Sistemas Operacionais I

Gerências de Processos: Escalonamento de CPU

Prof. Alexandre Duarte : http://alexandrend.com
Centro de Informática | Universidade Federal da Paraíba

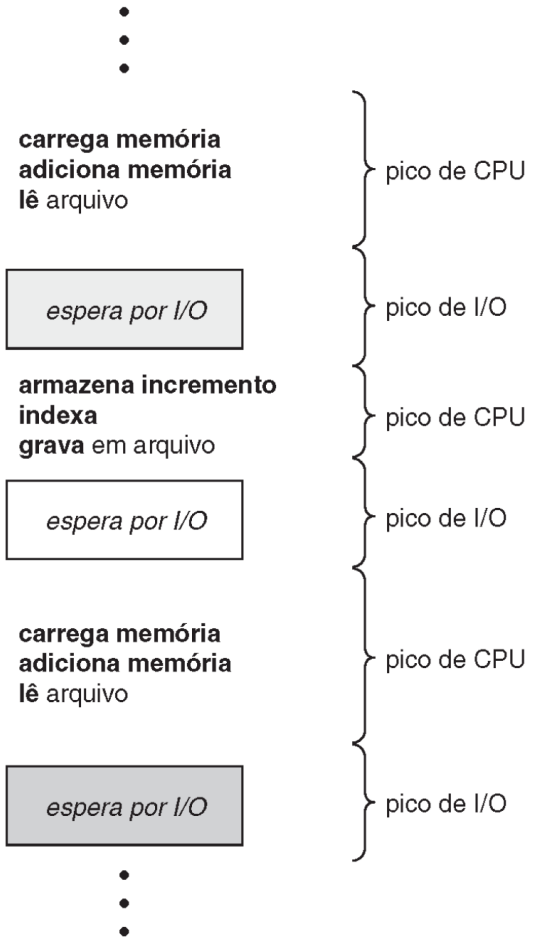
Estes slides são baseados no material que acompanha o livro Operating Systems Concepts de Silberschatz, Galvin and Gagne

