Algoritmos de Escalonamento de CPU

Aluno: Victor Souza Santos
R.A.: 00000844512
Professor: Marcos Canejo
Curso: Ciências da Computação

1. Primeiro-a-Chegar, Primeiro-a-Ser-Atendido (FCFS)

- **Descrição Geral**

O algoritmo FCFS é o mais simples entre os algoritmos de escalonamento. Ele funciona de maneira bastante intuitiva, processando os pedidos na ordem em que chegam. Esse método é similar à política de atendimento em uma fila de banco ou supermercado.

- **Funcionamento Detalhado**

1. *Fila de Prontos*: Os processos são mantidos em uma fila na ordem de chegada;

2. *Execução Sequencial*: Quando a CPU está livre, o processo na frente da fila é executado até sua conclusão. Durante essa execução, a CPU não troca para outro processo;

3. *Tempo de Espera*: Cada processo deve esperar que todos os processos que chegaram antes dele sejam completados;

4. *Não Preemptivo*: Uma vez que um processo começa a executar, ele continua até completar. Não há interrupções a não ser por eventos como I/O.

- **Exemplificação**

Considere três processos com tempos de chegada e durações diferentes:

Processo	Tempo de Chegada	Duração
P1	0	4
P2	1	3
P3	2	1

No FCFS, a ordem de execução será P1, P2, P3. Apesar de P3 ser mais curto, ele deve esperar P1 e P2 serem completados.

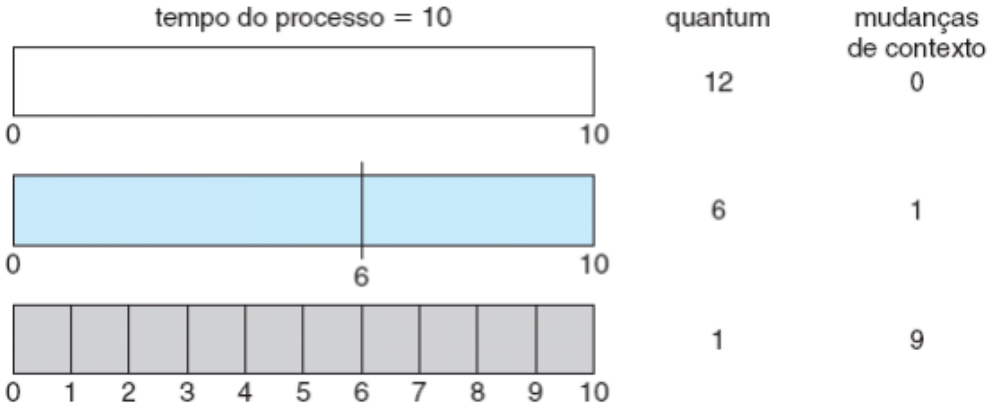


Figura 1: Com uma quantidade de tempo MENOR, AUMENTA as mudanças do contexto.

- **Vantagens**

- Simples: Fácil de entender e implementar.
- Justo: Atende aos processos na ordem de chegada, sem priorização.

- **Desvantagens**

- Tempo Médio de Espera: Pode ser elevado se processos longos chegarem antes dos curtos.
- Problema do Comboio: Processos curtos podem ficar presos atrás de processos longos, levando a altos tempos de resposta.

2. Menor-Job-Primeiro (SJF)

- **Descrição Geral**

O algoritmo SJF seleciona o processo com o menor tempo de execução para ser o próximo a executar. Este método é mais eficiente em termos de tempo de resposta médio do que o FCFS.

- **Funcionamento Detalhado**

1. *Estimativa do Tempo de Execução*: Cada processo tem um tempo de execução estimado;

2. *Ordenação por Duração*: Os processos são ordenados pela duração estimada;

3. *Execução do Menor Primeiro*: O processo com a menor duração é selecionado para execução;

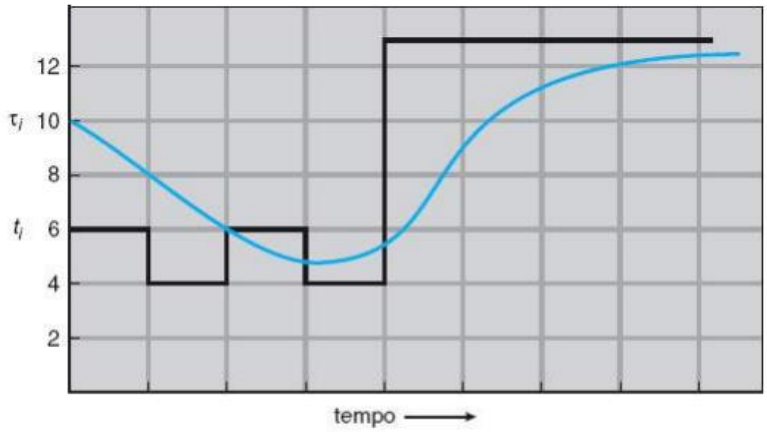
4. *Variações*: SJF pode ser preemptivo (Shortest Remaining Time First - SRTF) ou não preemptivo. No modo preemptivo, se um novo processo com duração menor chega, ele pode interromper o processo atual.

