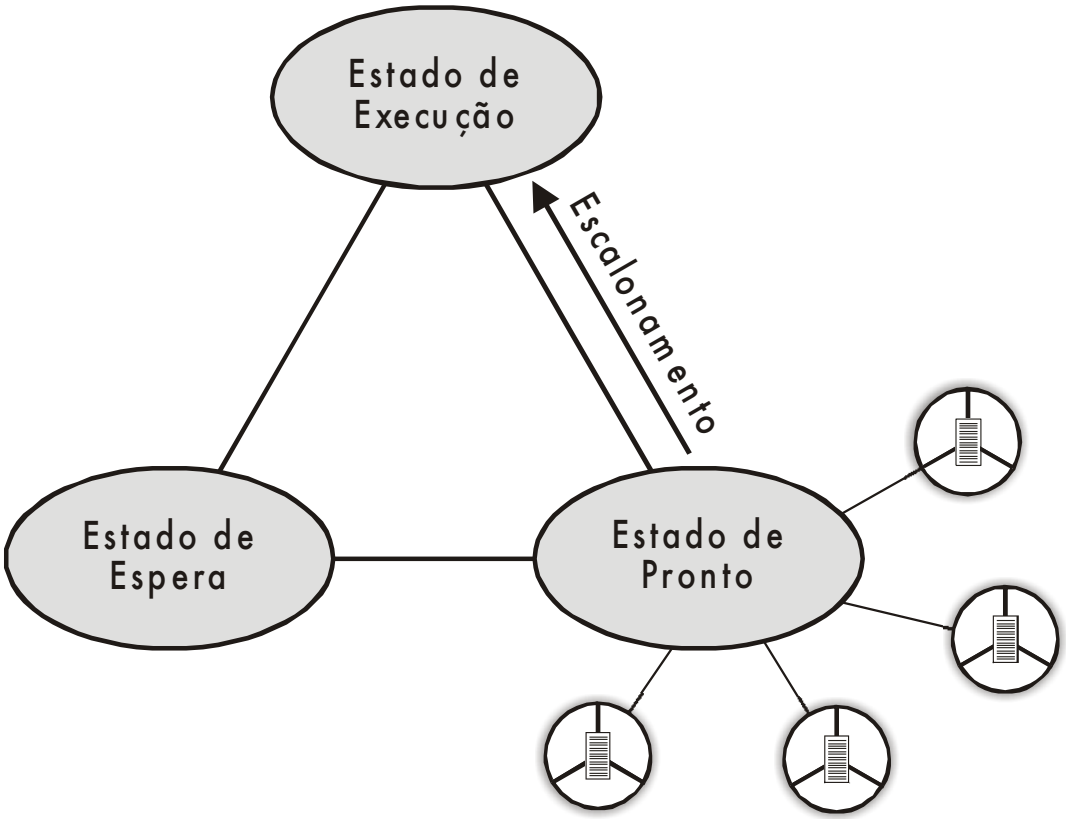
Escalonamento de Processos

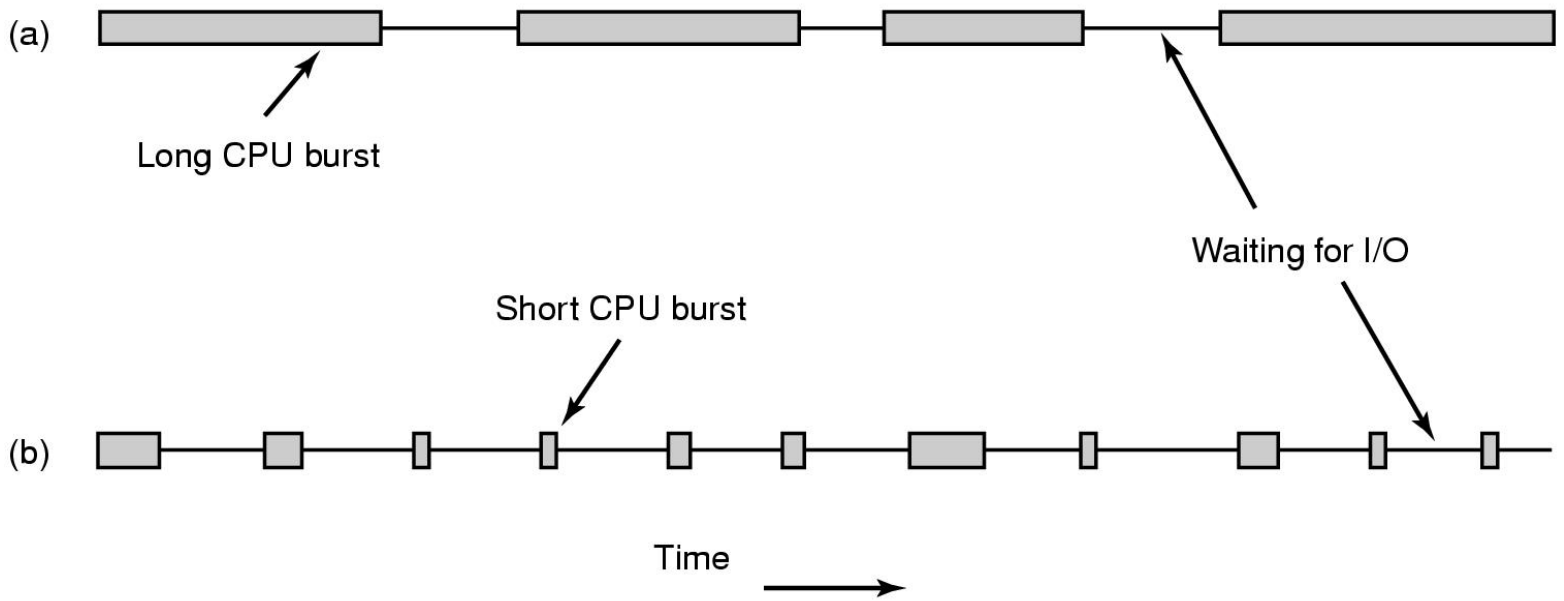
ESCALONAMENTO DE PROCESSOS

Quando mais de um processo está ativo, o S.O. deve escolher qual deve executar em primeiro lugar

Uso do Escalador (e de um algoritmo de escalonamento)



Escalonamento
Comportamento do processo



- Surtos de uso da UCP se alternam com períodos de espera por E/S
  - (a) Um processo orientado à UCP (*CPU-bound*)
  - (b) Um processo orientado à E/S (*I/O bound*)

Objetivos do Algoritmo de Escalonamento

Todos os sistemas

- **Justiça** – dar a cada processo uma porção justa de UCP
- **Aplicação da política** – verificar se a política estabelecida é cumprida
- **Equilíbrio** – manter ocupadas todas as partes do sistema

Objetivos do Algoritmo de Escalonamento (cont.)

Sistemas em Lote

- **Vazão (*throughput*)** – maximizar o número de jobs por hora
