



HỆ ĐIỀU HÀNH

CHƯƠNG 4: ĐỊNH THỜI CPU (PHẦN 2)

Định thời CPU là hoạt động quan trọng của thành phần quản lý tiến trình và có ảnh hưởng rất lớn đến hiệu suất máy tính cũng như trải nghiệm của người dùng. Trong chương này, người học được trình bày về mục đích và các tiêu chuẩn định thời, cũng như các chiến lược định thời CPU cơ bản.



NỘI DUNG

4. Các giải thuật định thời

- **Shortest Job First (SJF)**
- **Shortest Remaining Time First (SRTF)**
- **Priority Scheduling**
- **Round Robin**
- **Highest Response Ratio Next (HRRN)**
- **Multilevel Queue**
- **Multilevel Feedback Queue**



CÁC GIẢI THUẬT ĐỊNH THỜI

4.3. Shortest-Job-First (SJF)

04.



4.3. Shortest-Job-First (SJF)

- **Hàm chọn lựa:** tiến trình có thời gian yêu cầu thực thi (CPU burst) ngắn nhất sẽ được chọn.

Khi CPU trống, OS sẽ chọn tiến trình có **CPU Burst ngắn nhất** để được thực thi tiếp theo.

Giải thuật này sử dụng chiều dài thời gian thực thi của tiến trình làm căn cứ để chọn lựa.

- **SJF có thể được hiện thực với cả hai chế độ quyết định:** trưng dụng và không trưng dụng.



4.3. Shortest-Job-First (SJF)

- SJF ở chế độ không trưng dụng:

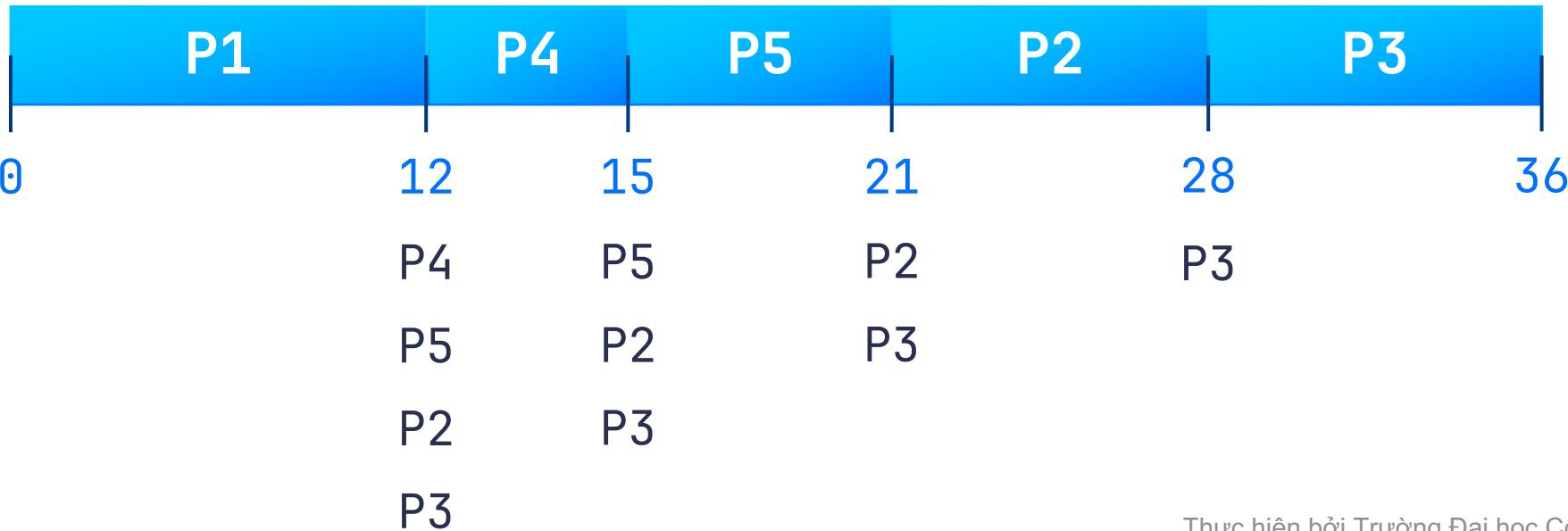
- Hàm chọn lựa được thực thi khi CPU trống.
- Khi tiến trình được cấp CPU thì sẽ thực thi cho đến khi kết thúc.
- Khi một tiến trình kết thúc, một tiến trình đã sẵn sàng khác có thời gian thực thi ngắn nhất sẽ được chọn.



Ví dụ: SJF ở chế độ không trung dung

Giản đồ Gantt

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6





Ví dụ: SJF ở chế độ không trung dung

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

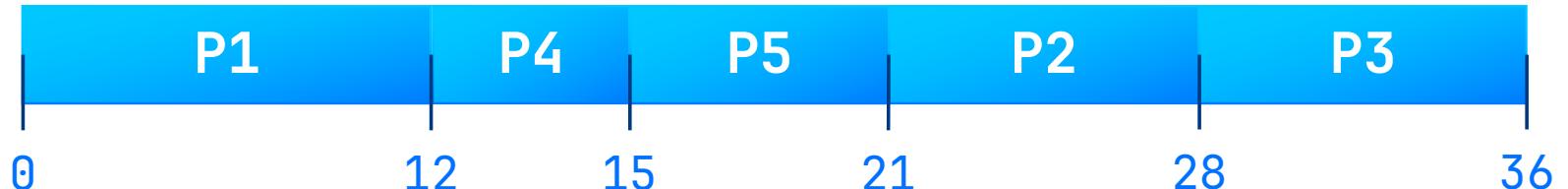
Thời gian đáp ứng:

$$P1 = 0 \quad P2 = 19 \quad P3 = 23$$

$$P4 = 3 \quad P5 = 3$$

$$\begin{aligned}ART &= (0+19+23+3+3)/5 \\&= 9.6\end{aligned}$$

Giản đồ Gantt





Ví dụ: SJF ở chế độ không trung dung

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

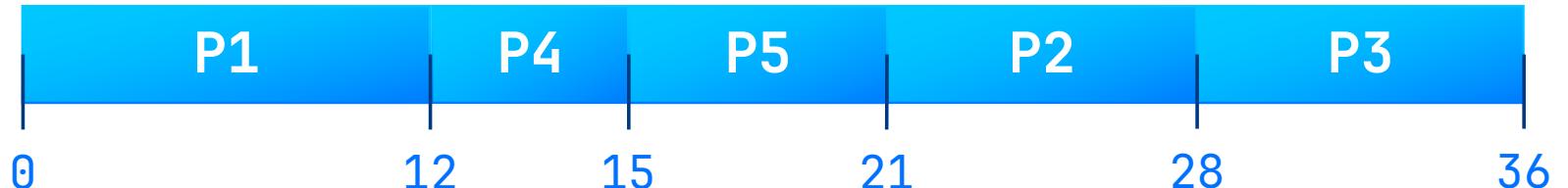
Thời gian hoàn thành:

$$P1 = 12 \quad P2 = 26 \quad P3 = 31$$

$$P4 = 6 \quad P5 = 9$$

$$\begin{aligned}ATaT &= (12+26+31+6+9)/5 \\&= 16.8\end{aligned}$$

Giản đồ Gantt





Ví dụ: SJF ở chế độ không trung dung

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

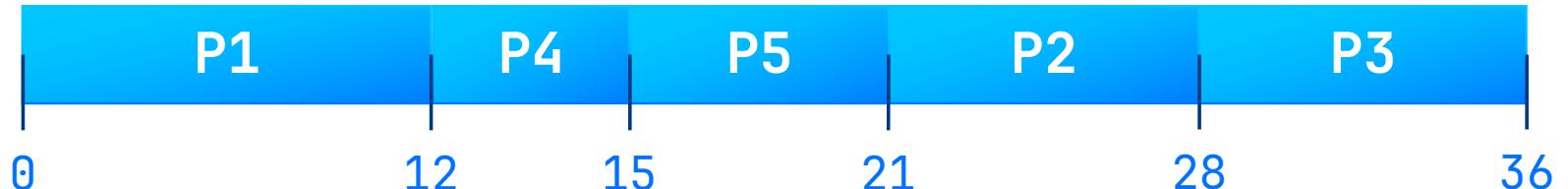
Thời gian chờ:

$$P1 = 0 \quad P2 = 19 \quad P3 = 23$$

$$P4 = 3 \quad P5 = 3$$

$$\begin{aligned} AWT &= (0+19+23+3+3)/5 \\ &= 9.6 \end{aligned}$$

Giản đồ Gantt





4.3. Shortest-Job-First (SJF)

- SJF ở chế độ trung dung

- Hàm chọn lựa được thực thi khi có tiến trình mới xuất hiện hoặc có tiến trình kết thúc.
- Khi có tiến trình mới xuất hiện với **CPU-burst nhỏ hơn thời gian yêu cầu còn lại** (remaining time) của tiến trình đang thực thi, **tiến trình mới sẽ được chọn và tiến trình đang thực thi sẽ bị dừng lại**.
- Khi một tiến trình kết thúc, một tiến trình khác có CPU-burst (hoặc thời gian yêu cầu còn lại) nhỏ nhất sẽ được chọn tiếp theo.
- SJF ở chế độ trung dung còn được gọi là **Shortest-Remaining-Time-First (SRTF)**.

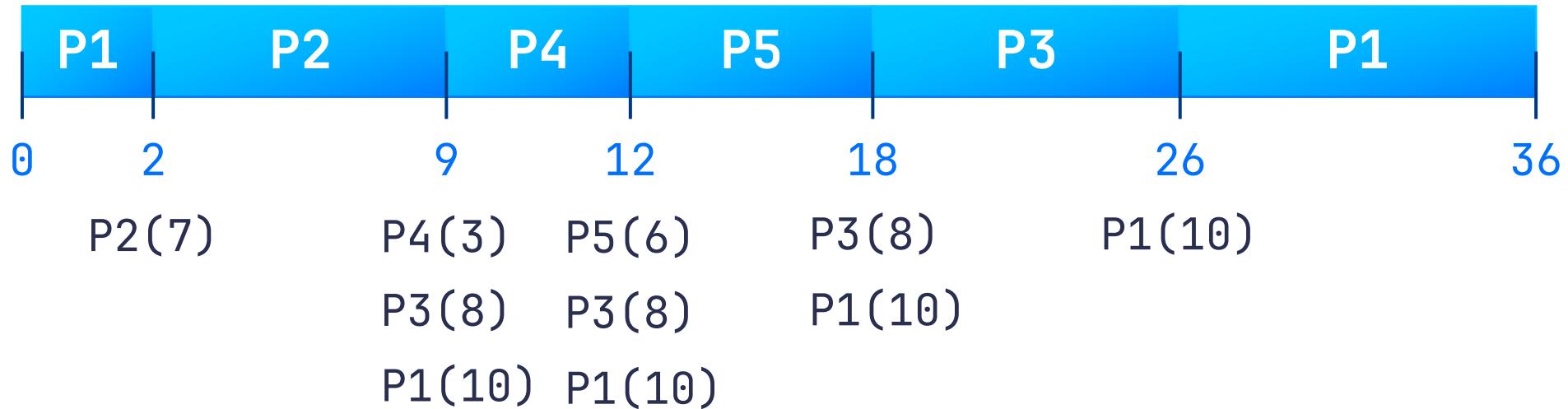
SJF là tối ưu về thời gian đợi: có thời gian chờ đợi trung bình ngắn nhất với một tập tiến trình cho trước.



Ví dụ: SJF ở chế độ trung dụng

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

Giản đồ Gantt

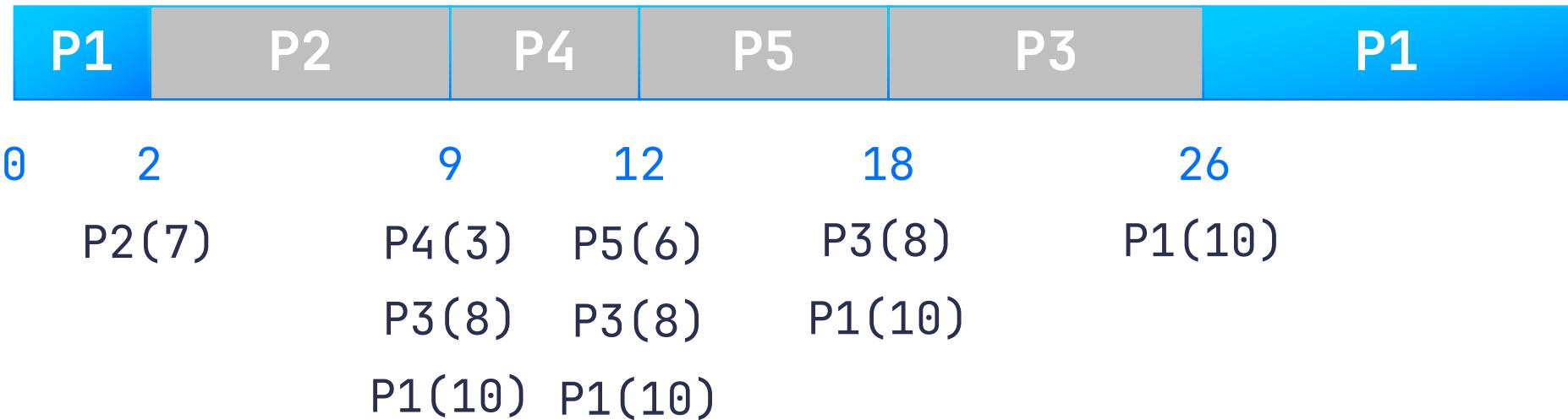




Ví dụ: SJF ở chế độ trung dụng

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

Giản đồ Gantt





Ví dụ: SJF ở chế độ trung dụng

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

Thời gian đáp ứng:

$$P1 = 0 \quad P2 = 0 \quad P3 = 13$$

$$P4 = 0 \quad P5 = 0$$

$$\begin{aligned}ART &= (0+0+13+0+0)/5 \\&= 2.6\end{aligned}$$

Giản đồ Gantt





Ví dụ: SJF ở chế độ trung dụng

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

Thời gian hoàn thành:

$$P1 = 36 \quad P2 = 7 \quad P3 = 21$$

$$P4 = 3 \quad P5 = 6$$

$$\begin{aligned}ATaT &= (0+0+13+0+0)/5 \\&= 14.6\end{aligned}$$

Giản đồ Gantt





Ví dụ: SJF ở chế độ trung dụng

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

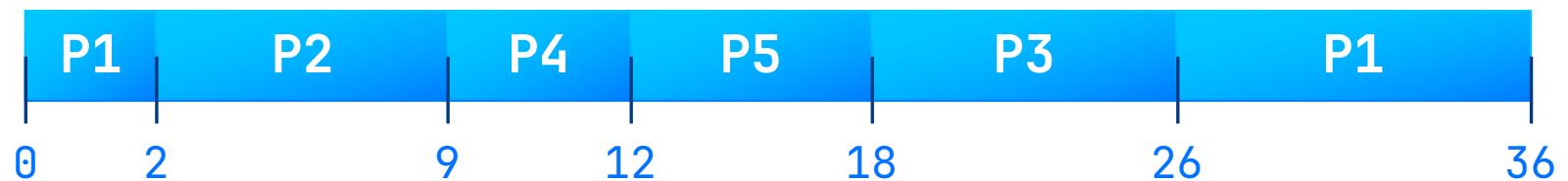
Thời gian chờ:

$$P1 = 24 \quad P2 = 0 \quad P3 = 13$$

$$P4 = 0 \quad P5 = 0$$

$$\begin{aligned} AWT &= (24+0+13+0+0)/5 \\ &= 7.4 \end{aligned}$$

Giản đồ Gantt





4.3. Shortest-Job-First (SJF)

Nhận xét về giải thuật SJF

- Có thể xảy ra tình trạng “đói” tài nguyên (**starvation**) đối với các tiến trình có CPU-burst lớn nếu có nhiều tiến trình với CPU-burst nhỏ (liên tục) xuất hiện trong hệ thống.
- Cơ chế không trưng dụng không phù hợp cho hệ thống time sharing (interactive).
- Giải thuật SJF ngầm định rằng độ ưu tiên được xác định dựa theo độ dài CPU-burst.
 - Các tiến trình hướng CPU (CPU-bound) có độ ưu tiên thấp hơn so với tiến trình hướng I/O (I/O-bound).
 - Tuy nhiên, khi một tiến trình hướng CPU được thực thi thì nó độc chiếm CPU cho đến khi kết thúc.



4.3. Shortest-Job-First (SJF)

Nhận xét về giải thuật SJF

- **Ưu điểm:** SJF tối ưu trong việc giảm thời gian đợi trung bình.
- **Hạn chế:** Cần phải ước lượng lượng thời gian cần CPU tiếp theo của tiến trình.

→ Giải pháp cho vấn đề này?



4.3. Shortest-Job-First (SJF)

Nhận xét về giải thuật SJF

- Thời gian sử dụng CPU chính là độ dài của CPU burst:
 - Trung bình tất cả các CPU Burst đo được trong quá khứ.
 - Nhưng thông thường những CPU Burst càng mới càng phản ánh đúng hành của tiến trình trong tương lai.
- Một kỹ thuật thường dùng là sử dụng trung bình hàm mũ (exponential averaging):

$$\tau_{n+1} = \alpha t_n + (1 - \alpha)\tau_n, 0 \leq \alpha \leq 1$$

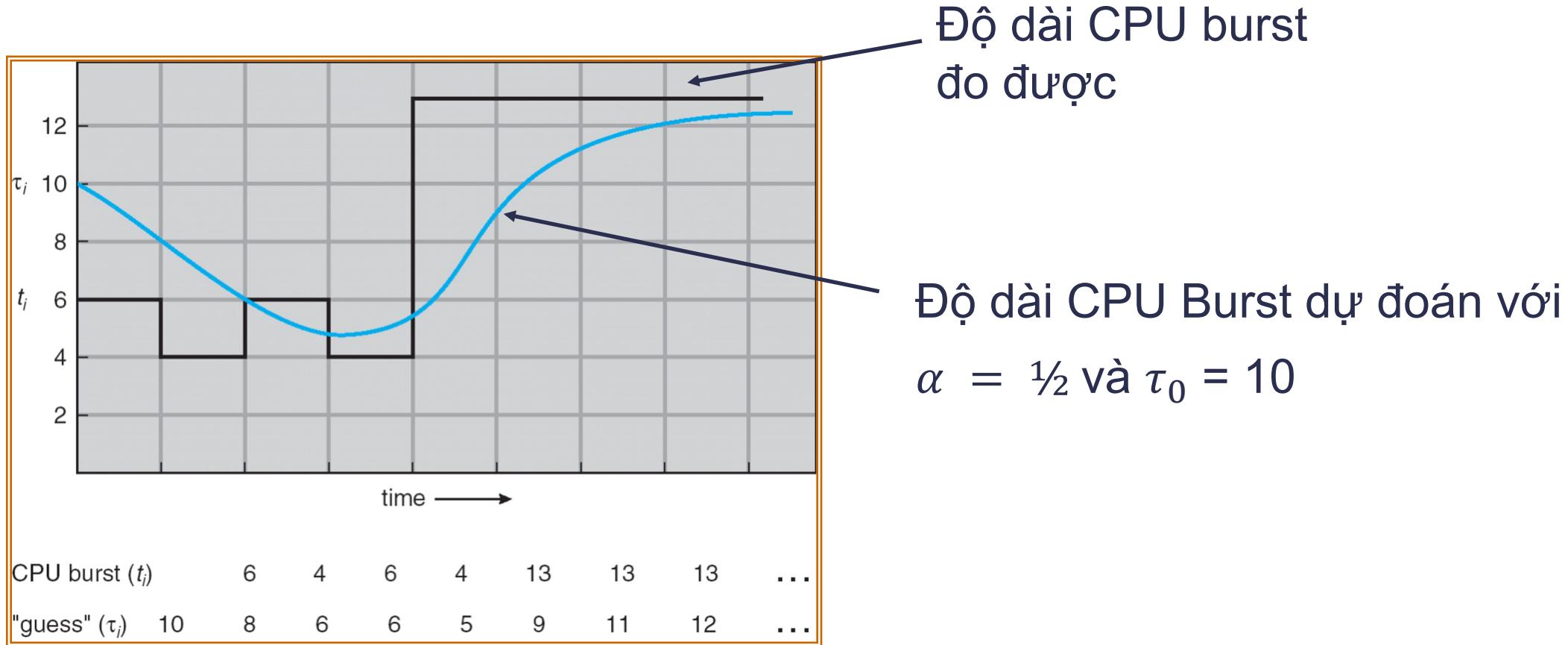
$$\tau_{n+1} = \alpha t_n + (1 - \alpha)\alpha t_{n-1} + \dots + (1 - \alpha)^j \alpha t_{n-j} + \dots + (1 - \alpha)^{n+1}\tau_0, 0 \leq \alpha \leq 1$$

Nếu chọn $\alpha = 1/2$ thì có nghĩa là trị đo được t_n và trị dự đoán τ_n được xem quan trọng như nhau.



4.3. Shortest-Job-First (SJF)

Dự đoán thời gian sử dụng CPU





CÁC GIẢI THUẬT ĐỊNH THỜI

4.4. Định thời theo độ ưu tiên - Priority Scheduling

04.