Objetivos

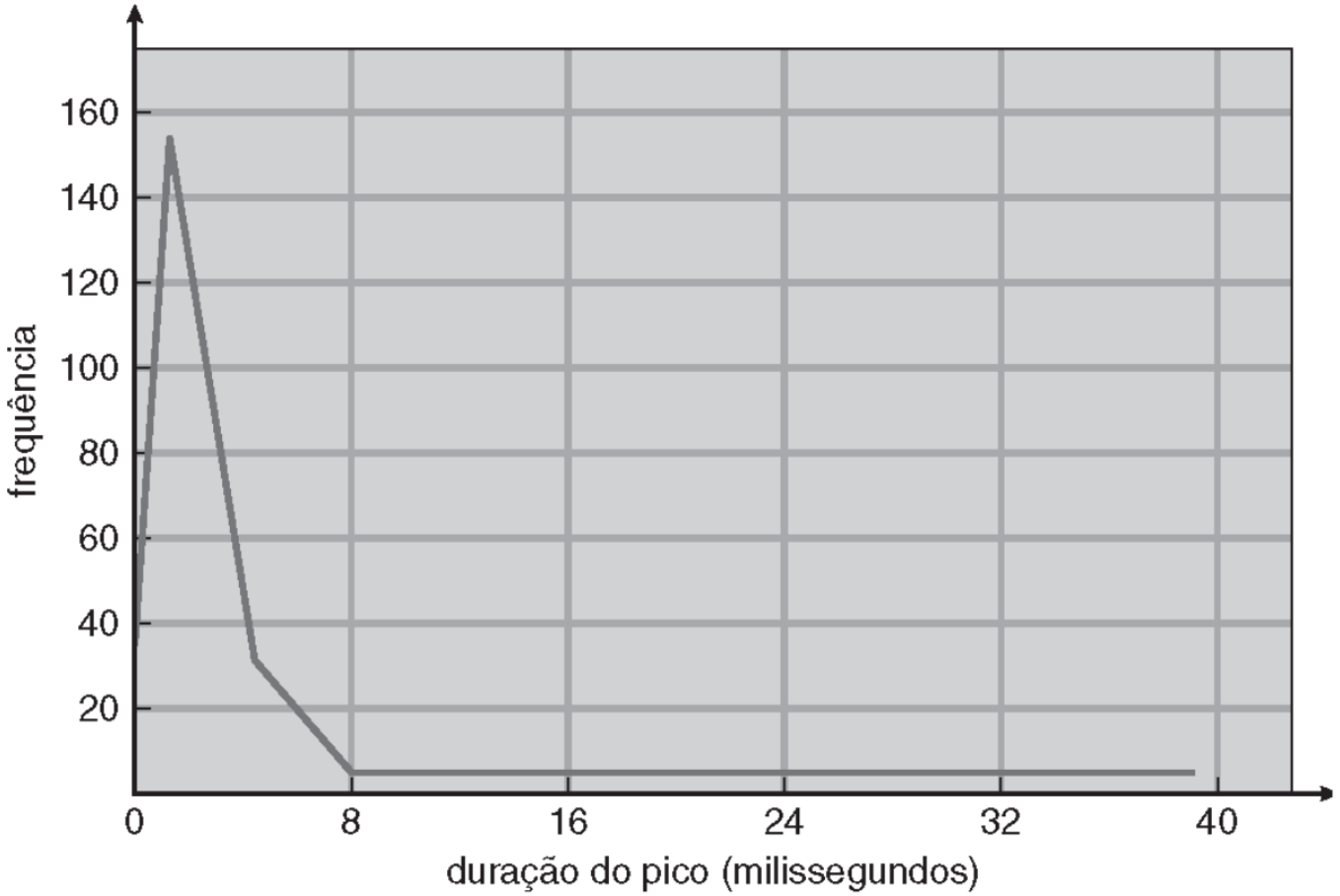
- Introduzir o conceito de escalonamento de CPU, base para os sistemas operacionais multiprogramados
- Descrever vários algoritmos de escalonamento de CPU
- Discutir os critérios de avaliação para selecionar um algoritmo de escalonamento de CPU para um determinado sistema

Conceitos básicos

- Utilização máxima de CPU é obtida com multiprogramação
- Ciclos de rajadas de CPU–E/S
  - A execução de um processo é um ciclo de rajadas alternadas de CPU e espera por E/S



Histograma com as durações dos picos de CPU



Escalonador de CPU

- Seleciona um entre os processos em memória prontos para executar e aloca a CPU para ele

- O escalonamento de CPU pode ocorrer quando um processo:
  1. muda do estado executando para esperando
  2. muda do estado executando para pronto
  3. muda do estado esperando para pronto
  4. é finalizado

- Se o escalonamento ocorre apenas nas condições 1 e 4, então o esquema de escalonamento é **não-preemptivo**

- Em 2 e 3 é dito **preemptivo**. Requer dispositivos de hardware especiais. Ex: timer

Despachante

- O módulo despachante transfere o controle da CPU para o processo selecionado pelo escalonador de curto prazo, o que envolve:
  - trocar contexto
  - alterar para o modo usuário
  - fazer um salto para o endereço de memória correto para que o processo seja reiniciado

- **Latência de despacho:** é o tempo necessário para o despachante parar um processo e reiniciar um outro

Critérios de escalonamento

- **Utilização de CPU:** manter a CPU o mais ocupada possível
- **Vazão:** # de processos que concluem sua execução por unidade de tempo
- **Tempo de retorno:** quantidade necessária de tempo para executar um processo
- **Tempo de espera:** quantidade de tempo que um processo aguardou na fila de prontos
- **Tempo de resposta:** quantidade de tempo entre a requisição de execução de um programa e a produção da primeira resposta (sistemas de compartilhamento de tempo)

