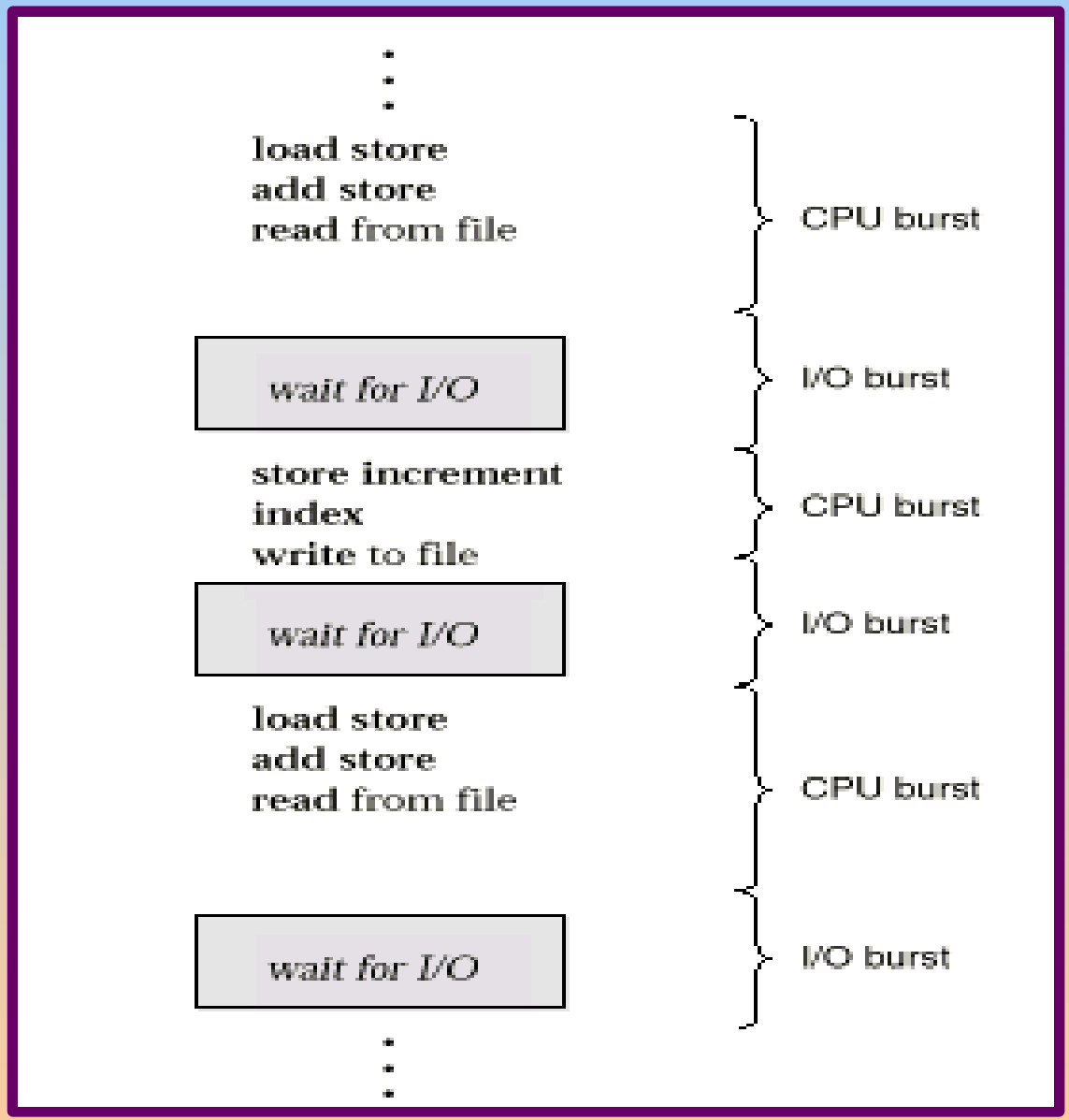
Escalonamento da CPU

- Conceitos básicos
- Critérios de escalonamento
- Algoritmos de escalonamento

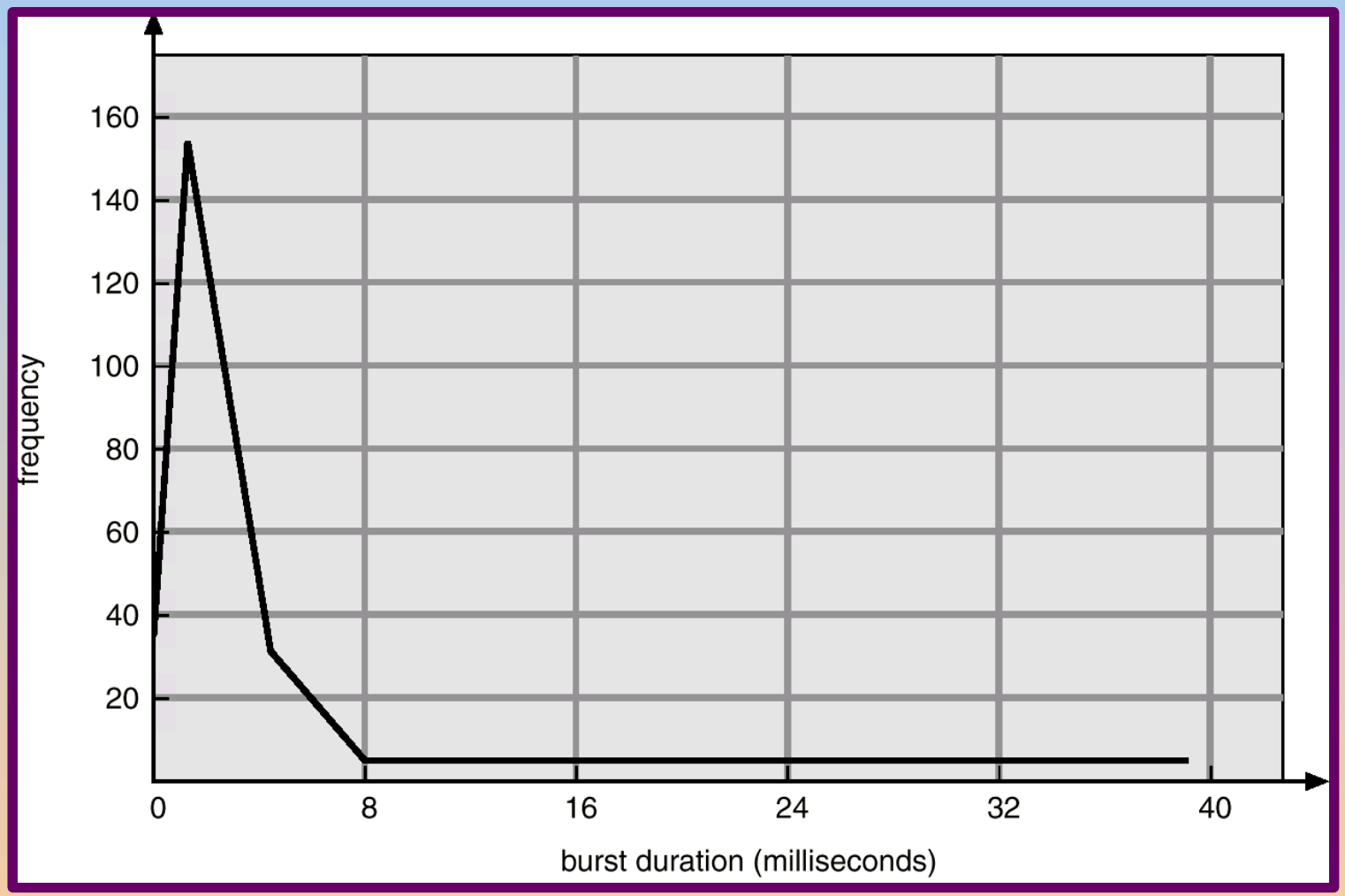
Conceitos básicos

- Objetivo básico da multiprogramação: utilização máxima da CPU
- CPU–I/O Burst Cycle(ciclo de uso de CPU e E/S) – Execução de um processo consiste de um ciclo alternante entre execução na CPU e espera por E/S.
- Distribuição de ciclos de CPU: dependentes do tipo de processo (CPU-bound ou I/O Bound)

Sequência alternante de ciclos de CPU e E/S



Histograma de duração de ciclos de uso da CPU



Escalonador de CPU

- Seleciona entre processos na memória que estão prontos para executar e aloca a CPU para um deles
- Decisões de escalonamento de CPU podem ser necessárias quando um processo:

1. Muda do estado de execução para o estado de espera
2. Muda do estado de execução para o estado de pronto
3. Muda do estado de espera para o estado de pronto
4. Termina

- O escalonamento em 1 e 4 é não-preemptivo, pois não há nenhuma escolha a ser feita.
- Em 2 e 3, o escalonamento é preemptivo.