- **Tempo de retorno (*turnaround*)–** minimizar o tempo entre a submissão e o retorno
- **Utilização da UCP** – manter a UCP ocupada o tempo todo

Sistemas Interativos

- **Tempo de Resposta** – responder rapidamente às requisições
- **Proporcionalidade** – satisfazer às expectativas dos usuários

Sistemas de Tempo Real

- **Cumprimento dos prazos** – evitar perda de dados
- **Previsibilidade –** evitar a degradação da qualidade em sistemas multimídia

Processos

Escalonamento de Processos

- **Problema: critérios são contraditórios**
- **Cada processo é único e imprevisível (diferenças na quantidade de cálculos e E/S)**
- **Estratégias para o escalonamento:**
  - **Escalonamento Preemptivo**
  - **Escalonamento Não-Preemptivo**

Escalonamento em Sistemas em Lote

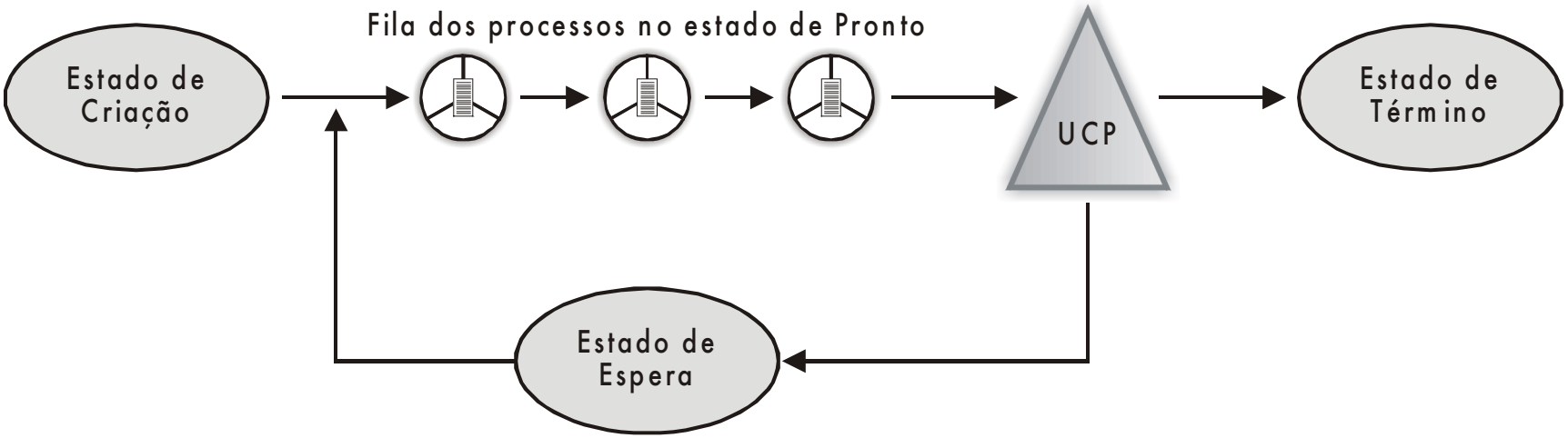
❑ Primeiro a Chegar, Primeiro a ser Servido