- **Exemplificação**

Considere quatro processos:

Processo	Tempo de Chegada	Duração
P1	0	6
P2	1	8
P3	2	7
P4	3	3

A ordem de execução será P1, P4, P3, P2 para SJF não preemptivo. Para SJF preemptivo, pode haver interrupções se processos com durações menores chegarem.



pico de CPU (t_i)	6	4	6	4	13	13	13	...
"convidado" (τ_i)	10	8	6	6	5	9	11	12	...

Figura 2: Previsão da duração do próximo pico de CPU

- **Vantagens**

- Eficiência: Minimiza o tempo médio de espera.
- Bom Desempenho: Processos curtos são atendidos rapidamente.

- **Desvantagens**

- Dificuldade na Estimativa: É difícil prever o tempo de execução de um processo.
- Inanição: Processos longos podem sofrer inanição se houver uma chegada constante de processos curtos.

3. Escalonamento por Prioridades

- **Descrição Geral**

O escalonamento por prioridades aloca a CPU aos processos com base em suas prioridades. Cada processo é associado a uma prioridade, e a CPU é alocada ao processo com a maior prioridade.

- **Funcionamento Detalhado**

1. *Atribuição de Prioridades*: Cada processo recebe uma prioridade, que pode ser estática (fixa) ou dinâmica (alterada durante a execução);

2. *Ordenação por Prioridade*: Os processos são ordenados com base na prioridade;

3. *Execução*: O processo com a maior prioridade é selecionado para execução;

4. *Preempção*: Pode ser preemptivo ou não preemptivo. No modo preemptivo, um processo de alta prioridade pode interromper um processo de menor prioridade.

- **Exemplificação**

Considere quatro processos com diferentes prioridades (menor número indica maior prioridade):

| Processo | Tempo de Chegada | Prioridade | Duração |

P1	0	2	4
P2	1	1	3
P3	2	4	1
P4	3	3	2

A ordem de execução será P2, P1, P4, P3 no caso de prioridades estáticas.

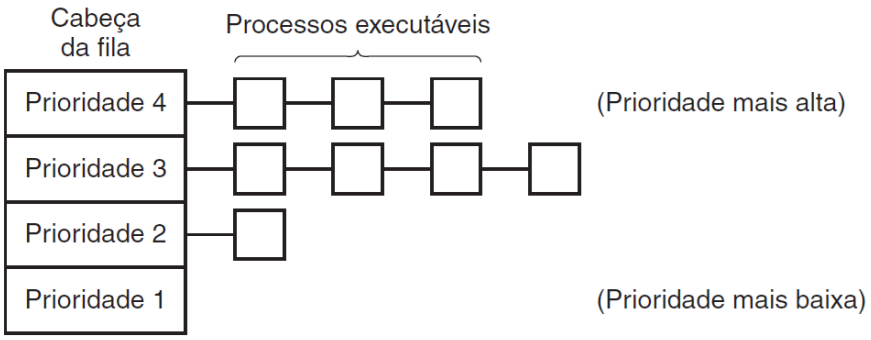


Figura 3: Divisória de como se executam os processos por sua prioridade.

- **Vantagens**

- Flexível: Permite a implementação de políticas específicas.
- Eficiência: Processos críticos podem ser atendidos rapidamente.

- **Desvantagens**

- Inanição: Processos com baixa prioridade podem não ser atendidos.
- Complexidade: Gestão de prioridades pode ser complexa.

4. Round-Robin (RR)

- **Descrição Geral**

O Round-Robin é um algoritmo de escalonamento amplamente utilizado em sistemas de tempo compartilhado. Cada processo recebe um quantum de tempo durante o qual pode executar. Após o término do quantum, o processo é colocado no final da fila.

- **Funcionamento Detalhado**

1. *Fila Circular*: Os processos são organizados em uma fila circular;

2. *Quantum*: Cada processo é alocado um tempo fixo (quantum) para execução;

3. *Troca de Contexto*: Após o término do quantum, o processo é movido para o final da fila e o próximo processo é executado;
4. *Repetição*: Este ciclo continua até que todos os processos sejam concluídos.