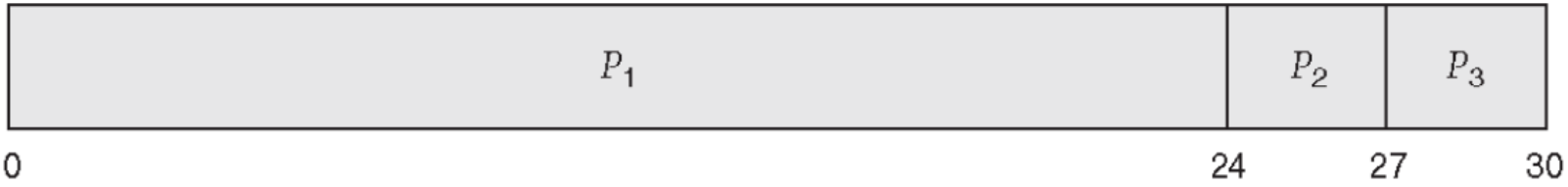
Critérios de otimização para algoritmos de escalonamento

- Utilização máxima de CPU
- Vazão máxima
- Tempo de retorno mínimo
- Tempo de espera mínimo
- Tempo de resposta mínimo

“Primeiro a Entrar, Primeiro a ser Atendido” (FCFS)

Processo	Duração do Pico
P_1	24
P_2	3
P_3	3

Suponha que os processos cheguem na ordem: P_1 , P_2 , P_3
Seriam escalonados como a seguir



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

Continuando com o FCFS

Agora, suponha que os processos cheguem na ordem P_2 , P_3 , P_1

O escalonamento seria

- Tempos de espera: $P_1 = 6; P_2 = 0; P_3 = 3$
- Tempo de espera médio: (6 + 0 + 3)/3 = 3
- Muito melhor que no caso anterior!!!
- FCFS não tem preempção
- Um processo limitado por CPU -> “Efeito comboio

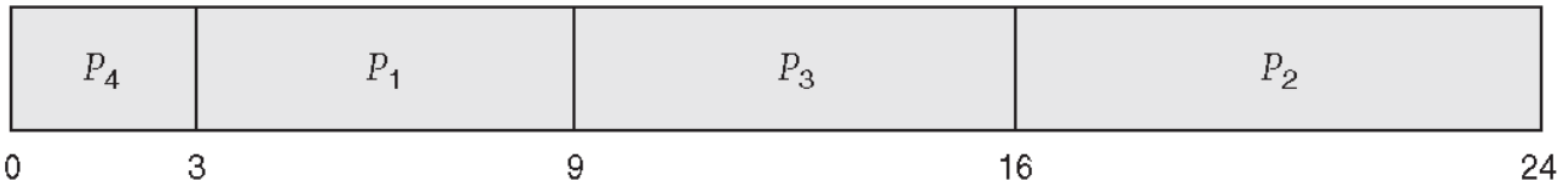
Menor Job Primeiro (SJF)

- Associa a cada processo a duração do seu próximo pico de CPU
- Usa essas durações para escalonar o processo com a menor duração de pico de CPU

Exemplo do SJF

Processo	Duração do Pico
P_1	6
P_2	8
P_3	7
P_4	3

O SJF escalonaria da seguinte forma:



- Tempo de espera médio = (3 + 16 + 9 + 0) / 4 = 7

Menor Job Primeiro (SJF)

- SJF é comprovadamente ótimo: dá o menor tempo médio de espera para um dado conjunto de processos

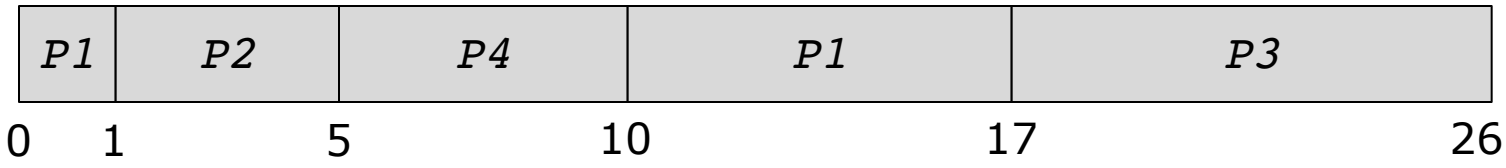
- Facilmente comprovado
  - Um curto antes do longo: O primeiro tem tempo de espera 0. Os tempos seguintes são definidos pelos anteriores

- A dificuldade reside em saber qual será a duração do próximo pico de CPU!

