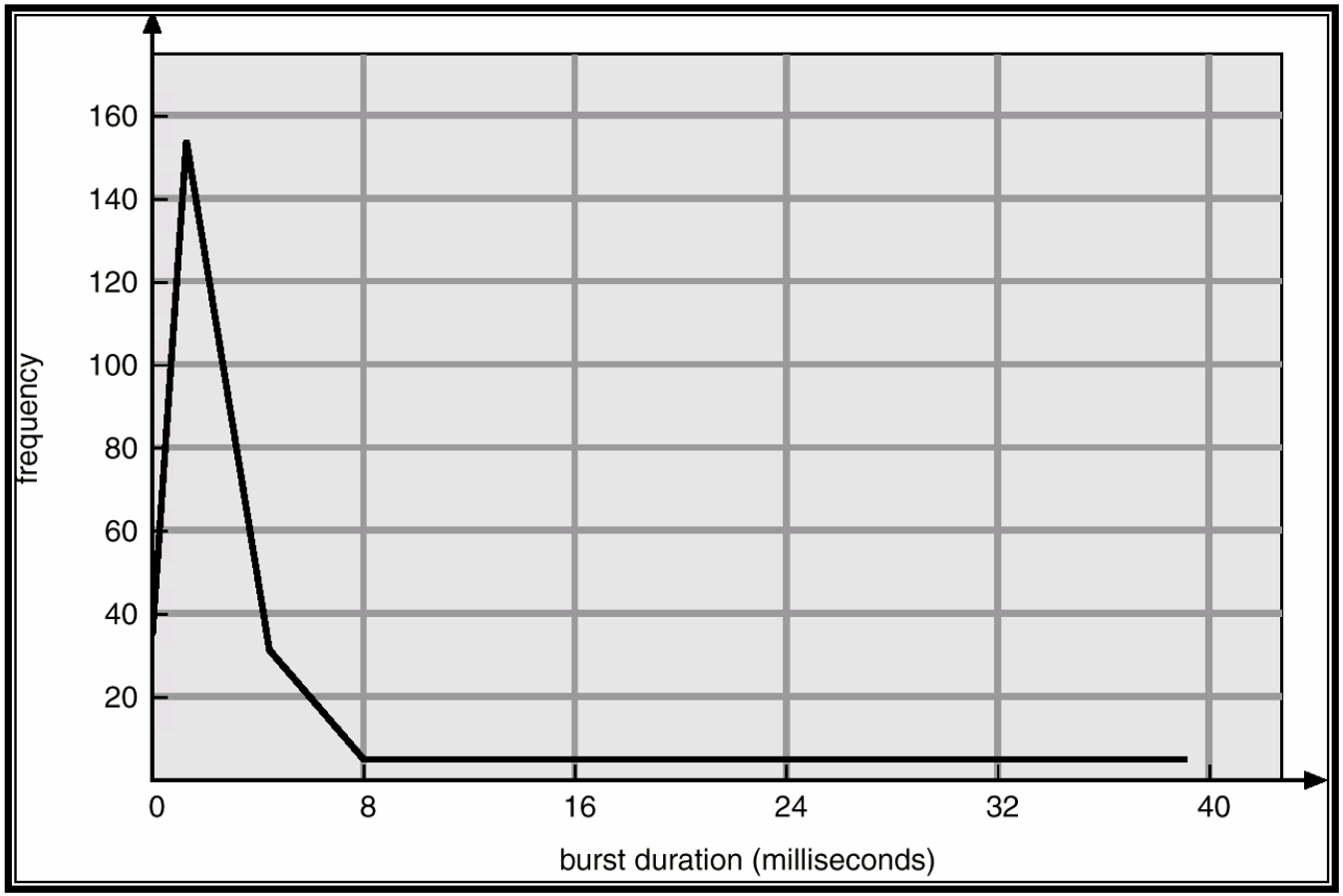
- Các khái niệm cơ bản
- Tiêu chí cho việc định thời
- Các giải thuật định thời
- Định thời đa xử lý
- Định thời thời gian thực
- Đánh giá giải thuật

Các khái niệm cơ bản (1)
Chu kỳ CPU-I/O (CPU-I/O Burst Cycle)

- Kỹ thuật đa chương giúp việc sử dụng CPU đạt hiệu năng cao nhất.
- Chu kỳ CPU-I/O
  - Sự thực thi của quá trình bao gồm nhiều chu kỳ CPU-I/O
  - Một chu kỳ CPU-I/O bao gồm chu kỳ thực thi CPU (CPU burst) và chu kỳ chờ đợi vào/ra (I/O burst).
- Sự phân bổ sử dụng CPU
  - Chương trình hướng nhập xuất (I/O-bound) thường có nhiều chu kỳ CPU ngắn.
  - Chương trình hướng xử lý (CPU-bound) thường có nhiều chu kỳ CPU dài.



Các khái niệm cơ bản (2)
Biểu đồ so sánh thời gian sử dụng CPU



Histogram of CPU-burst Times

Các khái niệm cơ bản (3)
Bộ định thời CPU (CPU Scheduler)

- Được thực hiện bởi bộ định thời ngắn kỳ (short-term scheduler), còn được gọi là bộ định thời (CPU scheduler)
- Chọn một trong các quá trình trong hàng đợi sẵn sàng và giao CPU cho nó thực thi
- Quyết định định thời xảy ra khi một quá trình:
  - Chuyển từ trạng thái đang chạy sang trạng thái chờ đợi
  - Chuyển từ trạng thái đang chạy sang trạng thái sẵn sàng
  - Chuyển từ trạng thái chờ đợi sang sẵn sàng
  - Kết thúc
- Định thời không trưng dụng (nonpreempty): quá trình sẽ giữ CPU và chỉ giải phóng CPU khi nó cần (trường hợp 1 và 4)
- Định thời trưng dụng (preempty): các trường hợp khác

Các khái niệm cơ bản (4)
Bộ phân phối (Dispatcher)

- Có nhiệm vụ trao quyền điều khiển CPU cho quá trình được chọn bởi bộ định thời CPU
- Công việc này bao gồm:
  - Chuyển ngữ cảnh
  - Chuyển sang chế độ người dùng
  - Nhảy tới vị trí của chương trình người dùng để khởi động lại chương trình đó
- Độ trễ phân phối (Dispatch Latency): thời gian dispatcher cần để ngưng một quá trình và khởi động lại quá trình khác

