Curso de Ciência da Computação
Pontifícia Universidade Católica de Minas Gerais

Sistemas Operacionais

Capítulo VI - Escalonamento de CPU

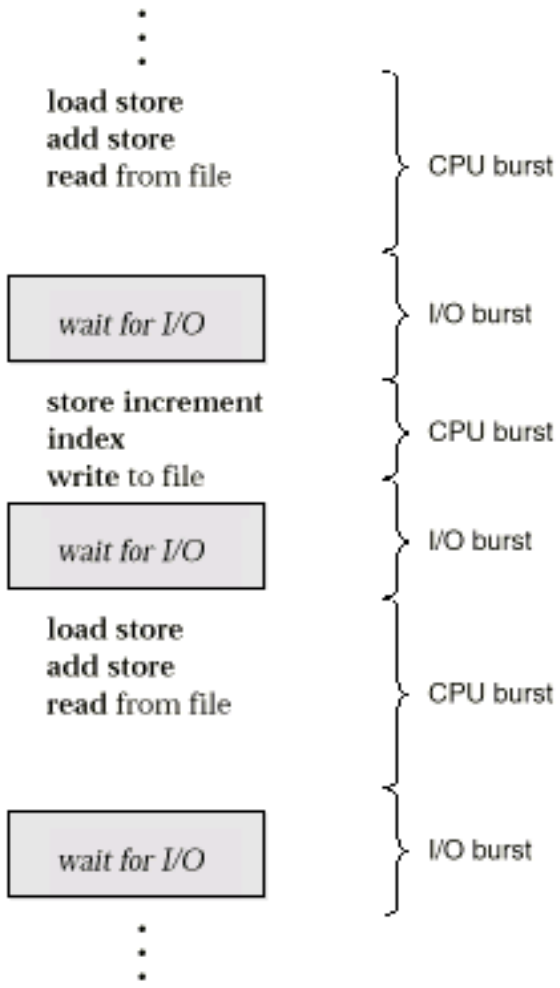
Conceitos Básicos

- A utilização máxima de CPU é obtida através de multiprogramação. Com um só processador, nunca haverá mais de um processo em execução.