Menor Job Primeiro (SJF)

- Sem preempção
- Com preempção

Processo	Tempo de chegada	Tempo de pico
P1	0	8
P2	1	4
P3	2	9
P4	3	5



Escalonamento por prioridade

- Cada processo recebe um nível de prioridade
- A CPU é alocada para o processo com a maior prioridade (menor valor númerico)
  - Preemptivo
  - Não-preemptivo
- O SJF é um algoritmo de escalonamento por prioridade onde a prioridade é a duração do próximo pico de CPU
- Prioridades internas ou externas ao SO
- Problema ➡ **Starvation**: processos de baixa prioridade podem nunca rodar
- Solução ➡ **Envelhecimento**: a medida que o tempo passa a prioridade dos processos aumenta

Round Robin (RR)

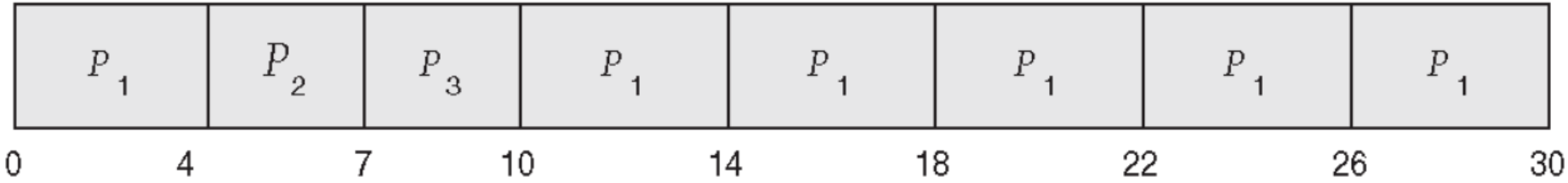
- Cada processo recebe uma pequena quantidade de tempo de CPU (*quantum*), usualmente entre 10 e 100 milissegundos.
  - Depois que esse tempo se esgota o processo é interrompido e inserido no fim da fila de prontos.

- Se existem *n* processos na fila de prontos e o quantum é *q*, então cada processo recebe 1/n do tempo de CPU em pedaços de tamanho máximo *q*.
  - Nenhum processo espera mais que do que (*n*-1)*q* unidades de tempo