Tiêu chí cho việc định thời biểu (1)
Các tiêu chí (Criteria)

- Hiệu suất sử dụng CPU: giữ CPU luôn bận nhiều nhất có thể.
- Thông lượng (Throughput): số lượng quá trình hoàn thành trên một đơn vị thời gian.
- Thời gian xoay vòng (Turnaround time): tổng thời gian trôi qua từ khi một quá trình được đưa lên đến khi nó hoàn thành, bao gồm: thời gian thực thi, thời gian chờ I/O, thời gian trong hàng đợi sẵn sàng, và các phí tổn khác.
- Thời gian chờ đợi (Waiting time): tổng thời gian trong trong hàng đợi sẵn sàng (ready queue).
- Thời gian đáp ứng (Response time): lượng thời gian từ lúc một yêu cầu được đệ trình cho đến khi tín hiệu trả lời đầu tiên xuất hiện (dùng cho môi trường chia thời gian).

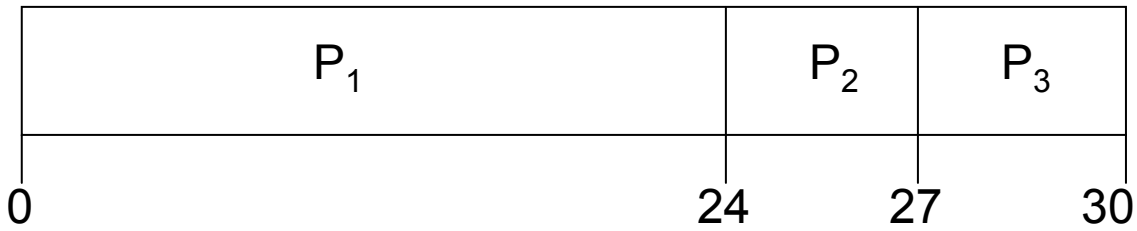
Tiêu chí cho việc định thời biểu (2)
Tiêu chí tối ưu hóa

- Hiệu suất sử dụng CPU tối đa
- Thông lượng tối đa
- Thời gian xoay vòng tối thiểu
- Thời gian chờ đợi tối thiểu
- Thời gian đáp ứng tối thiểu

Các giải thuật định thời (1)
Giải thuật First-Come, First-Served (FCFS)

Process	TG sử dụng CPU
P_1	24
P_2	3
P_3	3

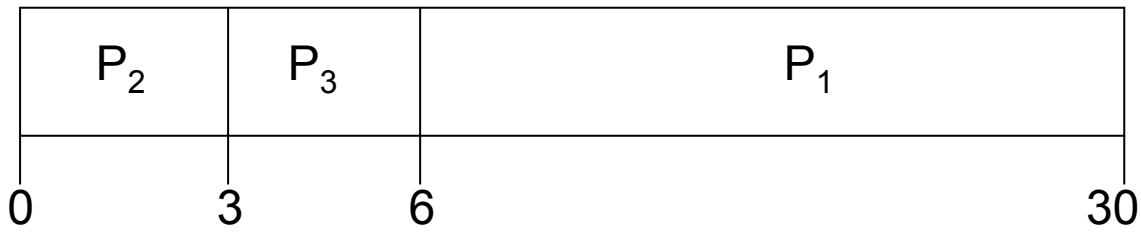
- Giả sử các quá trình xuất hiện theo thứ tự P_1, P_2, P_3. Biểu đồ Gantt cho lịch biểu là:



- Thời gian chờ đợi: $P_1 = 0$; $P_2 = 24$; $P_3 = 27$
- Thời gian chờ đợi trung bình: $(0 + 24 + 27)/3 = 17$

Các giải thuật định thời (2)
Giải thuật First-Come, First-Served (FCFS)

- Giả sử các quá trình xuất hiện theo thứ tự P_2, P_3,, P_1. Biểu đồ Gantt cho lịch biểu là:



- Thời gian chờ đợi: P_1= 6; P_2 = 0; P_3 = 3
- Thời gian chờ đợi trung bình: (6 + 0 + 3)/3 = 3
- Tốt hơn nhiều so với trường hợp trước.

⇨ Tránh "tác động nối đuôi": quá trình ngắn nằm sau quá trình dài.

- FCFS là giải thuật định thời không trưng dụng, không thích hợp cho hệ thống chia thời gian.

Các giải thuật định thời (3)
Giải thuật Shortest-Job-First (SJF)

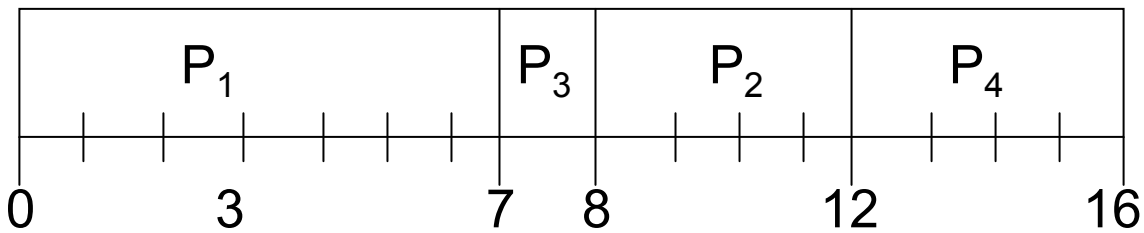
- Kết hợp với mỗi quá trình độ dài thời gian mà nó sẽ sử dụng CPU lần kế tiếp. Sử dụng các độ dài này để định thời cho quá trình với thời gian ngắn nhất.
- Có hai thể thức thực hiện giải thuật:
  - Không trưng dụng: một khi CPU được giao cho một quá trình, nó không thể bị trưng dụng để giao cho quá trình khác có thời gian ngắn hơn cho đến khi quá trình này sử dụng CPU xong.
  - Trưng dụng: nếu một quá trình mới đến có thời gian sử dụng CPU ngắn hơn thời gian thực thi còn lại của quá trình đang chạy, thì ưu tiên giao CPU cho quá trình mới đến. Cách thức này được gọi là Shortest-Remaining-Time-First (SRTF).
- SJF là tối ưu – nó tạo ra thời gian chờ đợi trung bình ngắn nhất.