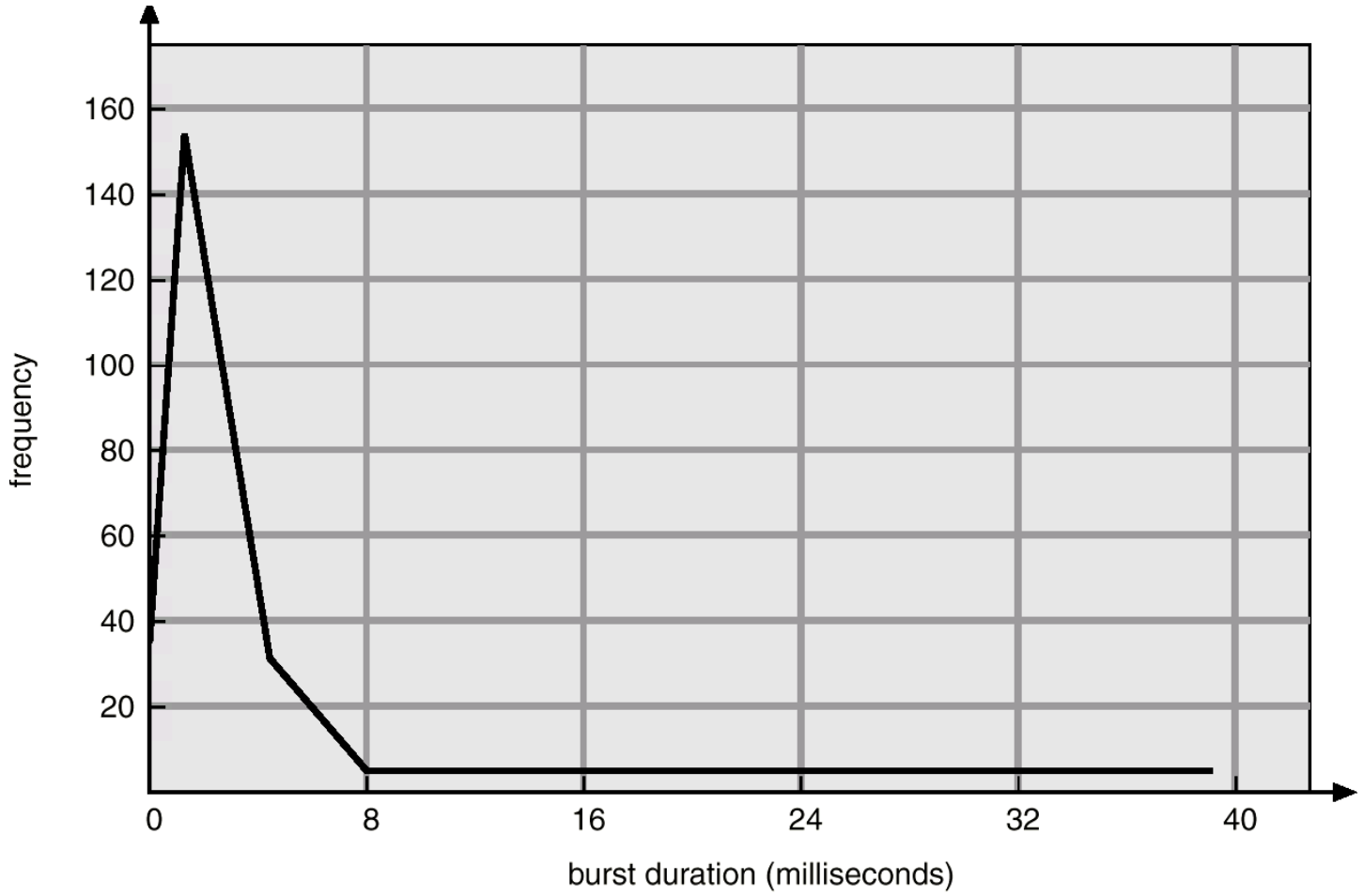
- Ciclos de surtos de CPU-I/O - A execução de um processo consiste na alternância entre um *ciclo* de execução de CPU e um ciclo de espera de I/O.

- Uma distribuição de surtos de CPU pode ser observada.

Seqüência Alternada de Surtos de CPU e I/O



Histograma de Tempo de Surtos de CPU



Escalonador de CPU

- Seleciona um dentre os processos na memória prontos a serem executados e aloca a CPU para ele.
- Decisões de escalonamento de CPU ocorrem quando um processo:
  1. Passa do estado de execução para espera.
  2. Passa do estado de execução para pronto.
  3. Passa do estado de espera para pronto.
  4. Termina.
- Quando o escalonamento ocorrem apenas **nos** casos 1 e 4 são chamados *não-preemptivos - o processo não é interrompido:*
  - Windows 3.x.
- Os demais são chamados *preemptivos:*
  - *Unix.*

Dispatcher

- O módulo de *dispatcher* (executor) dá o controle da CPU ao processo selecionado pelo escalonador de curto prazo. Isto envolve:
  - a mudança do contexto
  - a mudança para o modo usuário
  - pular para a posição adequada no programa do usuário e reiniciar este programa

- *Latência de dispatch* – tempo necessário ao dispatcher para interromper um processo e iniciar a próxima execução.