- **Exemplificação**

Considere quatro processos com tempos de chegada e durações diferentes e um quantum de 2 unidades de tempo:

Processo	Tempo de Chegada	Duração
P1	0	5
P2	1	4
P3	2	2
P4	3	1

A ordem de execução com um quantum de 2 será P1, P2, P3, P4, P1, P2, P1.

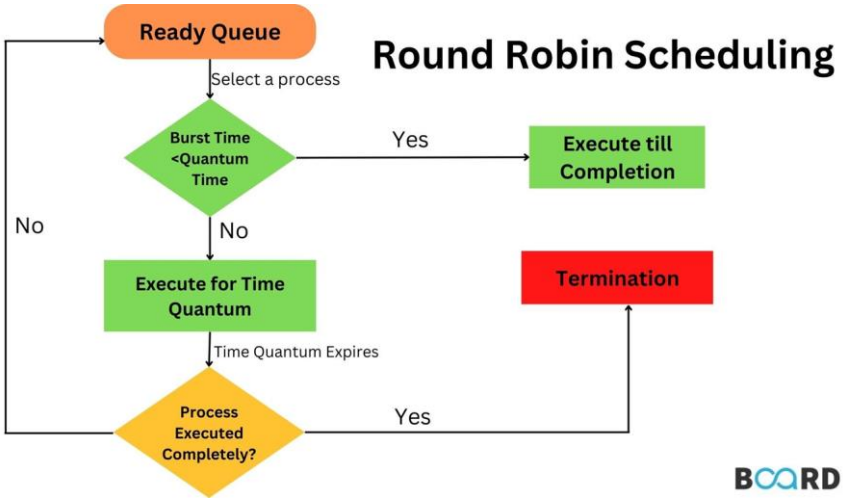


Figura 4: Fluxograma do algoritmo Round Robin (RR).

- **Vantagens**

- Justo: Todos os processos recebem tempo igual de CPU.
- Responsividade: Adequado para sistemas interativos, onde processos recebem atenção regular.

- **Desvantagens**

- Desempenho Dependente do Quantum: Se o quantum é muito pequeno, a troca de contexto frequente pode causar sobrecarga; se for muito grande, pode se assemelhar ao FCFS.
- Complexidade de Gestão de Quantum: Definir o tamanho adequado do quantum pode ser desafiador.

5. Conclusão

Os algoritmos de escalonamento de CPU têm diferentes características e adequações. O FCFS é simples mas pode causar longos tempos de espera. O SJF é eficiente para tempos de resposta, mas difícil de implementar devido à necessidade de estimativas precisas. O escalonamento por prioridades é flexível, mas pode causar inanição de processos. O Round-Robin é justo e adequado para sistemas interativos, mas o desempenho depende da escolha adequada do quantum. A escolha do algoritmo deve ser feita com base nas necessidades específicas do sistema e na natureza das cargas de trabalho esperadas.

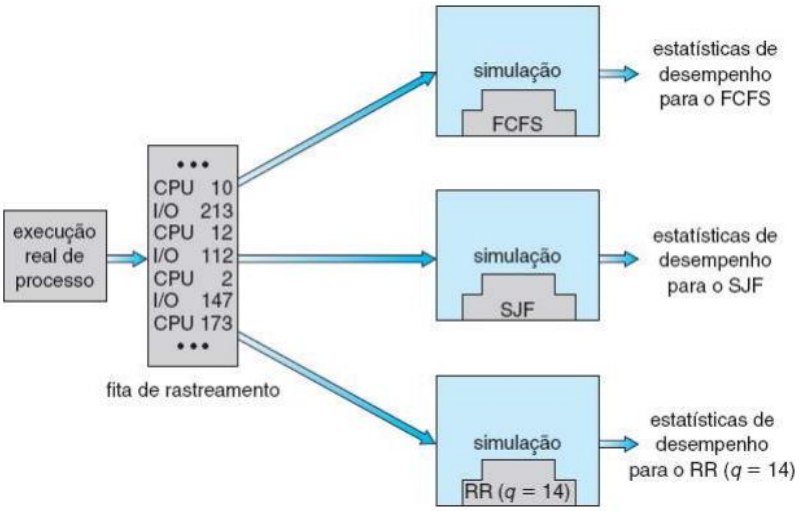


Figura 5: Comportamento de schedulers da CPU por simulação em algoritmos diferentes.

- **Referências**

- Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). Operating System Concepts. John Wiley & Sons.
- Stallings, W. (2014). Operating Systems: Internals and Design Principles. Pearson Education.