Dispatcher(Despachante)

- Módulo dispatcher dá controle da CPU para o processo selecionado pelo escalonador de CPU. Isto envolve:
  - Mudança de contexto
  - Mudança para o modo usuário
  - Desvio para o endereço adequado do programa do usuário, para reiniciar o programa

- *Dispatch latency(latência de despacho)* – intervalo de tempo que o dispatcher leva entre parar um processo e começar a execução de outro.

Critérios de escalonamento

- Utilização de CPU – manter a CPU o mais ocupada possível
- Produtividade – número de processos que terminam sua execução por unidade de tempo
- Tempo de processamento – quantidade de tempo para executar um processo particular
- Tempo de espera – quantidade de tempo que um processo espera na fila de processos prontos.
- Tempo de resposta – quantidade de tempo requerida desde que uma requisição foi submetida até que a primeira resposta seja produzida, não necessariamenta saída do processo. É uma medida importante em sistemas de tempo compartilhado.

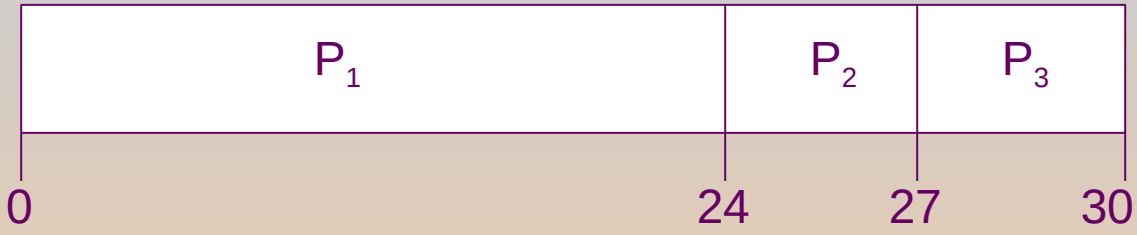
Critérios de otimização

- Maximizar utilização da CPU
- Maximizar produtividade
- Minimizar tempo de processamento
- Minimizar tempo de espera
- Minimizar tempo de resposta

Escalonamento First-Come, First-Served (FCFS)
Primeiro que chega, primeiro que é servido

Processo	Fase de uso de CPU(duração em ms)
P_1	24
P_2	3
P_3	3

- Suponha que os processos chegaram na ordem: P_1 , P_2 , P_3
  O Diagrama de Gantt para este escalonamento é:



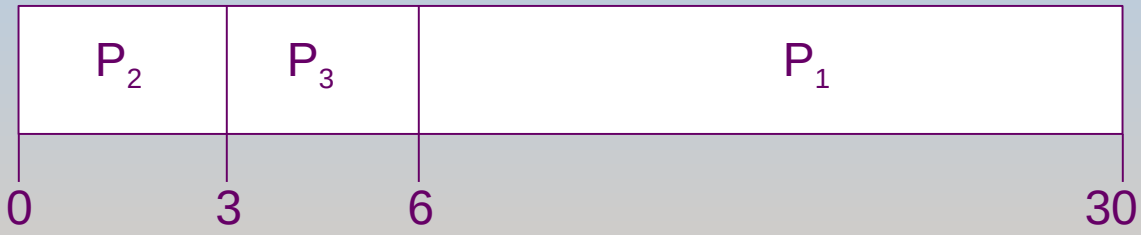
- Tempo de espera para P_1 = 0; P_2 = 24; P_3 = 27
- Tempo médio de espera: (0 + 24 + 27)/3 = 17

Escalonamento FCFS (Cont.)

Suponha que os processos chegaram na ordem

P_2 , P_3 , P_1 .