Critérios de Escalonamento

- Características para comparação de algoritmos:
  - Utilização de CPU - manter a CPU o mais ocupada possível
  - Throughput - # de processos que completam sua execução por unidade de tempo
  - Tempo de retorno - quantidade de tempo para executar um processo
  - Tempo de espera - quantidade de tempo que um processo gasta na fila de processos prontos
  - Tempo de resposta - quantidade de tempo gasto entre a requisição e a produção da primeira resposta

Critérios de Otimização

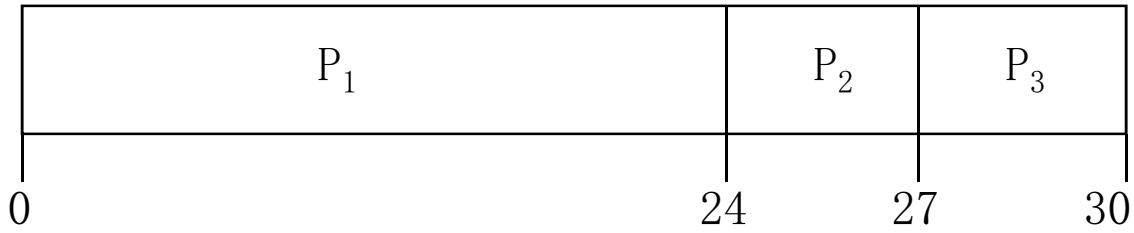
- Máxima utilização de CPU
- Máximo throughput
- Mínimo tempo de retorno
- Mínimo tempo de espera
- Mínimo tempo de resposta

Primeiro a chegar é servido (FCFS)

- Exemplo:

Processo	Duração de Surto
P_1	24
P_2	3
P_3	3

- Suponha que os processos chegam na ordem: P_1, P_2, P_3 O diagrama de Gantt para o escalonamento é:



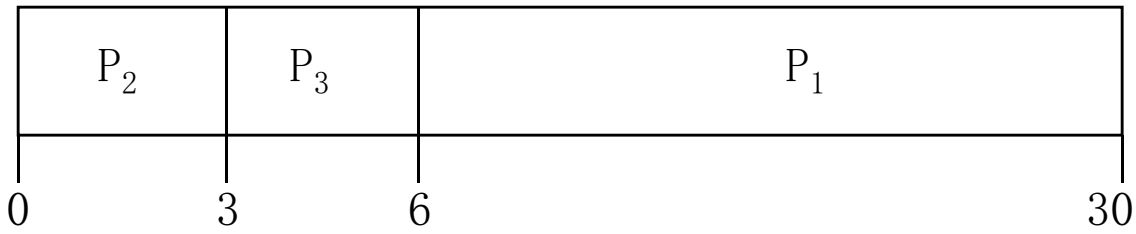
- O tempo de espera para P_1 = 0; P_2 = 24; P_3 = 27
- O tempo de espera médio: (0 + 24 + 27)/3 = 17

Primeiro a chegar é servido (FCFS)

Suponha que os processos cheguem na ordem

$$P_2\ ,\ P_3\ ,\ P_1\ .$$

- O diagrama de Gantt para o escalonamento é:



- O tempo de espera para $P_1 = 6$; $P_2 = 0$; $P_3 = 3$
- O tempo de espera médio : (6 + 0 + 3)/3 = 3
- Muito melhor que o anterior.
- *Efeito Comboio:* pequenos processos atrás de longos processos

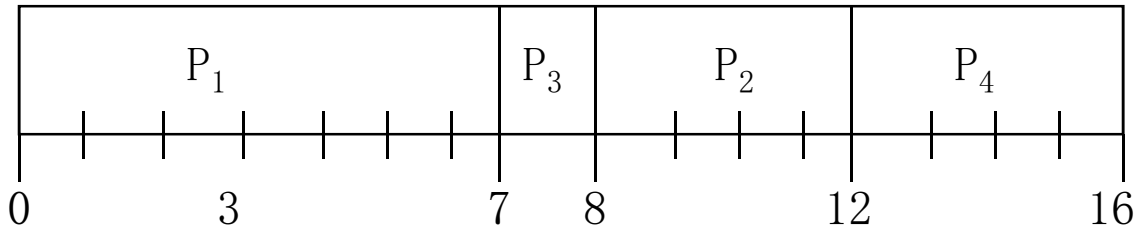
Escalonamento job mais curto primeiro (SJR)

- Associa a cada processos o tamanho do seu próximo surto de CPU. Usa estes valores para escalonar o processo com o menor tempo.
- Dois esquemas:
  - não-preemptivo - uma vez que a CPU é dada ao processo, não pode ser retirada até completar seu surto.
  - Preemptivo - se um novo processo chegar com um tempo de surto de CPU menor que o tempo restante do processo sendo executado, é feita a troca. Este esquema é conhecido como Menor-tempo-restante-primeiro (SRTF).
- SJF é ótimo - sempre dá o mínimo tempo médio de espera para o conjunto de processos.