4.4. Định thời theo độ ưu tiên – Priority Scheduling

- Mỗi tiến trình sẽ được gán một độ ưu tiên (thường biểu diễn bởi một con số).
- CPU sẽ được cấp cho tiến trình có độ ưu tiên cao nhất theo các giá trị số được gán (có thể theo thứ tự tăng dần hay giảm dần).
- Định thời sử dụng độ ưu tiên có thể:
 - Preemptive
 - Non-preemptive



4.4. Định thời theo độ ưu tiên – Priority Scheduling

Cách gán độ ưu tiên cho tiến trình

- SJF là một giải thuật định thời sử dụng độ ưu tiên được xác định dựa vào thời gian sử dụng CPU (giá trị được ước lượng).
- Ngoài ra, việc gán độ ưu tiên còn có thể dựa vào:
 - Yêu cầu về bộ nhớ.
 - Số lượng file được mở.
 - Tỉ lệ thời gian dùng cho I/O trên thời gian sử dụng CPU.
 - Các yêu cầu bên ngoài ví dụ như: số tiền người dùng trả khi thực thi công việc.



4.4. Định thời theo độ ưu tiên – Priority Scheduling

Hạn chế của định thời theo độ ưu tiên

- Vấn đề trì hoãn vô hạn định: tiến trình có độ ưu tiên thấp có thể không bao giờ được thực thi (do có những tiến trình độ ưu tiên cao hơn liên tục xuất hiện).
- Giải pháp: làm mới (aging) – độ ưu tiên của tiến trình sẽ tăng theo thời gian.



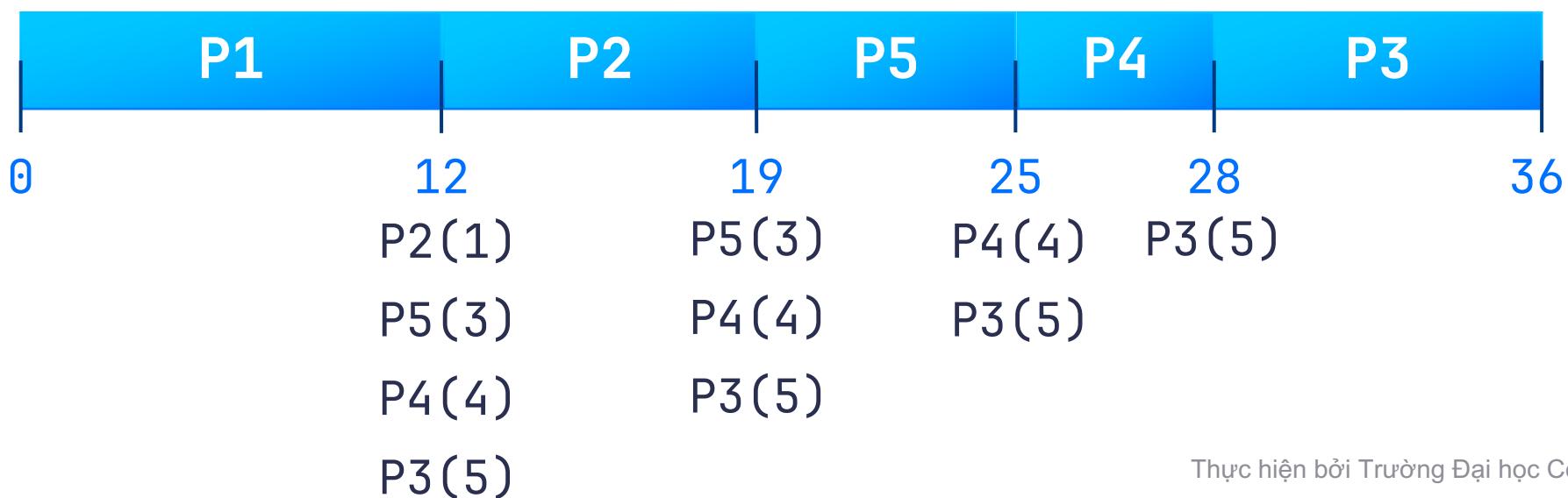
4.4. Ví dụ: Priority Scheduling

Thực hiện định thời cho các tiến trình sau sử dụng giải thuật Non-preemptive Priority Scheduling.

Quy ước: số Priority càng nhỏ thì độ ưu tiên càng cao.

Giản đồ Gantt (Non-preemptive)

Process	Arrival Time	Burst Time	Priority
P1	0	12	2
P2	2	7	1
P3	5	8	5
P4	9	3	4
P5	12	6	3





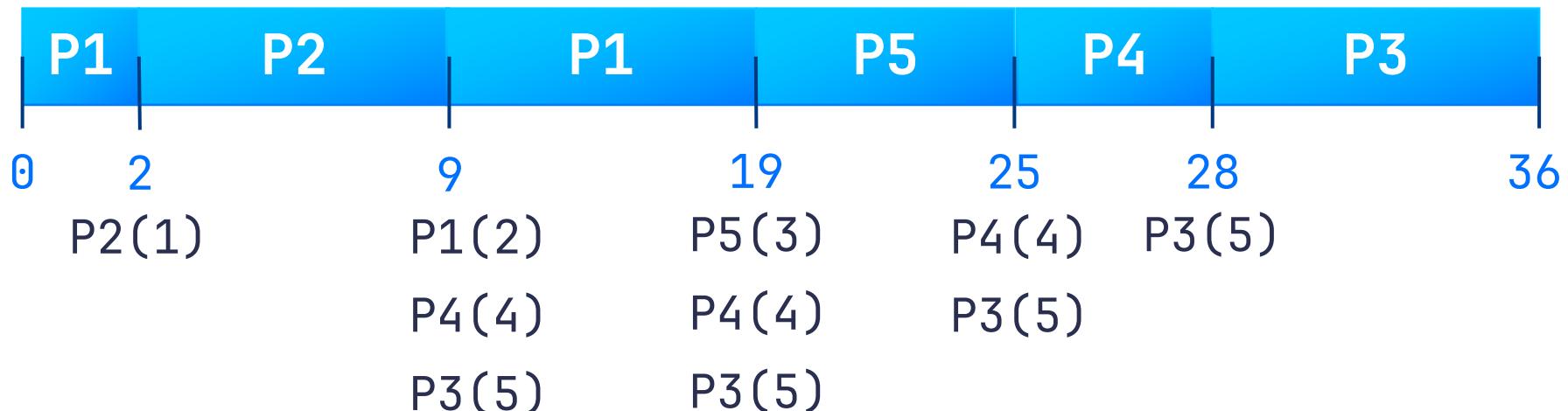
4.4. Ví dụ: Priority Scheduling

Thực hiện định thời cho các tiến trình sau sử dụng giải thuật Preemptive Priority Scheduling.

Quy ước: số Priority càng nhỏ thì độ ưu tiên càng cao.

Giản đồ Gantt (Preemptive)

Process	Arrival Time	Burst Time	Priority
P1	0	12	2
P2	2	7	1
P3	5	8	5
P4	9	3	4
P5	12	6	3





CÁC GIẢI THUẬT ĐỊNH THỜI

4.5. Round Robin (RR)

04.