- O Diagrama de Gantt para este escalonamento é:



- Tempo de espera $P_1 = 6; P_2 = 0, P_3 = 3$
- Tempo médio de espera: (6 + 0 + 3)/3 = 3
- Muito melhor que o caso anterior.
- *Processos pequenos ficam prejudicados se estão depois de processos grandes, pois FCFS não faz distinção entre a duração da fase de uso da CPU*

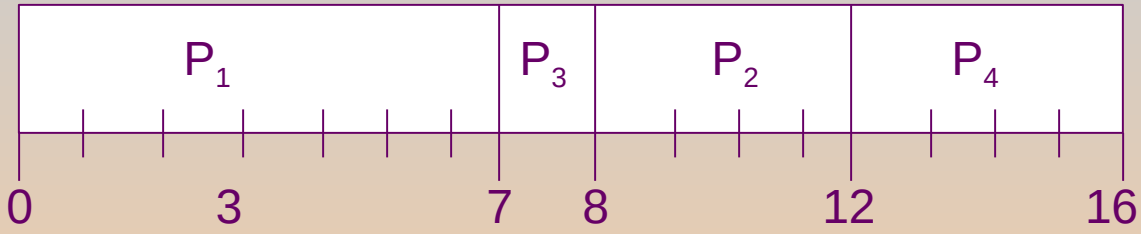
Escalonamento Shortest-Job-First(SJF)
Processo menor primeiro

- Associa a cada processo a duração da próxima fase de uso da CPU. O escalonamento escolhe o processo com menor duração da próxima fase.
- Dois esquemas básicos:
  - Não-preemptivo – uma vez que a CPU é alocada para um processo, ele a mantém até completar sua fase de utilização da CPU.
  - Premptivo – se um novo processo chega com uma duração de ciclo de CPU menor que o tempo restante para o processo corrente em execução, a CPU é alocada para este novo processo. Este esquema é conhecido como Shortest-Remaining-Time-First (SRTF).
- SJF é ótimo – ele minimiza o tempo médio de espera para um determinado conjunto de processos.

Exemplo de SJF não-preemptivo

Processo	Hora da Chegada	Fase de uso da CPU
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (não-preemptivo)

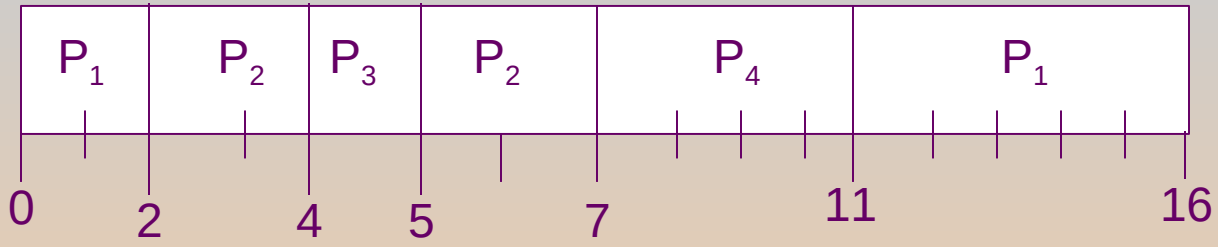


- Tempo médio de espera = (0 + 6 + 3 + 7)/4 = 4

Exemplo de SJF preemptivo

Processo	Hora da Chegada	Fase de uso da CPU
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (preemptivo)