Exemplo de SJF Não-preemptivo

Processo	Chegada	Tempo Surto
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (não-preemptivo)

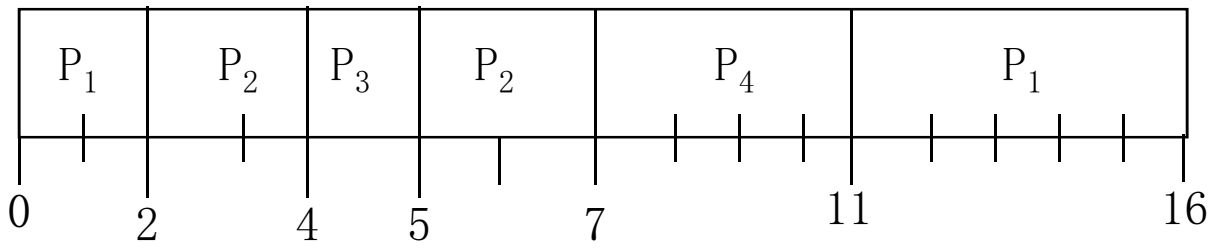


- Tempo médio de espera = (0 + 6 + 3 + 7)/4 = 4

Exemplo de SJF preemptivo

Processo	Chegada	Tempo de Surto
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

Escalonamento por prioridade

- Um número de prioridade (inteiro) é associado com cada processo
- A CPU é alocada para o processo com maior prioridade (menor valor ≡ maior prioridade).
  - Preemptivo
  - não-preemptivo
- SJF é um escalonamento por prioridade, onde a prioridade é dada pelo tempo de surto.
- Problema: Starvation - processos com baixa prioridade podem nunca serem executados.
- Solução ≡ Envelhecimento (Aging) - a prioridade aumenta com o tempo.

Round Robin (RR)

- Cada processo recebe uma pequena unidade de tempo de CPU (*time quantum*), geralmente 10-100 ms. Após este tempo, o processo é retirado e inserido no fim da fila de prontos.

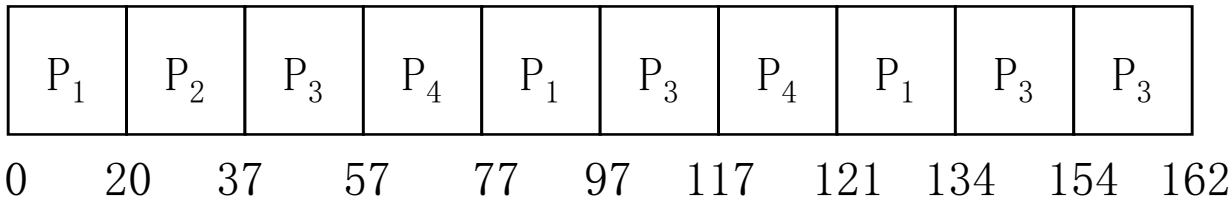
- Se existirem n processos na fila de prontos e o quantum for q, cada processo terá $1/n$ de tempo de CPU em parcelas de no máximo q unidades de tempo por vez. Nenhum processo esperará mais que $(n-1)q$ unidades de tempo.

- Performance
  - q grande $\Rightarrow$ FIFO
  - q pequeno $\Rightarrow$ q deve ser grande em relação ao tempo de troca de contexto, ou o overhead será muito alto.