Exemplo: RR com quantum = 4

Processo	Duração do Pico
P_1	24
P_2	3
P_3	3

- O escalonamento seria

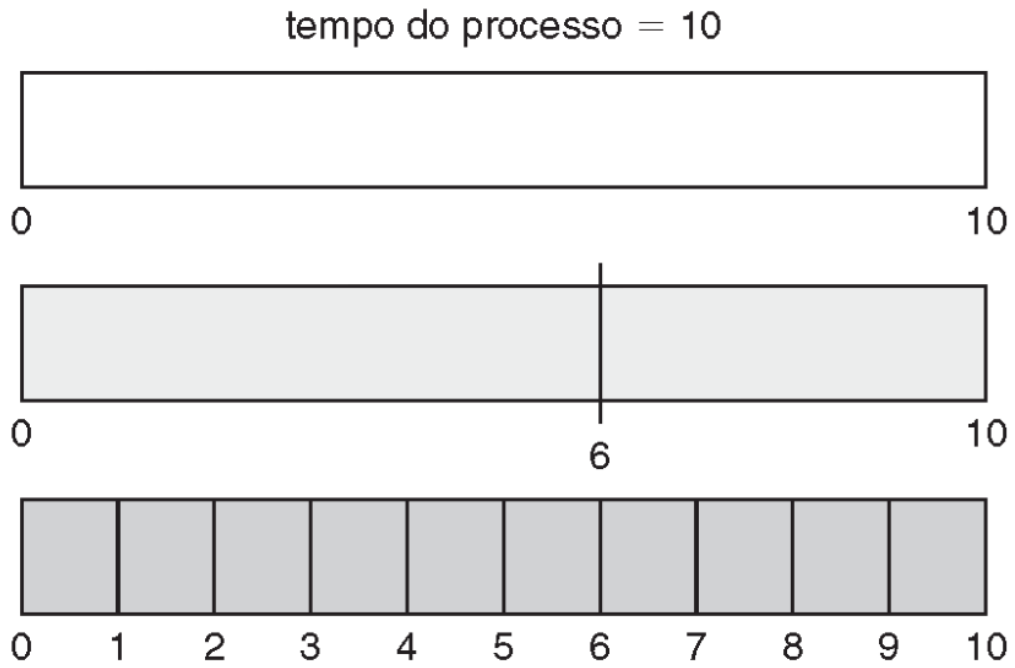


- Geralmente apresenta um tempo de retorno médio maior do que o do SFJ mas melhor tempo de resposta

Duração do quantum e trocas de contexto

Desempenho

- *q* grande ➡ FIFO
- *q pequeno* ➡ *q* precisa ser grande em relação ao tempo de troca de contexto, caso contrário o overhead será muito grande



quantum	mudanças de contexto
12	0
6	1
1	9

Exercício

Processo	P1	P2	P3	P4	P5
Pico de CPU	120	60	180	50	300

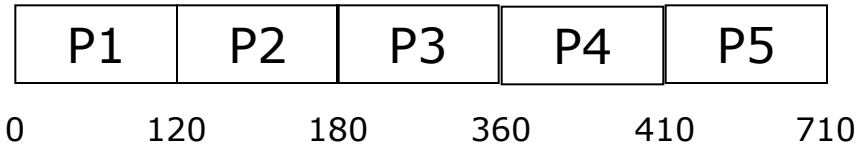
Desenha um diagrama de gantt mostrando o tempo de execução de cada processo para:

- FCFS
- SJF
- RR, quantum = 50

Exercício

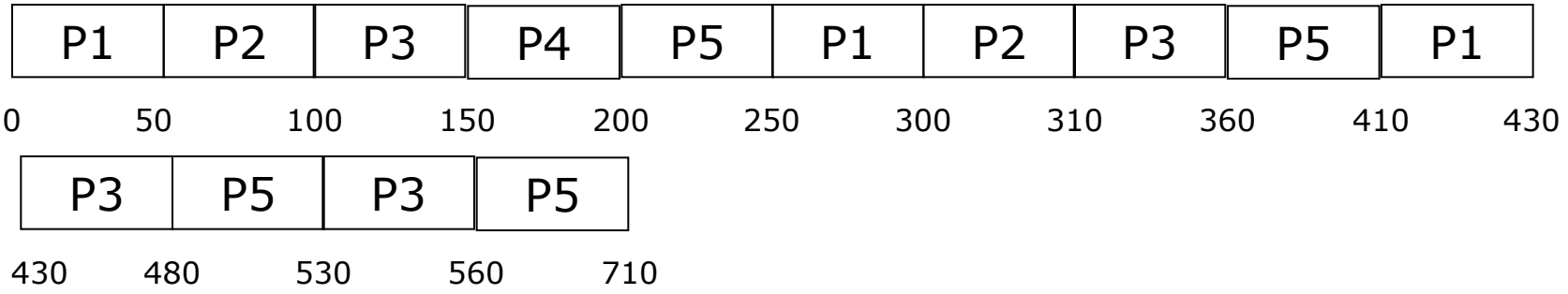
Processo	P1	P2	P3	P4	P5
Pico de CPU	120	60	180	50	300