- Tempo médio de espera = (9 + 1 + 0 +2)/4 = 3

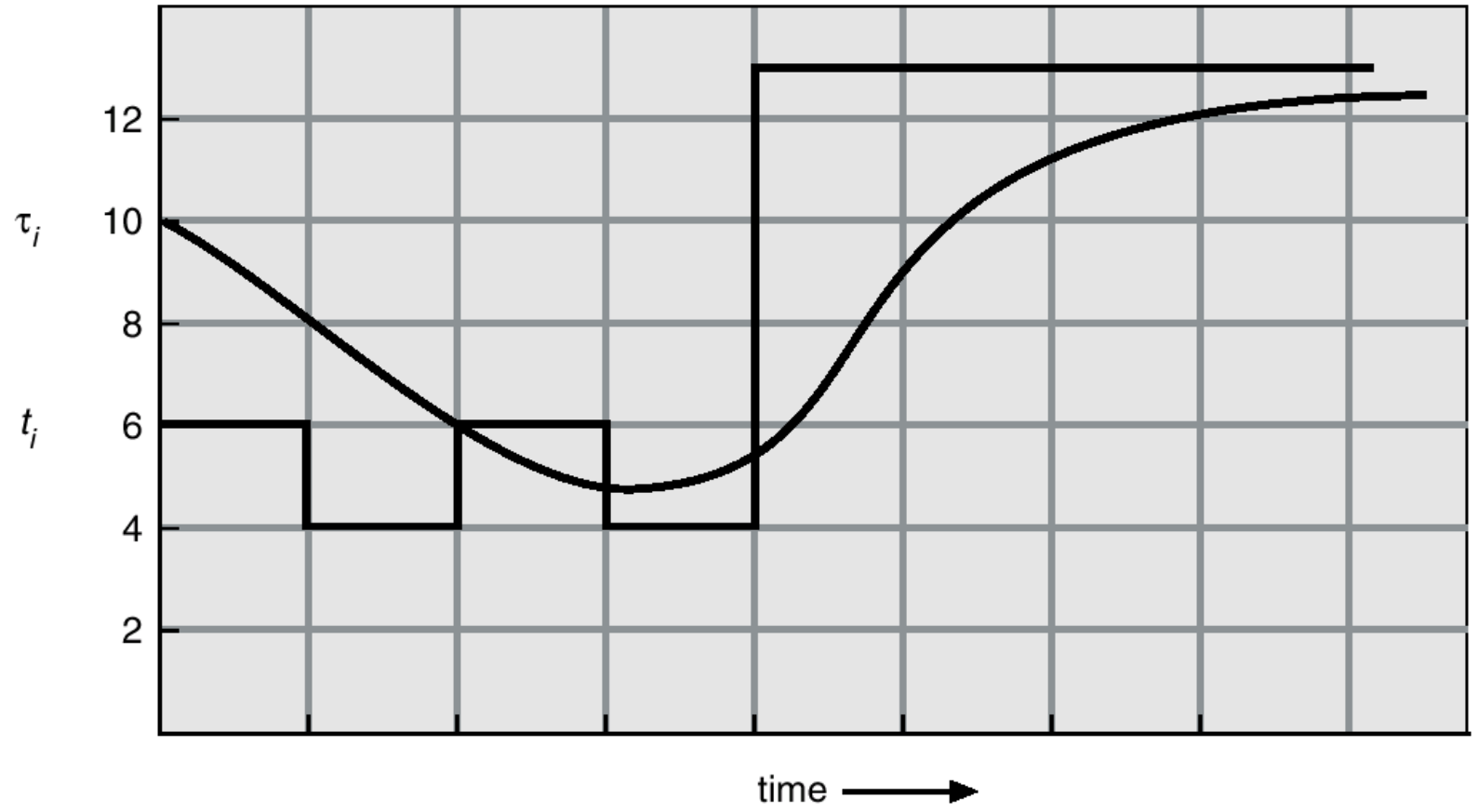
Determinando a previsão da próxima duração de uso da CPU

- Pode ser feita usando o tempo de uso prévio da CPU, usando média exponencial.

1. t_n = duração da n – ésima fase de CPU
2. τ_{n+1} = tempo previsto para a próxima fase de CPU
3. $\alpha, 0 \leq \alpha \leq 1$
4. Defina :

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

Previsão do tempo da fase da próxima utilização da CPU



CPU burst (t_i)		6	4	6	4	13	13	13	. . .
"guess" (τ_i)	10	8	6	6	5	9	11	12	. . .

Exemplo de média exponencial

- $\alpha = 0$
  - $\tau_{n+1} = \tau_n$
  - Comportamento recente do processo é desprezado
- $\alpha = 1$
  - $\tau_{n+1} = t_n$
  - Somente o comportamento recente do processo é levado em conta.
- Se expandirmos a fórmula, obtemos:

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

- Como ambos α and $(1 - \alpha)$ são menores ou iguais a 1, cada termo sucessivo tem um peso menor que seu predecessor.

Escalonamento por Prioridade

- Uma prioridade (número inteiro) é associada com cada processo.
- A CPU é alocada ao processo com a mais alta prioridade (quanto menor o número inteiro maior a prioridade)
  - Preemptiva
  - Não-preemptiva
- SJF é um escalonamento com prioridade onde a prioridade é a predição da duração da próxima fase de CPU
- Problema ≡ Starvation(inanição) – processos com baixa prioridade podem nunca executar.
- Solução ≡ Aging(envelhecimento) – a prioridade dos processos cresce com o passar do tempo.