Exemplo: RR com Quantum = 20

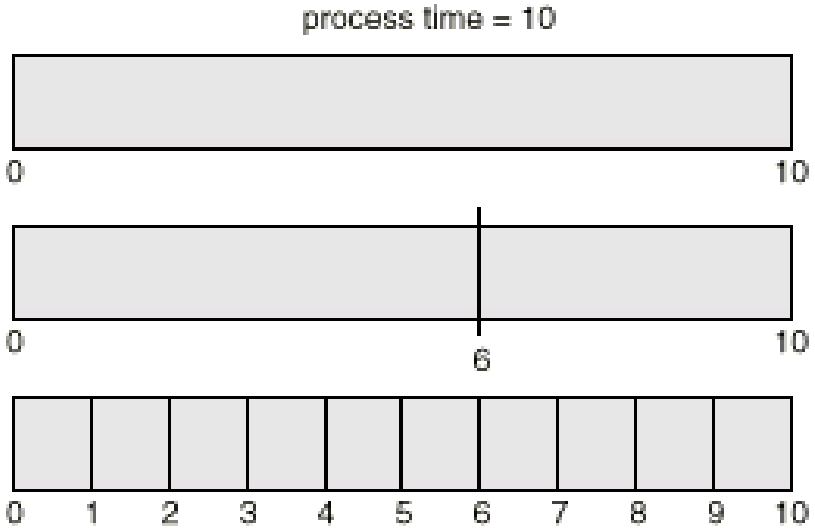
Processo	Tempo de Surto
P_1	53
P_2	17
P_3	68
P_4	24

- O diagrama de Gantt é:



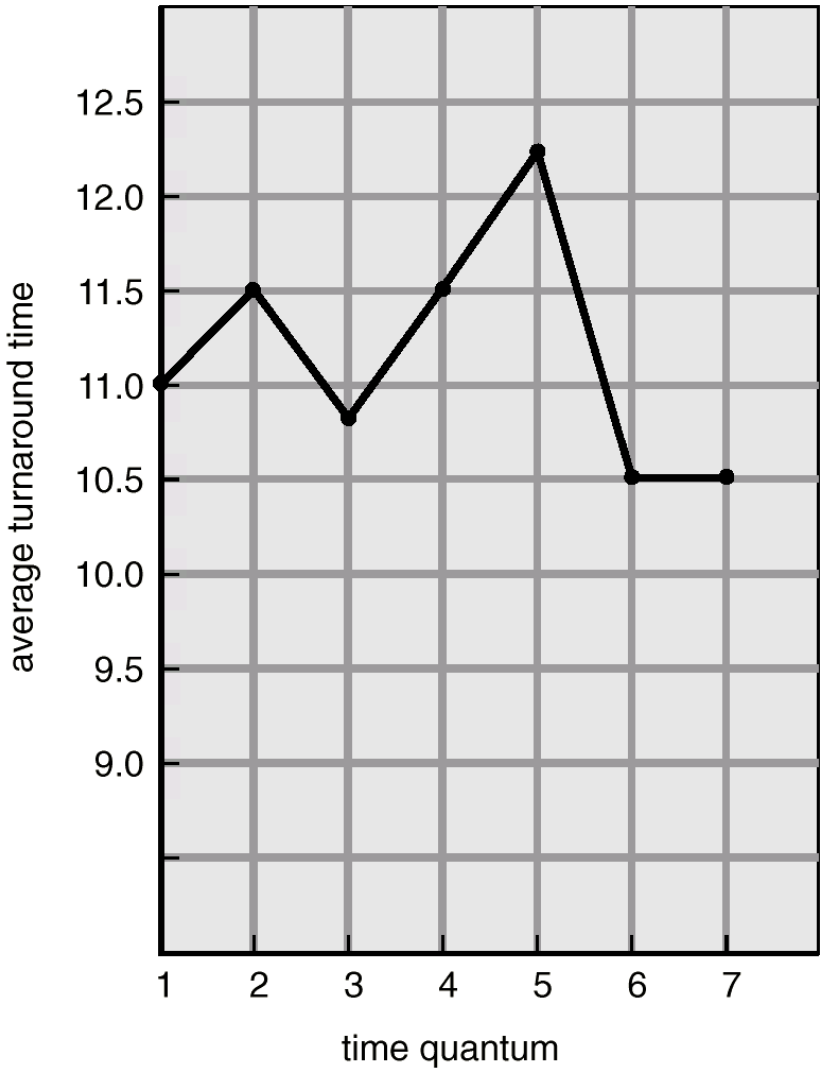
- Tipicamente, maior média de retorno que SJF, mas melhor *resposta.*