4.5. Round Robin (RR)

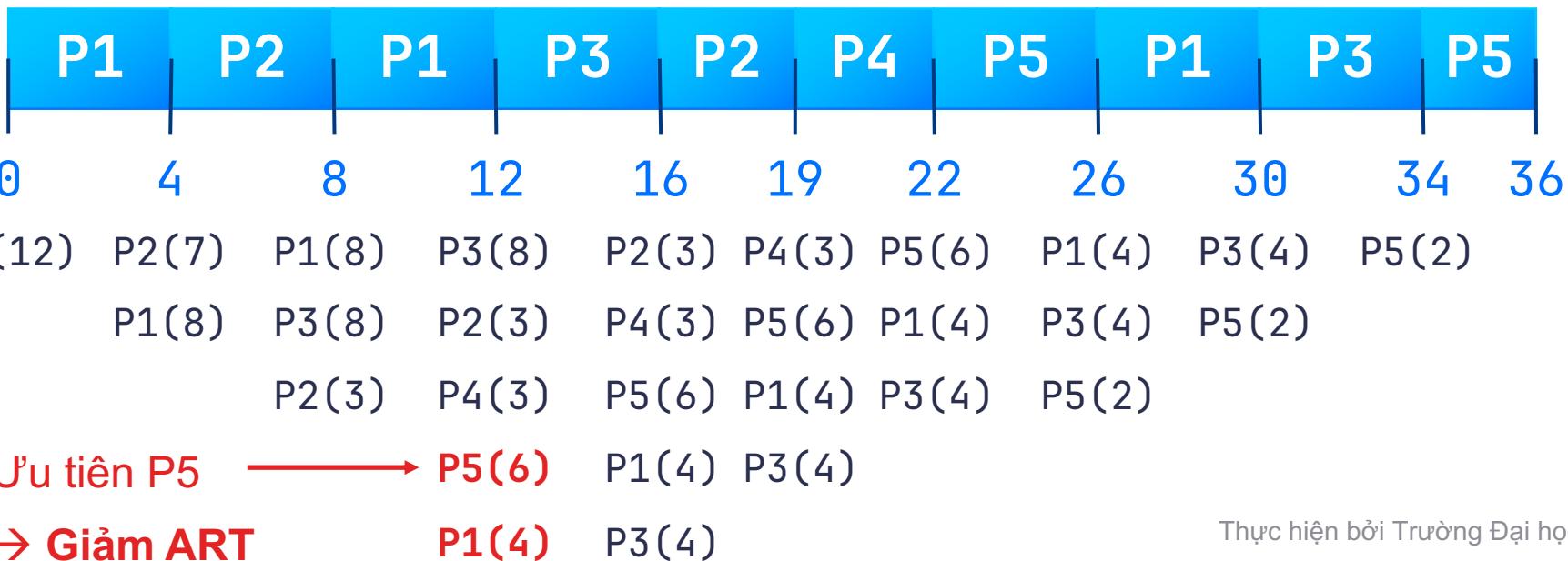
- Mỗi tiến trình nhận được một đơn vị thời gian CPU (**time slice**, **quantum time**) để thực thi. **Thông thường khoảng thời gian này nhỏ**, từ 10-100 ms.
- Sau khoảng thời gian đó, tiến trình bị đoạt quyền và trở về cuối *ready queue*.
- Gọi ***n*** là số lượng tiến trình trong *ready queue* và ***q*** là khoảng thời gian đơn vị mà CPU được cấp phát cho tiến trình (quantum time), khi đó, không có tiến trình nào phải chờ đợi quá **$(n - 1)q$** đơn vị thời gian.



Ví dụ: Round Robin

Giản đồ Gantt ($q = 4$)

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6





Ví dụ: Round Robin

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

Thời gian đáp ứng:

$$P1 = 0 \quad P2 = 2 \quad P3 = 7$$

$$P4 = 10 \quad P5 = 10$$

$$\begin{aligned}ART &= (0+2+7+10+10)/5 \\&= 5.8\end{aligned}$$

Giản đồ Gantt ($q = 4$)





Ví dụ: Round Robin

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

Thời gian hoàn thành:

$$P1 = 30 \quad P2 = 17 \quad P3 = 29$$

$$P4 = 13 \quad P5 = 24$$

$$\begin{aligned} ATAT &= (30+17+29+13+24)/5 \\ &= 22.6 \end{aligned}$$

Giản đồ Gantt ($q = 4$)





Ví dụ: Round Robin

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

Thời gian chờ:

$$P1 = 18 \quad P2 = 10 \quad P3 = 21$$

$$P4 = 10 \quad P5 = 18$$

$$\begin{aligned} AWT &= (30+17+29+13+24)/5 \\ &= 15.4 \end{aligned}$$

Giản đồ Gantt ($q = 4$)





4.5. Round Robin (RR)

Nhận xét về giải thuật Round Robin

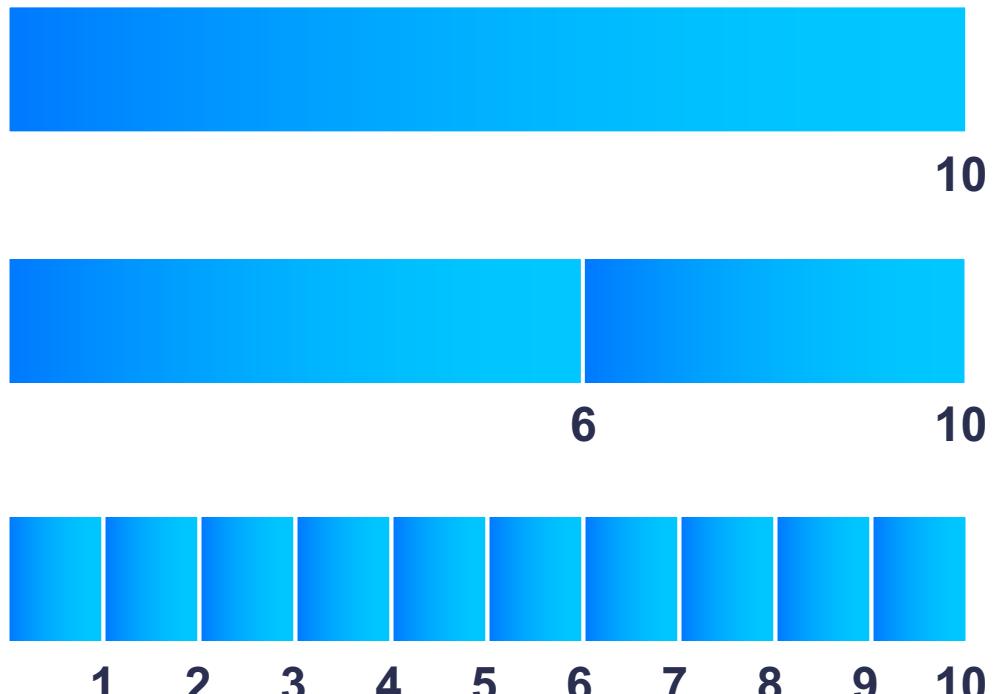
- **Nếu q lớn:** RR trở thành FCFS.
- **Nếu q nhỏ:** phải tốn chi phí chuyển ngữ cảnh giữa các tiến trình => q không nên quá nhỏ.
- Ưu tiên tiến trình hướng CPU (CPU-bound process).
- ***RR sử dụng một giả thiết ngầm là tất cả các tiến trình đều có tầm quan trọng ngang nhau.***
- **Ưu điểm:** Thời gian đáp ứng nhỏ.
- **Hạn chế:** Thời gian chờ đợi trung bình và thời gian hoàn thành trung bình của giải thuật RR thường khá lớn.