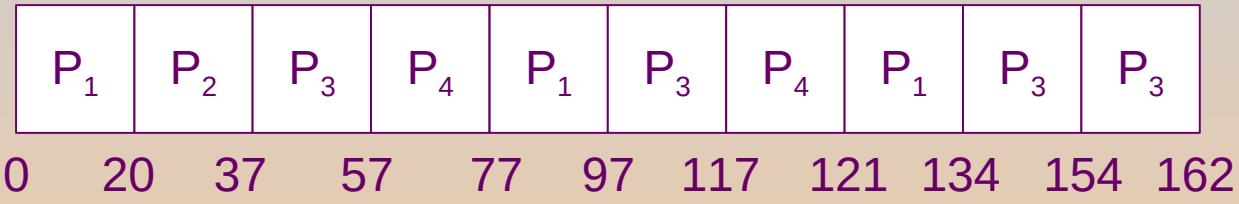
Escalonamento Round Robin (RR)
Alocação circular

- Cada processo recebe uma pequena unidade de tempo da CPU (quantum de tempo), usalmente 10-100 milisegundos. Quando este tempo termina, o processo retorna para o final da fila de processos prontos.
- Se existem n processos na fila de processos prontos e quantum de tempo é q, então cada processo recebe 1/n do tempo da CPU em intervalos de, no máximos, q unidades de tempo. Nenhum processo espera mais que (n-1)q unidades de tempo.
- O desempenho depende do tamanho de *q* :
  - *q grande* $\Rightarrow$ FIFO (First-In First-Out)
  - *q* pequeno $\Rightarrow$ q precisa ser grande com relação à mudança de contexto pois, caso contrário, o overhead é muito alto.

Exemplo de RR com quantum de tempo = 20

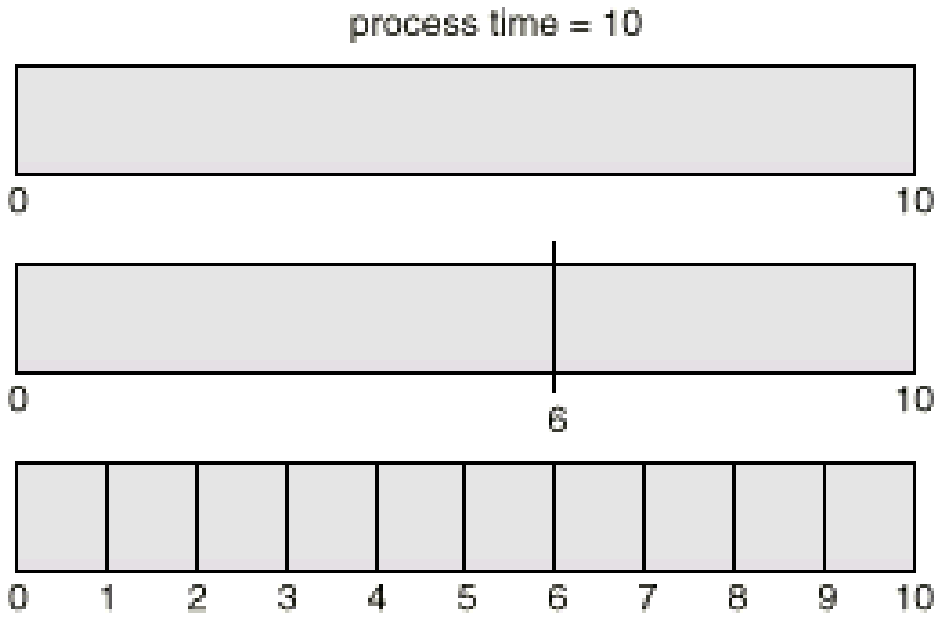
Processo	Fase de uso da CPU
P_1	53
P_2	17
P_3	68
P_4	24

- O diagrama de Gantt fica como :