FCFS



SJF

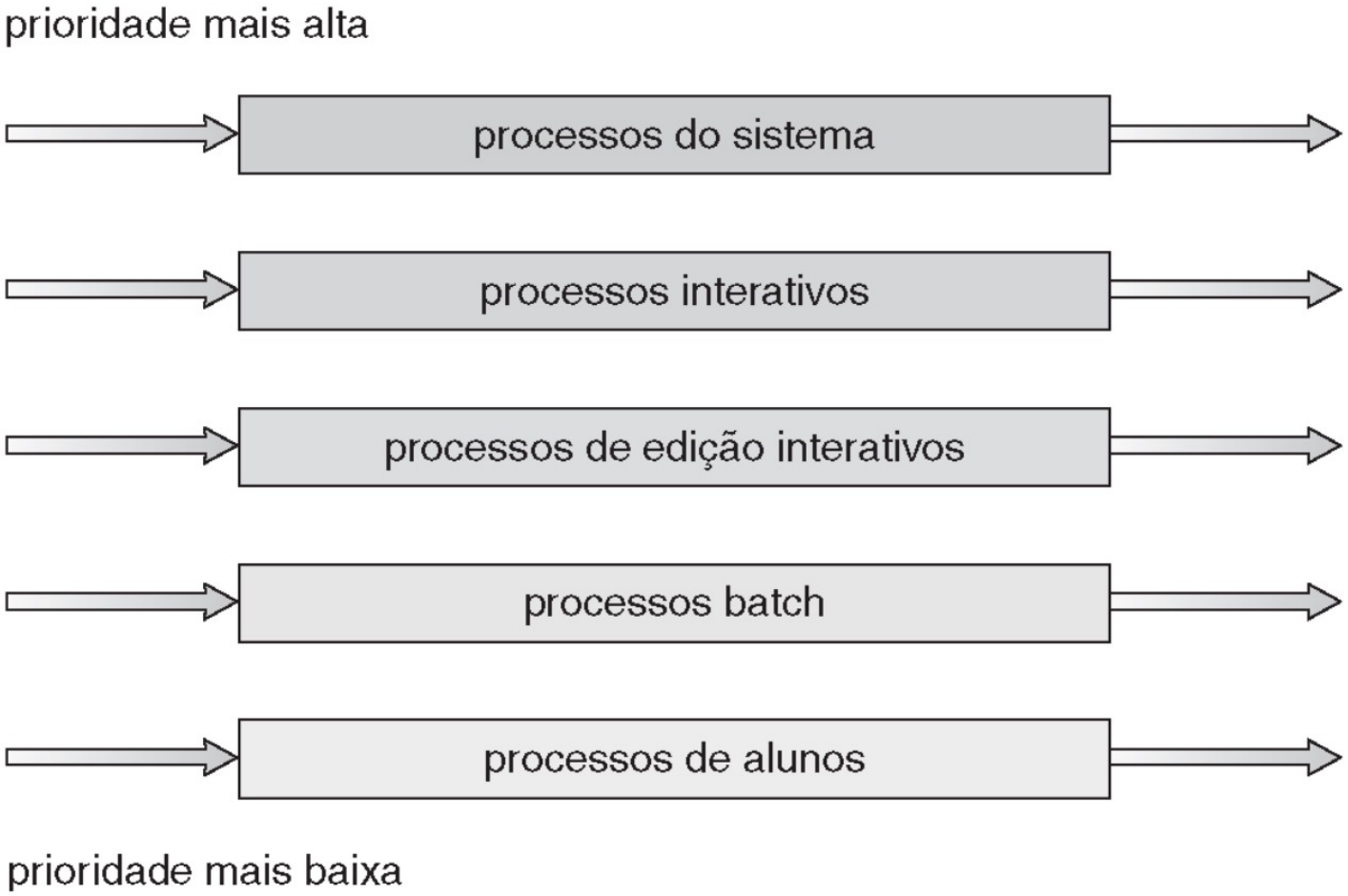
RR = 50 ~~P1~~ ~~P2~~ ~~P3~~ ~~P4~~ ~~P5~~