4.5. Round Robin (RR)

Quantum time và chuyển ngữ cảnh trong RR

Process time = 10



Quantum

12

0

Context switch

6

1

1

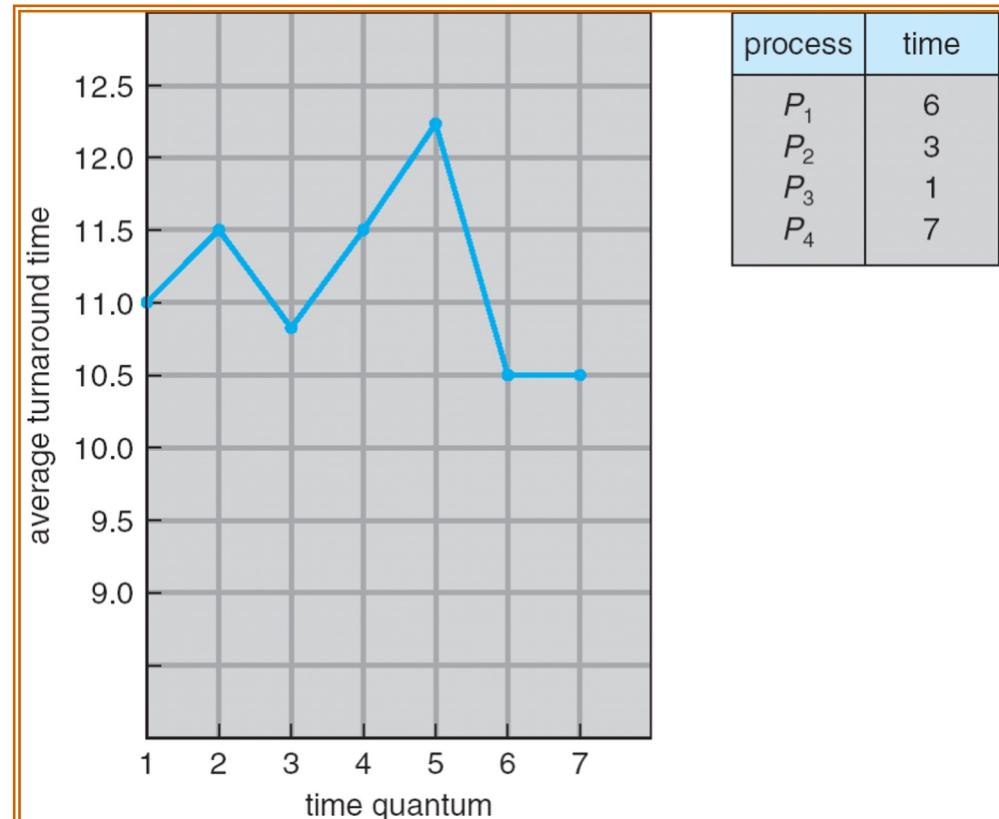
9



4.5. Round Robin (RR)

Quantum time và Thời gian hoàn thành

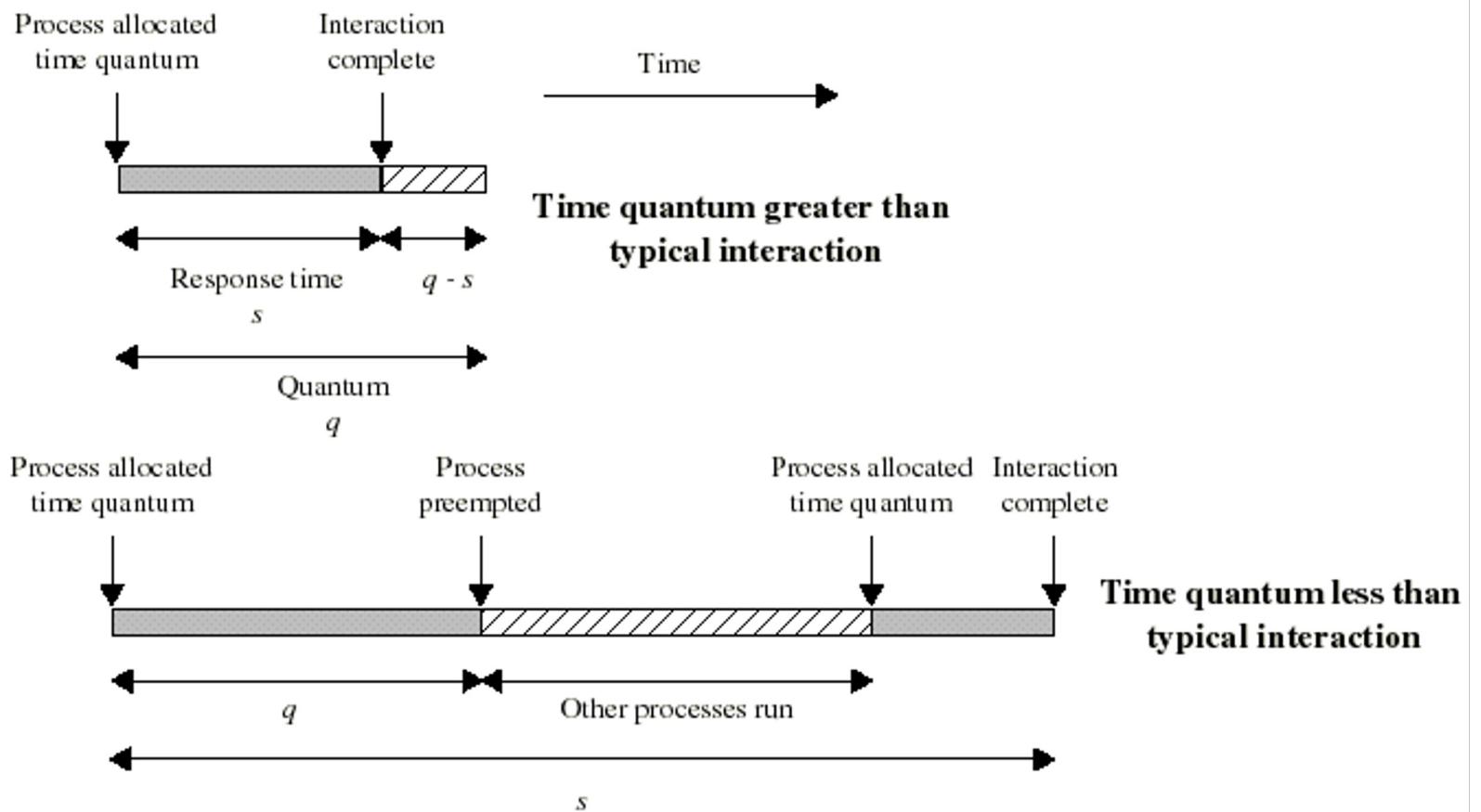
- Thời gian hoàn thành trung bình không chắc sẽ được cải thiện khi quantum lớn