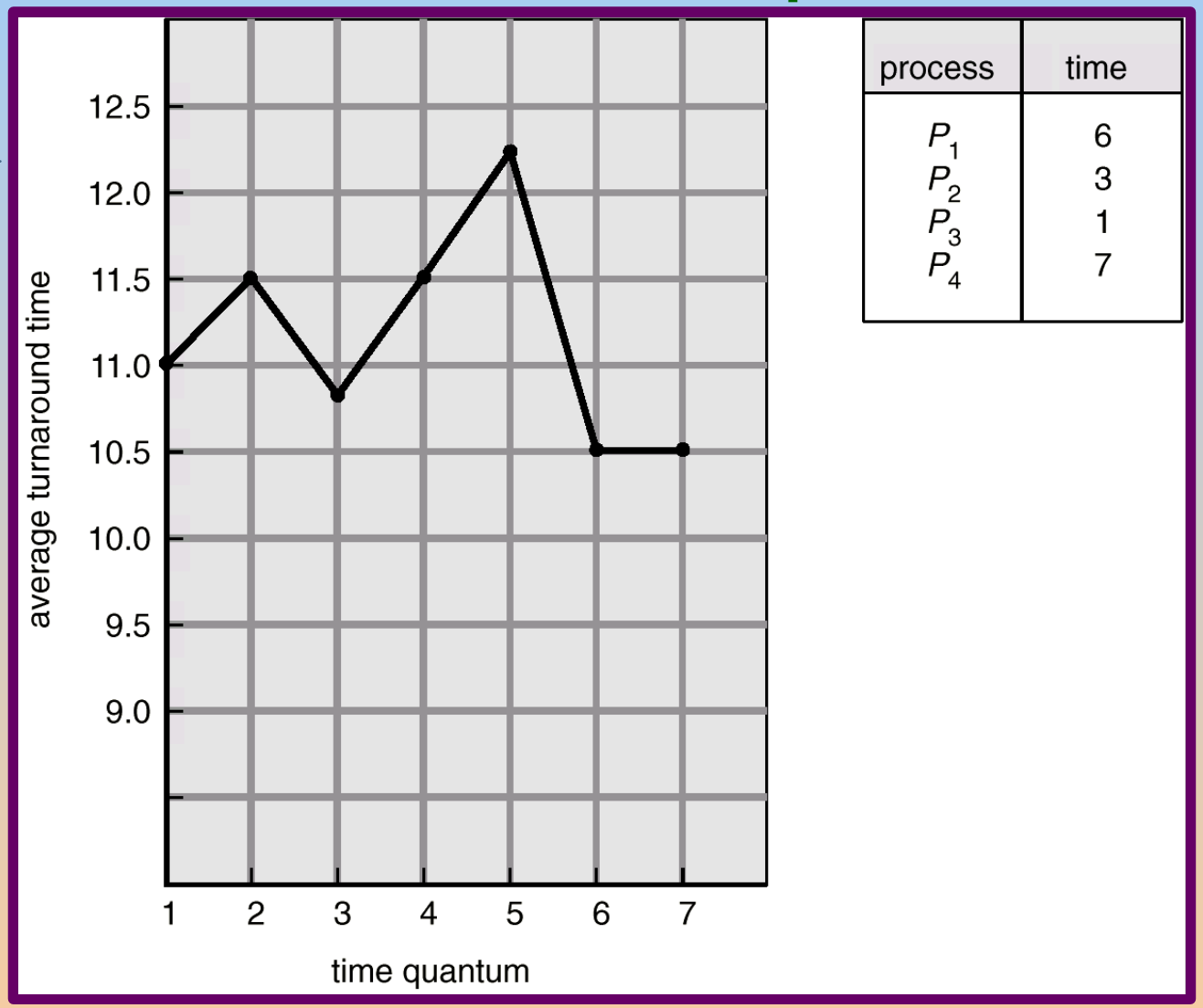
- Média de tempo de processamento é maior que SJF, mas tempo de resposta é melhor.