Filas de vários níveis

- A fila de prontos é particionada em várias filas separadas
  - Cada fila tem seu próprio algoritmo de escalonamento
    - Processos interativos: RR
    - Processos de lote: FCFS
- O escalonamento precisa ser feito também entre as filas
  - Prioridade fixa: escalona todos da primeira fila para depois passar para a próxima.
    - Possibilidade de haver *starvation*.
  - Fatia de tempo: cada fila recebe uma fatia de tempo da CPU para dividir entre seus processos
    - 80% do tempo para processos interativos utilizando RR
    - 20% para processos de lote utilizando FCFS

Escalonamento com várias filas



Filas com feedback

- Um processo pode mudar de fila
  - Forma de implementar envelhecimento

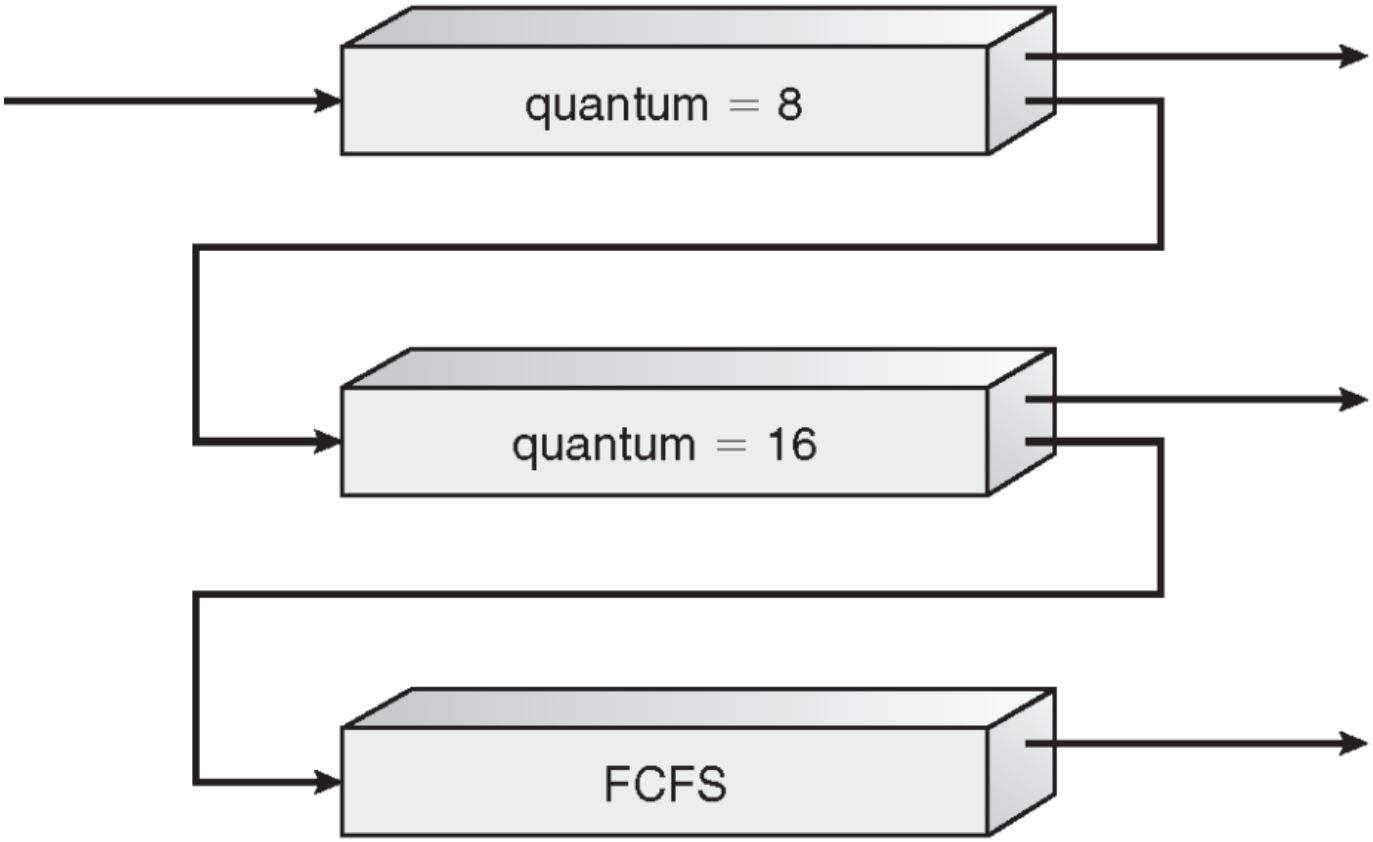
- Escalonador de várias filas com feedback é definido pelos seguintes parâmetros:
  - número de filas
  - algoritmo de escalonamento para cada fila
  - método para determinar em que fila um processo deve ser inserido
  - método para determinar quando promover um processo
  - método para determinar quando rebaixar um processo