Các giải thuật định thời (4)
Giải thuật Shortest-Job-First (SJF)

Ví dụ về SJF không trưng dụng

Process	Thời điểm đến	Tg sử dụng CPU
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (không trưng dụng)



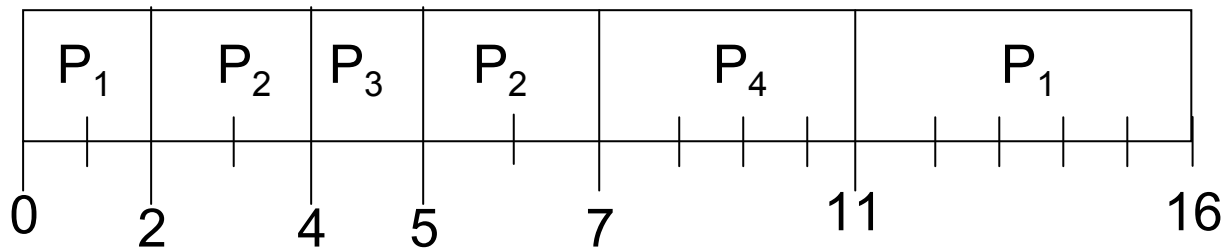
- Thời gian chờ đợi trung bình = (0 + 6 + 3 + 7)/4 = 4

Các giải thuật định thời (5)
Giải thuật Shortest-Job-First (SJF)

Ví dụ về SJF trưng dụng

Process	Thời điểm đến	Tg sdụng CPU
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (có trưng dụng)



- Thời gian chờ đợi trung bình = (9 + 1 + 0 +2)/4 = 3

Các giải thuật định thời (6)
Giải thuật Shortest-Job-First (SJF)

Xác định độ dài thời gian sử dụng CPU lần kế tiếp

- Chỉ có thể ước lượng độ dài.
- Độ dài ước lượng được tính toán dựa trên chiều dài thời gian sử dụng CPU lần trước đó, sử dụng công thức trung bình mũ:
  - t_n là thời gian sử dụng CPU thực tế lần thứ n.
  - τ_{n+1} là ước lượng thời gian sử dụng CPU lần thứ n+1
  - $\alpha, 0 \le \alpha \le 1$, là tham số điều khiển trọng số liên quan giữa lịch sử quá khứ và lịch sử mới đây trong việc ước lượng.
  - Định nghĩa:

$$\tau_{n+1} = \alpha\, t_n + (1-\alpha)\tau_n.$$

Các giải thuật định thời (7)
Giải thuật Shortest-Job-First (SJF)

Ví dụ về tính trung bình mũ

- α =0
  - $\tau_{n+1} = \tau_n$
  - Không tính đến lịch sử
- α =1
  - $\tau_{n+1} = t_n$
  - Chỉ tính đến thời gian sử dụng CPU thực gần nhất
- α =1/2, lịch sử quá khứ và gần đây có trọng số tương đương.
- Nếu mở rộng công thức, ta có:

$$\tau_{n+1} = \alpha\, t_n + (1-\alpha)\,\alpha\, t_{n-1} + \ldots + (1-\alpha)^j\,\alpha\, t_{n-j} + \ldots + (1-\alpha)^{n+1}\tau_0$$

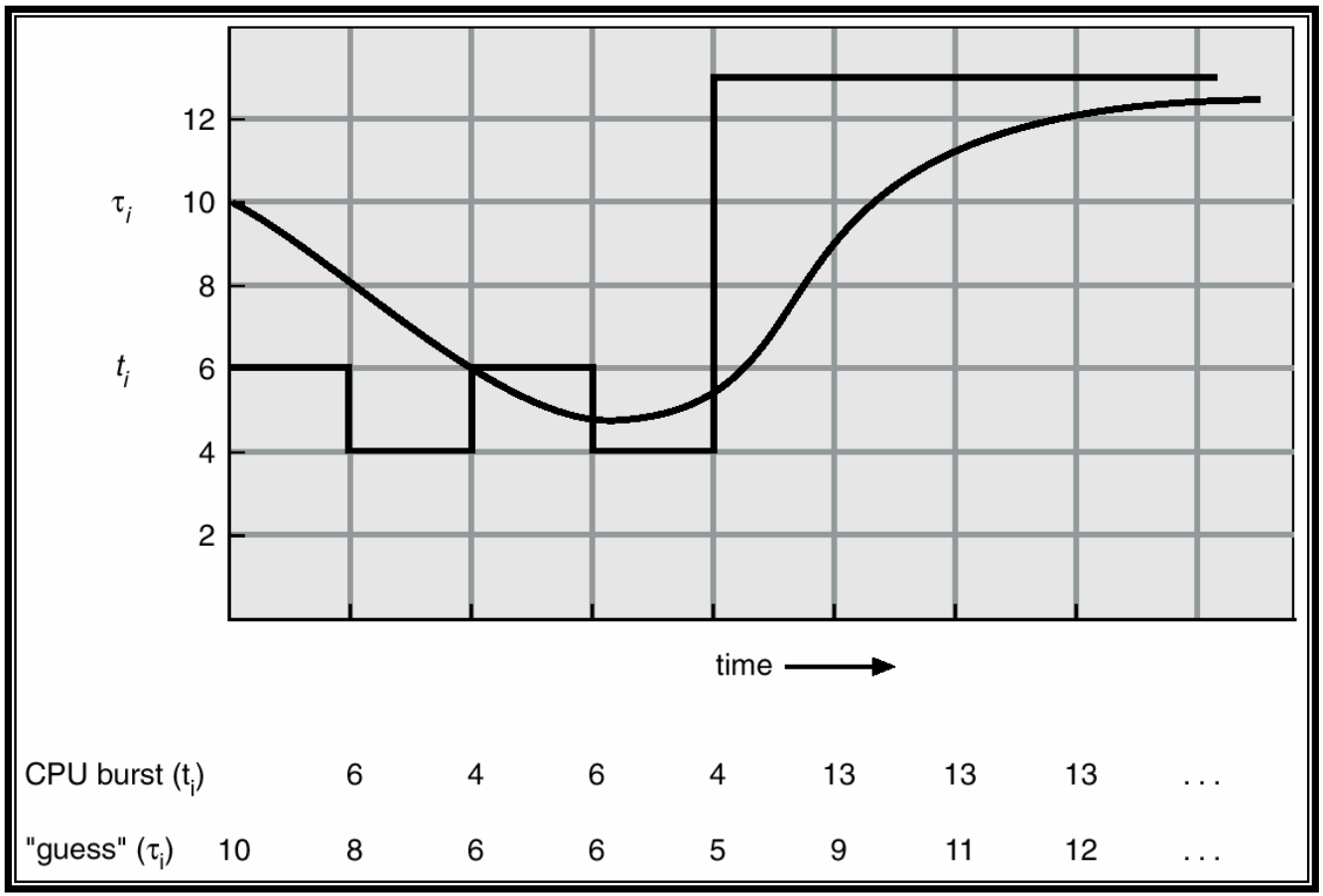
- Vì α và (1- α) nhỏ hơn hay bằng 1, mỗi số hạng tiếp theo có trong số nhỏ hơn số hạng trước đó.