Quantum de tempo e mudança de contexto



quantum	context switches
12	0
6	1
1	9

Tempo de processamento varia com o quantum de tempo

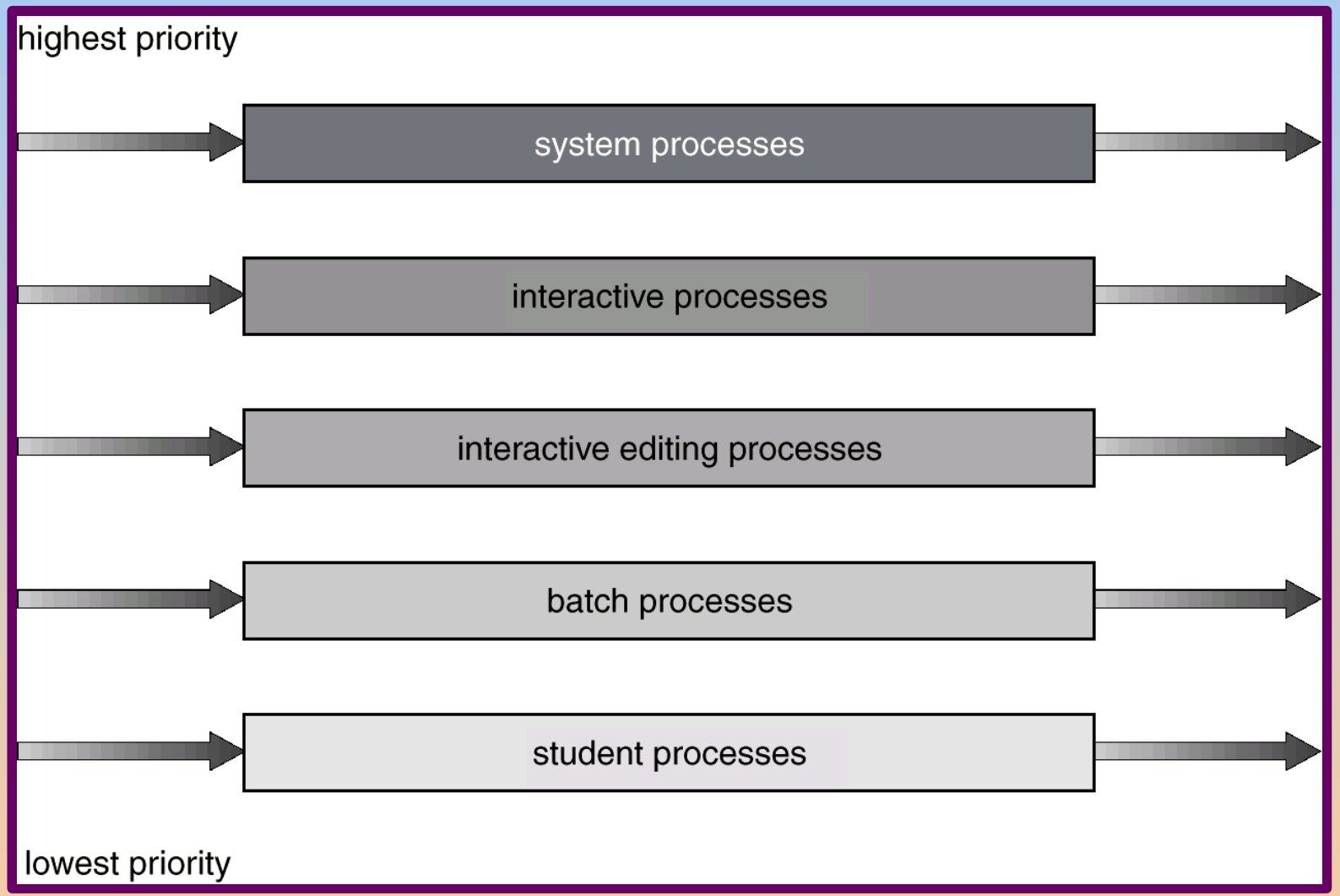


process	time
P_1	6
P_2	3
P_3	1
P_4	7

Alocação com várias filas

- Fila de processos prontos é dividida em duas classes de filas separadas: foreground (processos interativos) e background (batch)
- Cada classe tem seu próprio algoritmo de escalonamento:
  foreground – RR
  background – FCFS
- Escalonamento precisa ser feito entre as duas classes:
  - Escalonamento com prioridade fixa, isto é, tanto processos foreground quanto background tem a mesma prioridade. Existe a possibilidade de starvation.
  - Time slice(fatia de tempo) – cada classe recebe uma certa quantidade de tempo da CPU. Por exemplo, 80% do tempo para processos foreground sob controle de RR e 20% de tempo para processos background em FCFS.

Escalonamento com múltiplas filas



Múltiplas filas com realimentação(feedback)

- Um processo pode ser movido entre as várias filas; envelhecimento(aging) de processos pode ser implementado desta maneira.
- Um escalonador com múltiplas filas com realimentação é definido pelos seguintes parâmetros:
  - Número de filas
  - Algoritmos de escalonamento para cada fila
  - Método usado para determinar quando transferir um processo para uma fila de prioridade mais alta
  - Método usado para determinar quando transferir um processo para uma fila de prioridade mais baixa
  - Método usado para determinar em qual fila o processo irá entrar quando ele precisar usar a CPU.

Exemplos de múltiplas filas com realimentação

- Três filas:
  - Q_0 – quantum de tempo de 8 ms (RR)
  - Q_1 – quantum de tempo de 16 ms (RR)
  - Q_2 – FCFS
- Escalonamento
  - Um novo processo entra na fila Q_0 vindo da fila Q_2. Quando o processo ganha a CPU, ele recebe 8 ms. Se ele não terminar em 8 ms, ele é movido para a fila Q_1.
  - Em Q_1 , o processo recebe 16 ms. Se ele não completa a execução em 16 ms, ele é movido novamente para a fila Q_2.

Várias filas com realimentação