Exemplo de escalonamento com várias filas com feedback

- Três filas
  - Q_0 – RR com quantum de 8 milissegundos
  - Q_1 – RR com quantum de 16 milissegundos
  - Q_2 – FCFS

- Escalonamento
  - Um novo job entra na fila Q_0 e é servido em ordem FIFO. Quando ele recebe a CPU, em tem 8 milissegundos. Se ele não terminar é então movido para a fila Q_1.
  - Os jobs da fila Q_1 também são servidos em ordem FIFO, recebendo 16 milissegundos adicionais. Se ainda assim não terminar é movido para a fila Q_2.

Exemplo de escalonamento com várias filas com feedback



Escalonamento com múltiplos processadores

- O escalonamento de CPU é mais complexo quando múltiplas CPUs estão disponíveis
  - **Processadores homogêneos** em um multiprocessador
- Cada processador pode ter sua própria fila de prontos (compartilhamento de carga)
  - Problema: processador ocioso
- Todos os processos ficam em uma fila de prontos comum
  - **Multiprocessamento simétricos:** cada processador faz o seu próprio escalonamento
  - **Multiprocessamento assimétrico:** apenas um processador acessa as estruturas de dados do sistema, fila de prontos, escalonamento e E/S. diminuindo a necessidade de compartilhamento de dados.

Exercício

- SJF com preempção. Desenhe diagrama de Gantt e calcule (Tresposta, Tretorno e Tespera).

Processo	Tempo de chegada	Tempo de pico
P1	0	8
P2	1	6
P3	2	9
P4	3	2

Exercício

- Duas filas sem feedback F1 (RR) e F2 (FCFS), prioridade fixa, quantum = 4, sem preempção, P(F1) > P(F2). Desenhe diagrama de Gantt e calcule (Tresposta, Tretorno e Tespera).

Processo	Tempo de chegada	Fila	Tempo de pico
P1	0	FCFS	4
P2	2	RR	4
P3	3	FCFS	2
P4	6	FCFS	7
P5	7	RR	10
P6	11	RR	7

Exercício

- Duas filas com feedback F1 (RR) e F2 (FCFS), prioridade fixa, quantum = 4, sem preempção, P(F1) > P(F2). Executa apenas uma vez na F1, senão passa para F2. Desenhe diagrama de Gantt e calcule (Tresposta, Tretorno e Tespera).

Processo	Tempo de chegada	Fila	Tempo de pico
P1	0	FCFS	4
P2	2	RR	4
P3	3	FCFS	2
P4	6	FCFS	7
P5	7	RR	10
P6	11	RR	7

Exercício

- Três filas com feedback F1(RR/4), F2 (RR/6) e F3 (FCFS), prioridade fixa, sem preempção. Executa apenas uma vez na fila de maior prioridade, senão passa para a fila seguinte. Desenhe diagrama de Gantt e calcule o tempo médio de resposta, retorno e espera.

Processo	Tempo de chegada	Fila	Tempo de pico
P1	0	FCFS	6
P2	1	RR/4	12
P3	7	RR/6	8
P4	15	FCFS	4
P5	18	FCFS	2
P6	22	RR/4	8