Các giải thuật định thời (8)
Giải thuật Shortest-Job-First (SJR)

Dự tính thời gian sử dụng CPU lần kế tiếp

$\alpha = 1/2$

$\tau_0 = 10$



Các giải thuật định thời (9)
Giải thuật định thời ưu tiên (Priority Scheduling)

- Một chỉ số ưu tiên được gán cho mỗi quá trình
- CPU sẽ được cấp cho quá trình có độ ưu tiên cao nhất, nghĩa là chỉ số ưu tiên nhỏ nhất (smallest integer ≡ highest priority).
- Có thể trưng dụng hoặc không trưng dụng
- SJF là một trường hợp của định thời ưu tiên, trong đó độ ưu tiên là thời gian ước tính sử dụng CPU lần kế tiếp.
- Vấn đề đói CPU (starvation): các quá trình có độ ưu tiên thấp có khả năng không bao giờ được thực thi.
- Giải pháp: đặt thời hạn (aging) – khi thời gian trôi đi, độ ưu tiên của quá trình càng tăng theo.

Các giải thuật định thời (10)
Giải thuật định thời luân phiên (RR -Round-Robin)

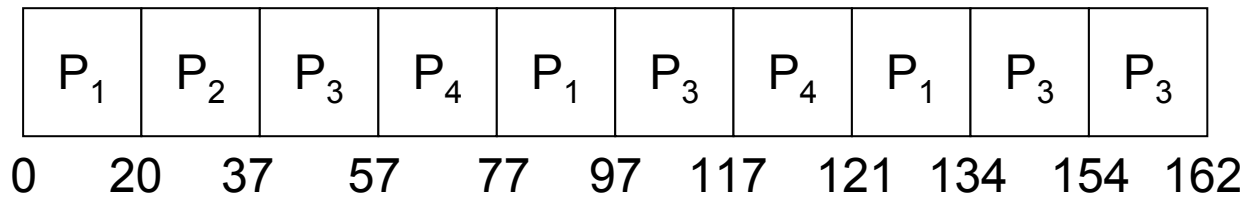
- Mỗi quá trình được CPU phục vụ trong một đơn vị thời gian nhỏ, gọi là định mức thời gian (time quantum), thường là 10-100 milliseconds.
- Sau khoảng thời gian này, CPU bị trưng dụng và giao cho quá trình khác, quá trình bị ngưng và chuyển vào hàng đợi sẵn sàng.
- Nếu có n quá trình đang nằm trong hàng đợi sẵn sàng và định mức thời gian là q, thì mỗi quá trình sẽ nhận được 1/n tổng thời gian của CPU, thời gian phục vụ của CPU cho nó tối đa là q. Sẽ không có quá trình nào phải chờ đợi quá lượng thời gian là $(n\text{-}1)q$.
- Hiệu năng:
  - q lớn $\Rightarrow$ FCFS (FIFO)
  - q nhỏ $\Rightarrow$ q phải đủ lớn do ta phải quan tâm đến việc chuyển đổi ngữ cảnh, nếu không, thời gian lãng phí sẽ rất cao.

Các giải thuật định thời (11)
Giải thuật định thời luân phiên (RR -Round-Robin)

Ví dụ về RR, định mức thời gian = 20

Process	Tg sử dụng CPU
P_1	53
P_2	17
P_3	68
P_4	24

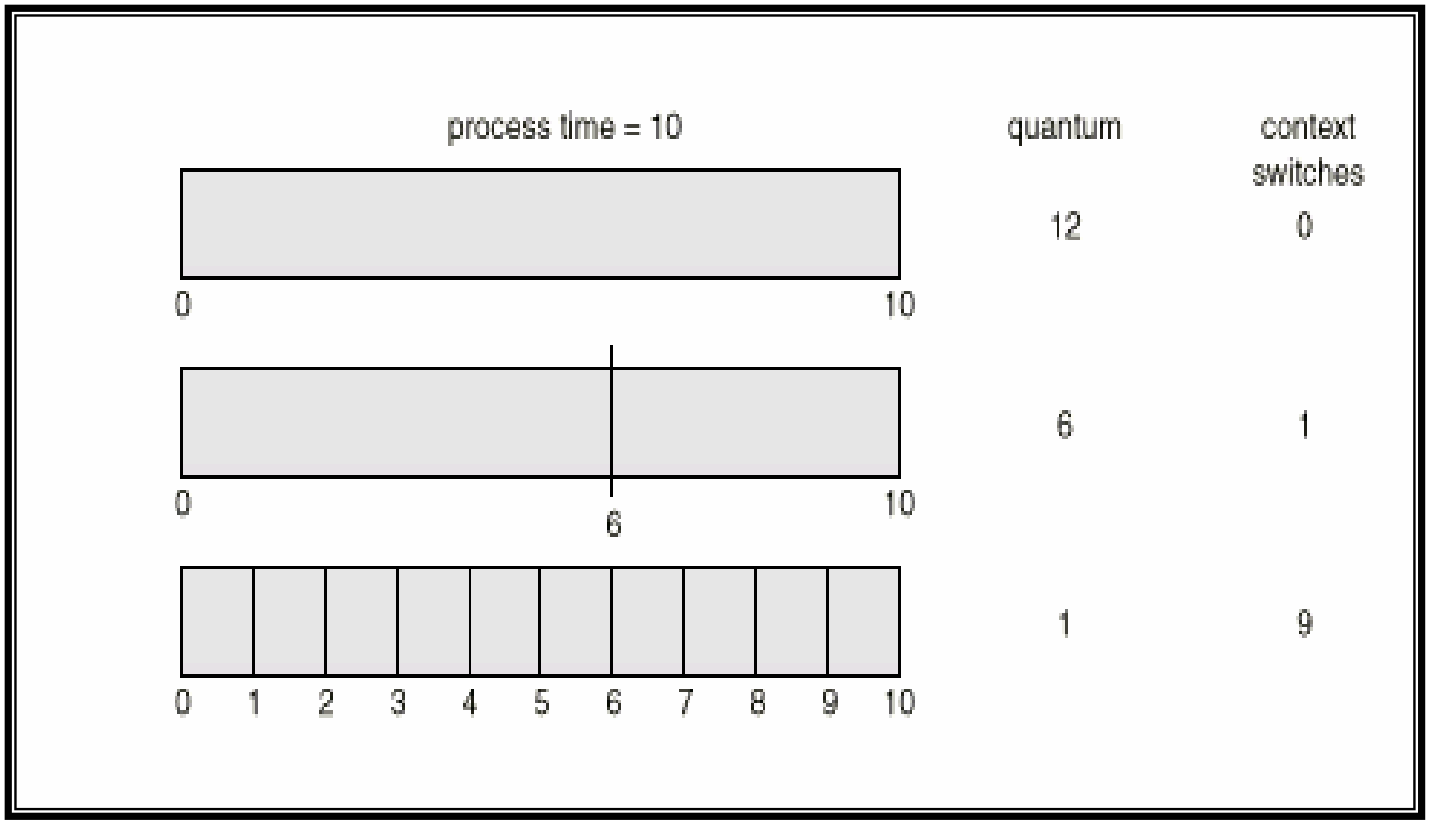
- Biểu đồ Gantt:



- Thông thường, RR có thời gian chờ đợi trung bình lớn hơn SJF, nhưng đảm bảo thời gian đáp ứng tốt hơn.

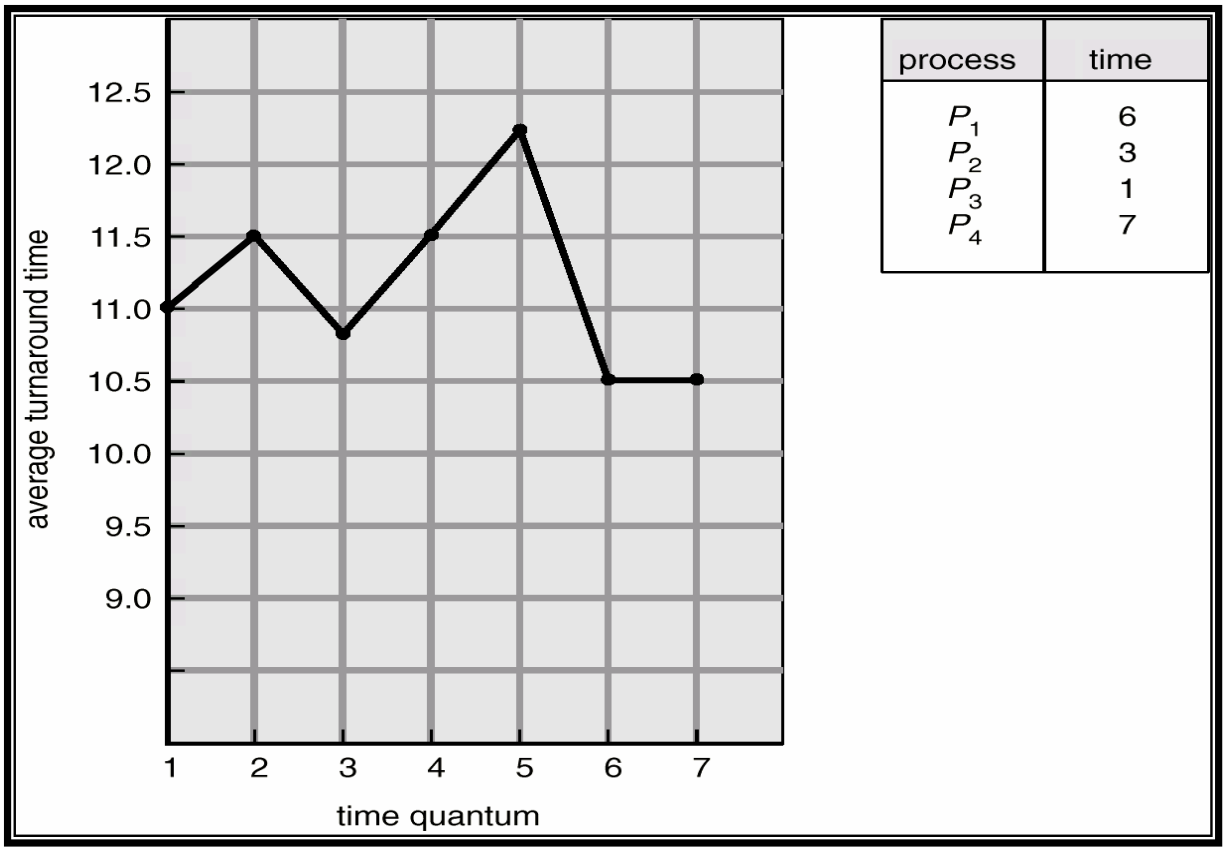
Các giải thuật định thời (12)
Giải thuật định thời luân phiên (RR -Round-Robin)

Định mức thời gian và thời gian chuyển ngữ cảnh



Các giải thuật định thời (13)
Giải thuật định thời luân phiên (RR -Round-Robin)