Como um Quantum de Tempo menor aumenta as trocas de contexto



quantum	context switches
12	0
6	1
1	9

Tempo de Retorno varia com o Tempo de Quantum

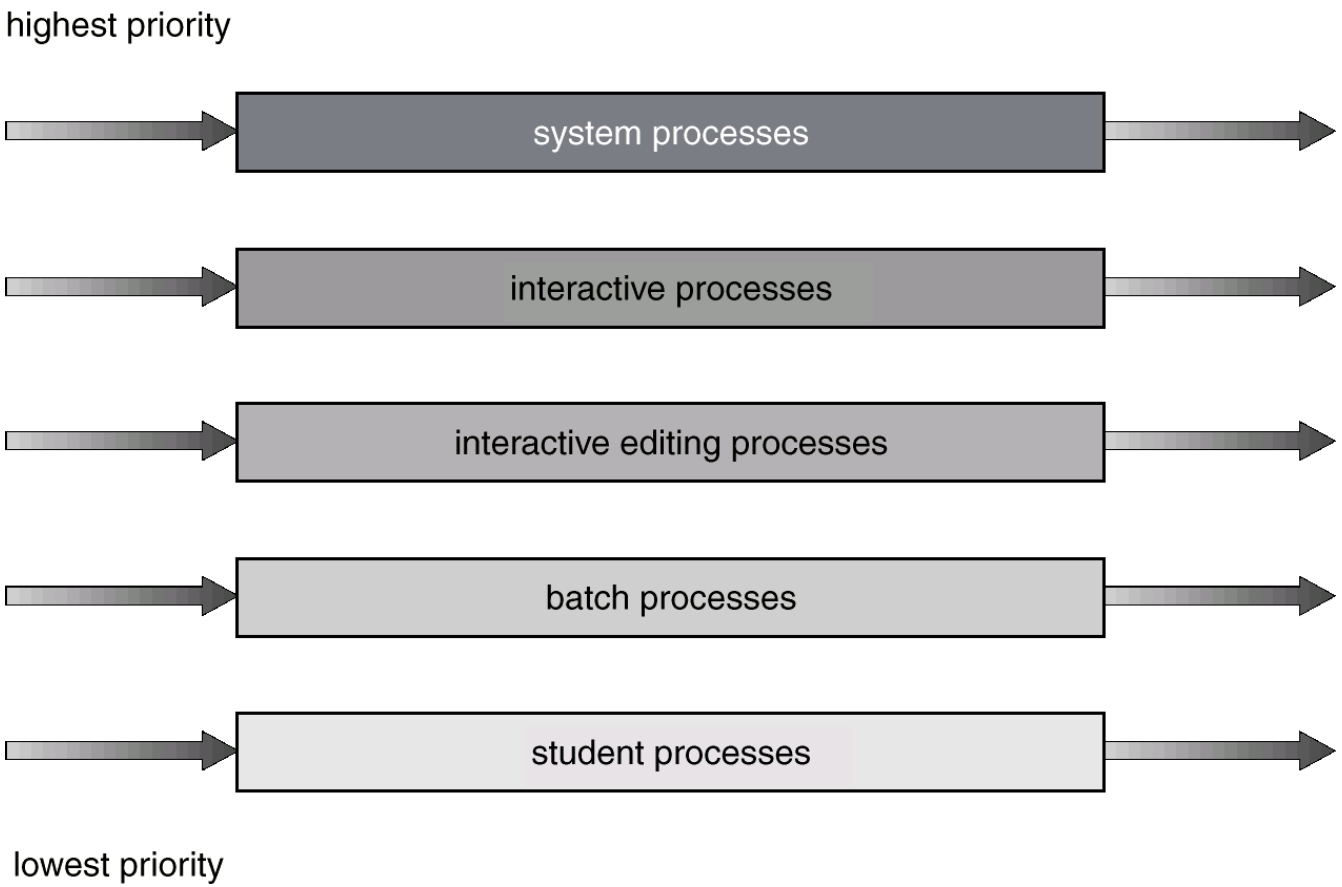


process	time
P_1	6
P_2	3
P_3	1
P_4	7

Filas Múltiplas

- A fila de prontos é particionada em filas separadas:
  primeiro plano (interativo)
  segundo plano (batch)
- Cada fila tem seu próprio algoritmo de escalonamento, por exemplo:
  - primeiro plano - RR
  - segundo plano - FCFS
- Deve haver escalonamento entre as filas.
  - Escalonamento de prioridade fixa: primeiro plano pode ter prioridade sobre o segundo plano. Possibilidade de starvation.
  - Fatia de tempo - cada fila recebe uma certa quantidade de tempo de CPU que pode ser escalonada entre seus processos. Por exemplo, 80% para primeiro plano em RR e 20% para o segundo plano em FCFS

Escalonamento em Filas Múltiplas

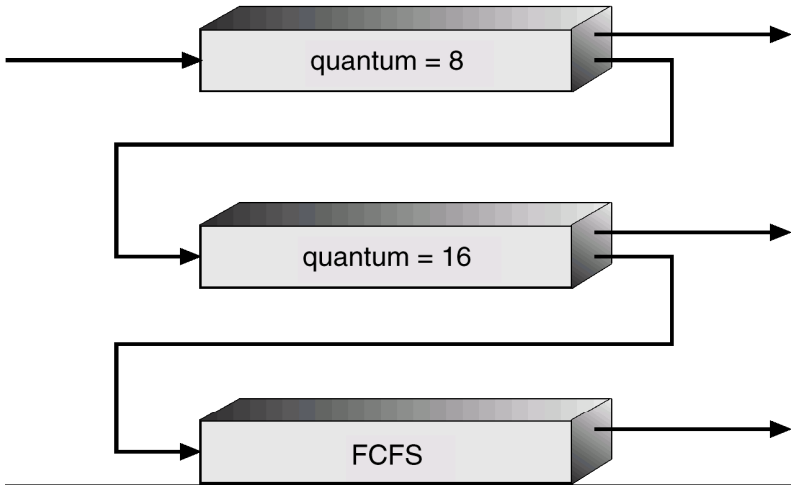


Filas Múltiplas com Realimentação

- Um processo pode passar de uma fila para outra. O envelhecimento pode ser implementado desta maneira.
- O escalonador de Filas Múltiplas com realimentação é definido pelos seguintes parâmetros:
  - número de filas
  - algoritmos de escalonamento para cada fila
  - método usado para determinar a promoção de um processo a uma fila de maior prioridade
  - método usado para determinar quando rebaixar um processo
  - método usado para determinar em qual fila o processo deve entrar quando precisar de serviço.

Exemplo de Filas Múltiplas com Realimentação

- Três filas:
  - Q_0 – quantum 8 ms
  - Q_1 – quantum 16 ms
  - Q_2 – FCFS



- Escalonador
  - Um novo job entra na fila Q_0 servido por FCFS. Quando recebe CPU, o job tem 8 ms. Se não terminar em 8 ms, o job é removido para Q_1.
  - Em Q_1 o job é servido por FCFS e recebe 16 ms adicionais. Se ainda não terminar, é transferido para Q_2.