4.5. Round Robin (RR)

Quantum time và Thời gian đáp ứng





4.5. Round Robin (RR)

Quantum time và hiệu suất hệ thống

- Khi thực hiện chuyển ngữ cảnh, ***kernel thread*** sẽ sử dụng CPU, không phải ***user thread***.
- **Phí tổn hệ thống (OS Overhead)**: thời gian OS sử dụng CPU để thực hiện chuyển ngữ cảnh.
- **Hiệu suất hệ thống**: tùy thuộc vào kích thước của ***quantum time***
 - **Nếu quantum time ngắn**: thời gian đáp ứng nhanh, nhưng phí tổn hệ thống lớn do số lần chuyển ngữ cảnh tăng.
 - **Nếu quantum time dài**: hiệu quả sử dụng CPU tốt hơn, nhưng thời gian đáp ứng cũng lớn.
 - **Nếu quantum time quá lớn, RR trở thành FCFS.**



4.5. Round Robin (RR)

Cách chọn quantum time

- Quantum time và thời gian cho chuyển ngữ cảnh:

Nếu quantum time là 20 ms và thời gian chuyển ngữ cảnh là 5 ms, như vậy OS overhead chiếm $5/25 = 20\%$.

Nếu quantum là 500 ms, thì phí tổn chỉ còn 1%. Nhưng nếu có nhiều người sử dụng trên hệ thống và thuộc loại interactive thì sẽ thấy đáp ứng rất chậm.

Tùy thuộc vào tập công việc mà lựa chọn quantum time.

Quantum time nên lớn trong tương quan so sánh với thời gian cho chuyển ngữ cảnh.

- Ví dụ với 4.3 BSD UNIX, quantum time là 1s.



CÁC GIẢI THUẬT ĐỊNH THỜI

4.6. Highest Response Ratio Next (HRRN)

04.



4.6. Highest Response Ratio Next (HRRN)

- Chọn tiến trình kế tiếp có giá trị **RR (Response ratio)** lớn nhất.
- Các tiến trình ngắn được ưu tiên hơn (vì *service time* nhỏ).

$$RR = \frac{\text{time spent waiting} + \text{expected service time}}{\text{expected service time}}$$

- Câu hỏi thảo luận:
 - So sánh với cơ chế Aging?
 - Ưu điểm và hạn chế của hướng tiếp cận này?



CÁC GIẢI THUẬT ĐỊNH THỜI

4.7. Multilevel Queue

04.



4.7. Multilevel Queue

- *Ready queue* được chia thành nhiều hàng đợi riêng biệt theo một số tiêu chuẩn sau:
 - Đặc điểm và yêu cầu định thời của tiến trình.
 - Phân loại tiến trình: Foreground (interactive) và background,...
- Tiến trình được gán cố định vào một hàng đợi, mỗi hàng đợi sử dụng giải thuật định thời riêng.



4.7. Multilevel Queue

Việc định thời giữa các hàng đợi

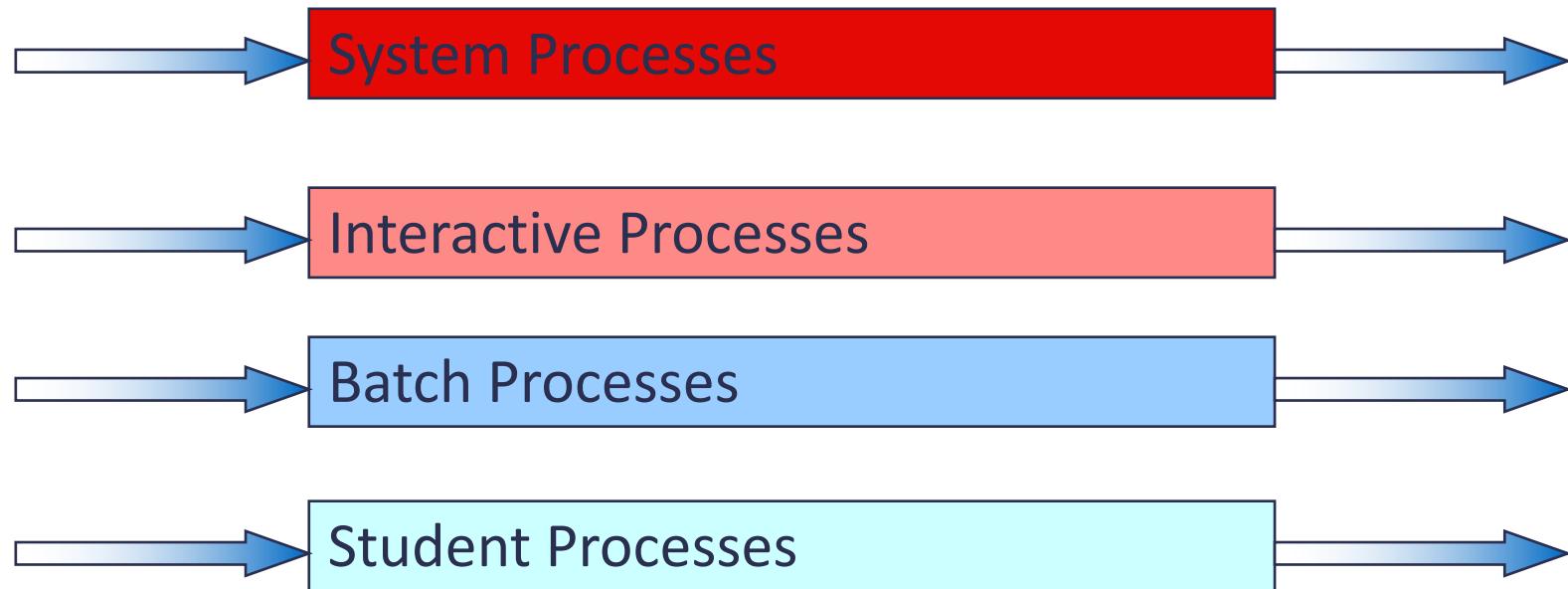
- Hệ điều hành cần phải định thời cho các hàng đợi:
 - **Fixed priority scheduling**: phục vụ từ hàng đợi có độ ưu tiên cao đến thấp → Có thể phát sinh vấn đề đói tài nguyên (**starvation**).
 - **Time slice**: mỗi hàng đợi được nhận một khoảng thời gian chiếm CPU và phân phối cho các tiến trình trong hàng đợi khoảng thời gian đó. Ví dụ: 80% cho hàng đợi foreground định thời bằng RR và 20% cho hàng đợi background định thời bằng giải thuật FCFS.



4.7. Multilevel Queue

Ví dụ phân chia hàng đợi và tiến trình

Độ ưu tiên cao nhất



Độ ưu tiên thấp nhất



4.7. Multilevel Queue

Hạn chế của Multilevel Queue

- Tiến trình không thể chuyển từ hàng đợi này sang hàng đợi khác:
 - Có thể làm giảm hiệu suất hệ thống trong trường hợp một hàng đợi có nhiều tiến trình trong khi các hàng đợi khác lại trống.
 - Khắc phục bằng cơ chế feedback: cho phép tiến trình di chuyển một cách thích hợp giữa các hàng đợi khác nhau.
 - Giải thuật: **Multilevel Feedback Queue**



CÁC GIẢI THUẬT ĐỊNH THỜI

4.8. Multilevel Feedback Queue

04.