Thời gian xoay vòng biến đổi với định mức thời gian



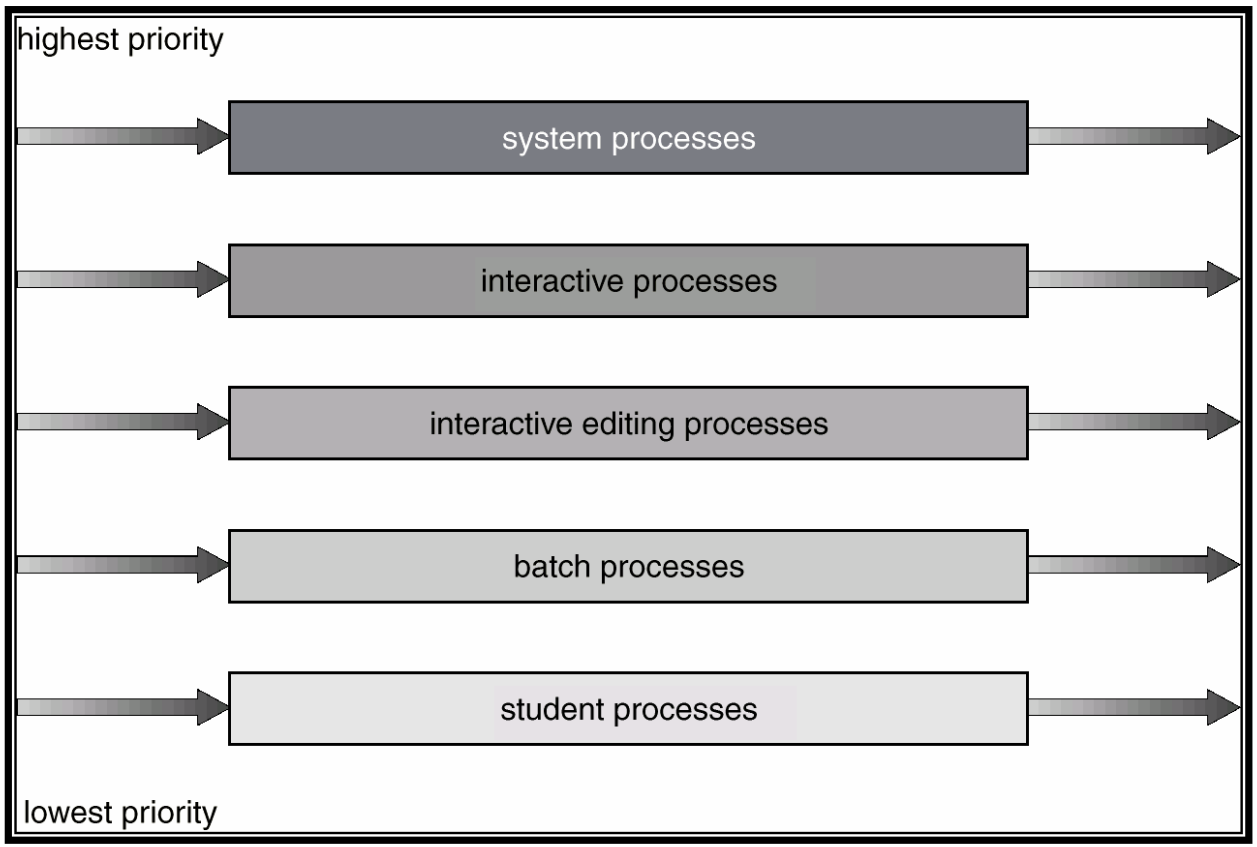
process	time
P_1	6
P_2	3
P_3	1
P_4	7

Các giải thuật định thời (14)
Giải thuật hàng đợi đa cấp (Multilevel Queue)

- Hàng đợi sẵn sàng được chia thành nhiều hàng đợi:
  - foreground (tương tác): cần thời gian đáp ứng nhanh hơn -> ưu tiên hơn;
  - background (bó): có thể đáp ứng chậm hơn -> ít ưu tiên hơn.
- Mỗi hàng đợi có giải thuật lập lịch biểu riêng:
  - foreground – RR
  - background – FCFS
- Thực hiện định thời giữa các hàng đợi:
  - Định thời với độ ưu tiên cố định (nghĩa là, phục vụ tất cả quá trình trong foreground trước rồi đến background). Có khả năng dẫn đến việc đói CPU.
  - Phần chia thời gian (time slice) – mỗi hàng đợi nhận được một khoảng thời gian phục vụ nào đó của CPU để định thời cho các quá trình nằm trong đó;
  - VD 80% cho foreground với RR và 20% cho background với FCFS

Các giải thuật định thời (15)
Giải thuật hàng đợi đa cấp (Multilevel Queue)

Ví dụ hàng đợi đa cấp



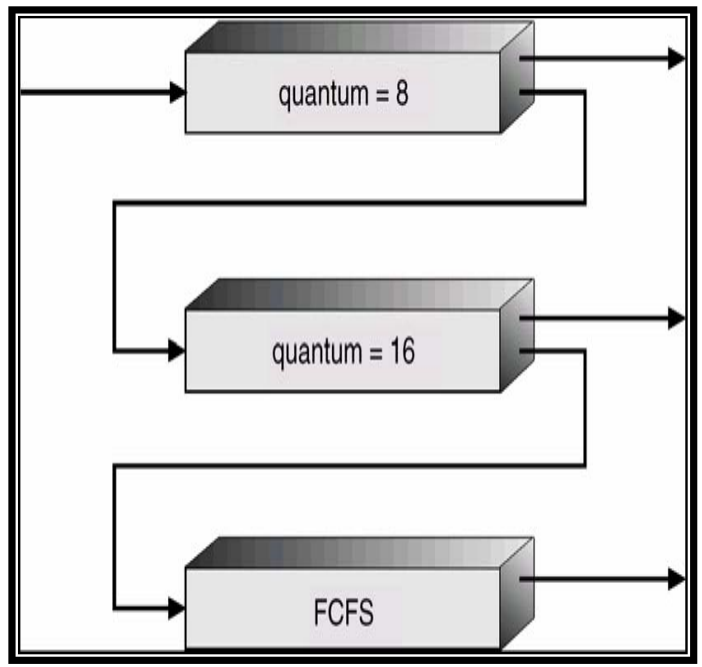
Các giải thuật định thời (16)

Giải thuật hàng đợi phản hồi đa cấp (Multilevel Feedback Queue)

- Một quá trình có thể di chuyển giữa các hàng đợi khác nhau.
- Cơ chế định thời hạn (aging) có thể được cài đặt theo cách:
  - Nếu quá trình dùng quá nhiều thời gian CPU, nó sẽ được di chuyển vào hàng đợi có độ ưu tiên thấp hơn;
  - Nếu quá trình đã chờ quá lâu trong 1 hàng đợi với độ ưu tiên thấp, nó sẽ được chuyển sang hàng đợi có độ ưu tiên cao hơn.
- Bộ định thời đa cấp có phản hồi có thể được định nghĩa bằng những tham số sau:
  - Số lượng hàng đợi
  - Giải thuật định thời cho từng hàng đợi
  - Phương thức dùng để quyết định khi nào thì nâng cấp một quá trình.
  - Phương thức dùng để quyết định khi nào thì hạ cấp một quá trình
  - Phương thức dùng để quyết định là nên đặt quá trình vào hàng đợi nào khi quá trình này cần được phục vụ.

Các giải thuật định thời (17)
Ví dụ về hàng đợi phản hồi đa cấp

- Có 3 hàng đợi:
  - Q_0:định mức thời gian là 8 milliseconds
  - Q_1: định mức thời gian là 16 milliseconds
  - Q_2: FCFS
- Định thời:
  - Một công việc mới sẽ bước vào Q_0 mà ở đó nó sẽ được phục vụ theo kiểu FCFS. Khi đạt được CPU, công việc sẽ nhận được thời gian 8 milliseconds. Nếu nó không hoàn thành trong vòng 8 milliseconds, công việc sẽ được chuyển sang Q_1.
  - Tại Q_1 công việc sẽ lại được phục vụ theo kiểu FCFS và nhận được thêm 16 milliseconds. Nếu nó vẫn chưa hoàn thành, nó sẽ bị tước CPU và chuyển qua Q_2.



Định thời đa xử lý
(Multiple-Processor Scheduling)

- Định thời CPU sẽ phức tạp hơn với nhiều CPU hiện hữu.
- Thường các CPU và bộ nhớ đồng nhất (homogeneous).
- Chia tải (load sharing):
  - Mỗi CPU có hàng đợi riêng dễ dẫn đến tình trạng mất cân bằng tải: một CPU có hàng đợi rỗng, trong khi CPU khác rất bận.
  - Giải pháp dùng hàng đợi chung cho tất cả các CPU.
- Việc định thời (scheduling) và chọn CPU để phân bổ (dispatch) có thể được quản lý:
  - Đa xử lý đối xứng (Symmetric Multiprocessor): mỗi CPU tự định thời từ hàng đợi chung. Có thể dẫn đến cạnh tranh chọn quá trình.
  - Đa xử lý không đối xứng (Asymmetric Multiprocessor): được quản lý bởi 1 trong các CPU -> Cấu trúc chủ-tớ (Master-Slave). Có thể dẫn đến bottleneck.

Định thời thời gian thực (1)
(Real-Time Scheduling)

- Hệ thống thời gian thực cứng: yêu cầu phải hoàn thành một công việc tới hạn trong lượng thời gian được đảm bảo.
  - Phải biết chính xác thời gian mà chức năng hệ điều hành cần cho mỗi thao tác.
  - Chỉ được thực hiện với các phần mềm có mục đích chuyên biệt trên nền phần cứng tận hiến.
- Hệ thống thời gian thực mềm: yêu cầu các quá trình tới hạn nhận được thứ tự ưu tiên cao hơn các quá trình khác.
  - Hệ thống phải có định thời trưng dụng
  - Quá trình thời thực phải có độ ưu tiên cao nhất và không giảm. Chấp nhận tình trạng không công bằng và đói tài nguyên.
  - Độ trễ điều phối phải nhỏ, bảo đảm đáp ứng nhanh cho quá trình thời thực.

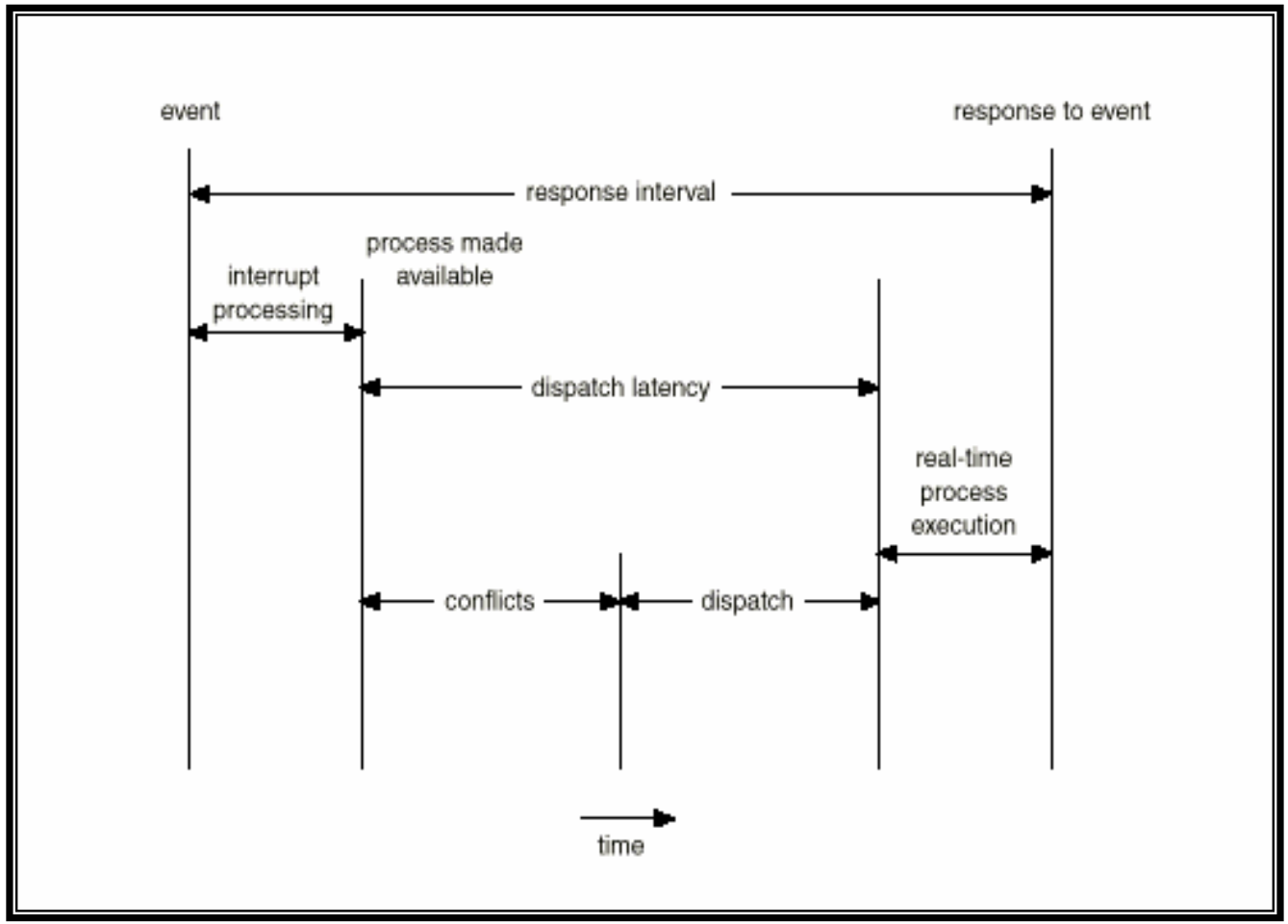
Định thời thời gian thực (2)