4.8. Multilevel Feedback Queue

Phân loại tiến trình dựa trên các đặc tính về **CPU-burst**.

Sử dụng chế độ **trưng dụng** (preemptive).

Sau một khoảng thời gian nào đó, các tiến trình hướng I/O và tiến trình interactive sẽ ở các hàng đợi có độ ưu tiên cao hơn còn các tiến trình hướng CPU sẽ ở các hàng đợi có độ ưu tiên thấp hơn.

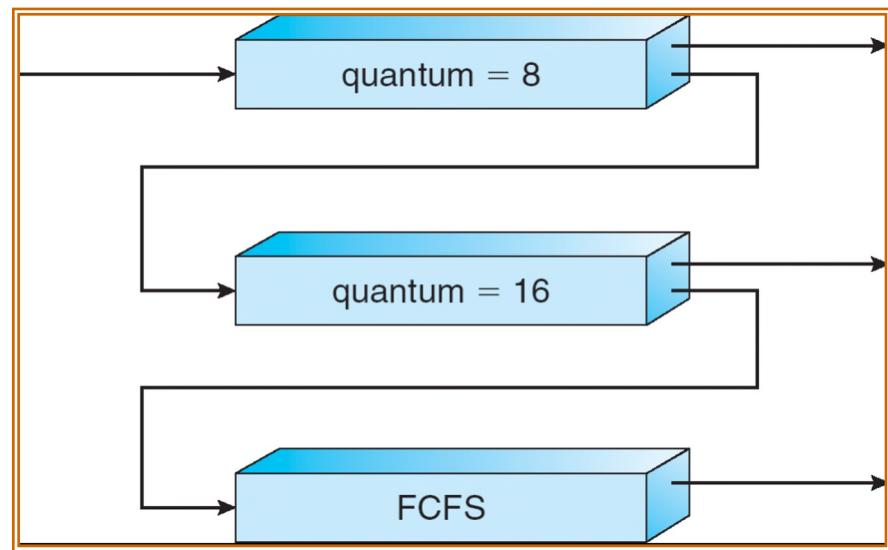
Một tiến trình đã chờ quá lâu ở một hàng đợi có độ ưu tiên thấp có thể được chuyển đến hàng đợi có độ ưu tiên cao hơn (cơ chế aging).



4.8. Multilevel Feedback Queue

Ví dụ về Multilevel Feedback Queue

- Ví dụ: Có 3 hàng đợi
 - Q0, dùng RR với quantum 8 ms
 - Q1, dùng RR với quantum 16 ms
 - Q2, dùng FCFS





4.8. Multilevel Feedback Queue

Các vấn đề với Multilevel Feedback Queue

- Định thời dùng multilevel feedback queue đòi hỏi phải giải quyết các vấn đề sau:

Số lượng hàng đợi bao nhiêu là thích hợp?

Dùng giải thuật định thời nào ở mỗi hàng đợi?

Làm sao để xác định thời điểm cần chuyển một tiến trình đến hàng đợi cao hơn hoặc thấp hơn?

Khi tiến trình yêu cầu được xử lý thì đưa vào hàng đợi nào là hợp lý nhất?



CÁC GIẢI THUẬT ĐỊNH THỜI

4.9. So sánh các giải thuật

04.



4.9. So sánh các giải thuật

- Giải thuật định thời nào là tốt nhất?
- Câu trả lời phụ thuộc các yếu tố sau:
 - Tần suất tải việc (System workload).
 - Sự hỗ trợ của phần cứng đối với dispatcher.
 - Sự tương quan về trọng số của các tiêu chuẩn định thời như response time, hiệu suất CPU, throughput,...
 - Phương pháp định lượng so sánh.



Câu hỏi ôn tập chương 4

- Tại sao phải định thời? Nêu các bộ định thời và mô tả về chúng?
- Các tiêu chuẩn định thời CPU?
- Có bao nhiêu giải thuật định thời? Kể tên?
- Mô tả và nêu ưu điểm, nhược điểm của từng giải thuật định thời? FCFS, SJF, SRTF, RR, Priority Scheduling, HRRN, MQ, MFQ.



Bài tập 1

Sử dụng các giải thuật FCFS, SJF, SRTF, Priority -Pre, RR (10) để tính các giá trị thời gian đợi, thời gian đáp ứng, thời gian hoàn thành trung bình và vẽ giản đồ Gantt.

Với RR, điều gì sẽ xảy ra khi P5 vào tại thời điểm P1 vừa hết quantum time?

Process	Arrival	Burst	Priority
P1	0	20	20
P2	25	25	30
P3	20	25	15
P4	35	15	35
P5	10	35	5
P6	15	50	10



Bài tập 2

Cho 5 tiến trình với thời gian vào và thời gian cần CPU tương ứng như bảng sau:

Process	Arrival	Burst
P1	0	10
P2	2	29
P3	4	3
P4	5	7
P5	7	12

Vẽ giản đồ Gantt và tính thời gian đợi trung bình, thời gian đáp ứng trung bình, thời gian hoàn thành trung bình cho các giải thuật sau:

- FCFS
- SJF preemptive
- RR với quantum time = 10



Bài tập 3

Xét tập các tiến trình sau (với thời gian vào hệ thống, thời gian yêu cầu CPU và độ ưu tiên kèm theo) :

Process	Arrival	Burst	Priority
P1	0	10	3
P2	1	3	2
P3	2	2	1
P4	3	1	2
P5	4	5	4

Vẽ giản đồ Gantt và tính thời gian đợi trung bình, thời gian đáp ứng trung bình, thời gian hoàn thành trung bình cho các giải thuật sau:

- SJF Preemptive
- RR với quantum time = 2
- Điều phối theo độ ưu tiên độc quyền (độ ưu tiên $1 > 2 > \dots$)



Bài tập 4

Vẽ giản đồ Gantt và tính thời gian đợi trung bình, thời gian đáp ứng trung bình, thời gian hoàn thành trung bình cho các giải thuật sau:

- a. FCFS, SJF
- b. RR với quantum time = 10

Process	Burst Time	Arrival Time
P1	10	5
P2	29	2
P3	3	0
P4	7	1
P5	12	7



Bài tập 5

Cho 4 tiến trình với thời gian vào hệ thống (arrival time) và burst time tương ứng như sau:

Process	Arrival Time	CPU Burst Time
P1	0	12
P2	2	7
P3	3	5
P4	5	9

Vẽ giản đồ Gantt và tính thời gian đợi trung bình, thời gian đáp ứng trung bình, thời gian hoàn thành trung bình cho các giải thuật sau:

- Shortest Remaining Time First (SRTF)
- Round Robin (RR) với quantum = 4



Bài tập 6

Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào Ready List và thời gian cần CPU tương ứng như bảng sau:

Process	Arrival Time	CPU Burst Time
P1	0	8
P2	2	19
P3	4	3
P4	5	6
P5	7	12

Vẽ giản đồ Gantt và tính thời gian đợi trung bình, thời gian đáp ứng trung bình, thời gian hoàn thành trung bình cho các giải thuật sau:

- FCFS
- SJF preemptive
- RR với quantum time = 6



THẢO LUẬN



Thực hiện bởi Trường Đại học Công nghệ Thông tin, ĐHQG-HCM