Giải pháp cho độ trễ điều phối:

- Unix: hầu hết các ấn bản đều chờ đợi lời gọi hệ thống hoàn thành → độ trễ điều phối dài.
- Giải pháp 1: đưa các điểm trưng dụng (preemption point) vào những lời gọi hệ thống dài (thường trong nhân).
- Giải pháp 2: cho phép trưng dụng nhân (Solaris 2)→ chạy một vi nhân thời thực (real-time micro-kernel) với độ ưu tiên thấp, khi đó vi nhân có thể bị ngắt (interruptible).

Định thời thời gian thực (3)

Độ trễ điều phối



Đánh giá giải thuật (1)
Mô hình xác định (Deterministic Modeling)

- Cách đánh giá và chọn giải thuật thích hợp?
- Mô hình xác định (Deterministic Modeling)
  - Dựa vào tải công việc (workload) được xác định trước và tính toán hiệu năng của mỗi giải thuật.
  - Đơn giản, nhanh, nhưng đòi hỏi tập hợp thông tin đầu vào chính xác, điều khó thực hiện trong thực tế.

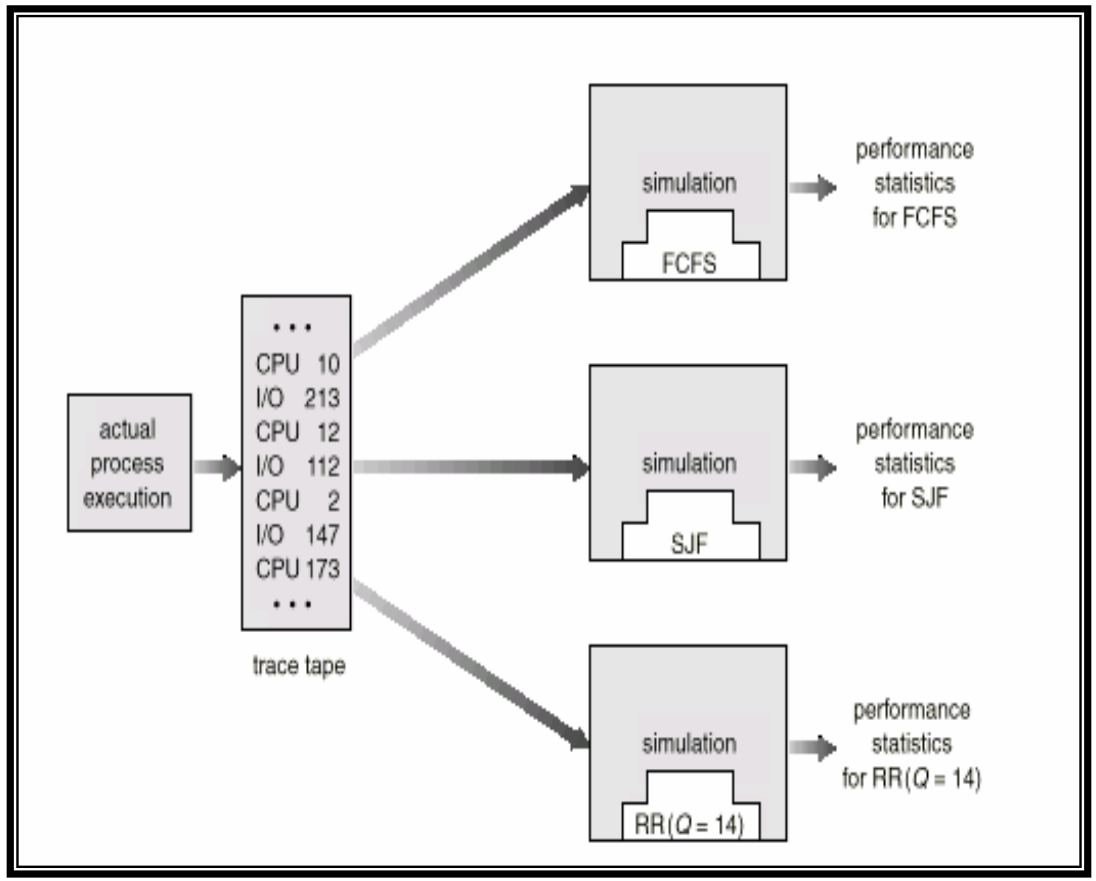
Đánh giá giải thuật (2)
Mô hình hàng đợi (Queueing Model)

- Mô hình hàng đợi (Queueing Model)
  - Dựa vào sự phân bổ chu kỳ CPU và I/O, điều có thể xác định được trong thực tế.
  - Hệ thống máy tính được mô tả như một mạng các server có các hàng đợi riêng (CPU cũng là một server).
  - Biết tốc độ đến và tốc độ phục vụ (dựa vào chu kỳ CPU-I/O) có thể tính khả năng sử dụng, chiều dài hàng đợi trung bình, thời gian chờ trung bình.
  - Công thức Little:
    - ✓ n: chiều dài hàng đợi trung bình
    - ✓ λ: tốc độ đến trung bình cho các quá trình mới (vd 4 quá trình/giây)
    - ✓ W: thời gian chờ trung bình trong hàng đợi

$$n = \lambda \times W$$

Đánh giá giải thuật (3)
Mô phỏng (Simulation)

- Dùng phần mềm mô phỏng hệ thống máy tính
- Khi thực thi mô phỏng, các thống kê hiệu năng của các giải thuật sẽ được ghi nhận và là cơ sở cho việc so sánh các giải thuật.



Đánh giá giải thuật (4)
Cài đặt (Implementation)

- Dùng giải thuật thật (code), đặt vào hệ điều hành thật, cùng với các phương tiện đo lường.
- Đạt độ chính xác cao.
- Đòi hỏi chi phí cao và người dùng có thể phản ứng của người dùng đối với sự thay đổi liên tục của hệ thống.