- **Conhecido também como *First come, first served* (FCFS), é não preemptivo**
- **O processo que chegar primeiro ao estado pronto é o selecionado para execução**
- **Algoritmo simples: só necessita uma fila, onde os processos que entram no estado pronto entram no seu final e são escalados quando chegam a seu início**
- **Quando o processo em execução termina seu processamento ou vai para o estado de espera, o primeiro processo da fila é escalonado**
- **Quando saem do estado de espera, todos os processos entram no final da fila dos prontos**

Escalonamento em Sistemas em Lote

❑ Primeiro a Chegar, Primeiro a ser Servido

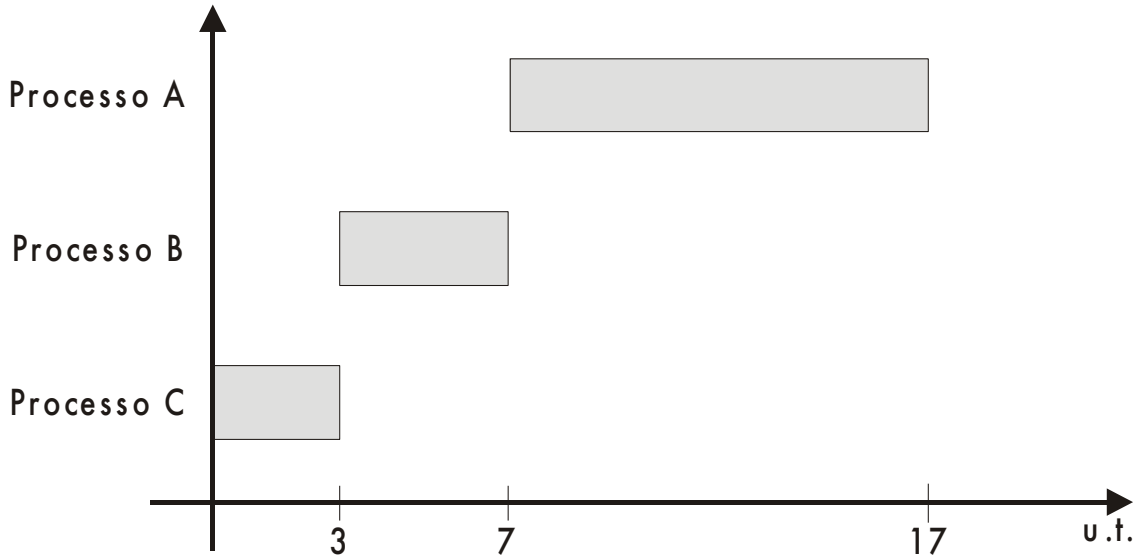
- É um algoritmo justo e fácil de implementar
- Desvantagens:
  - Imprevisibilidade do início do processo
  - a intercalação de processos orientados à computação e orientados à E/S podem trazer ineficiência



Escalonamento em Sistemas em Lote

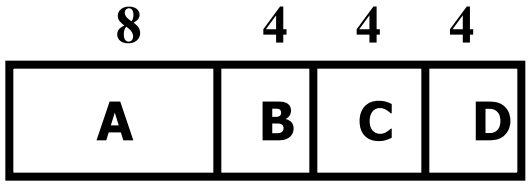
❑ JOB MAIS CURTO PRIMEIRO

- **Algoritmo mais apropriado para sistemas que executam jobs em batch (o tempo de execução do job é conhecido com antecedência)**
- **É não preemptivo**
- **Consiste em alocar processador aos menores jobs primeiro (de uma fila de jobs com mesma importância)**

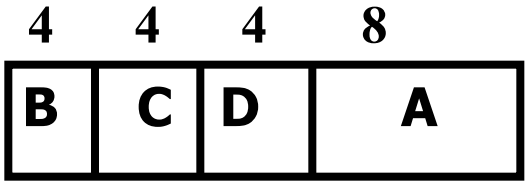


❑ JOB MAIS CURTO PRIMEIRO

Exemplo:



Turnaround médio =
(8+12+16+20)/4 = 14 por job



Turnaround médio =
(4+8+12+20)/4 = 11 por job

Considerando o caso de 4 jobs, com tempos de processamento iguais a: "a", "b", "c" e "d": o tempo médio de processamento de cada job é dado por

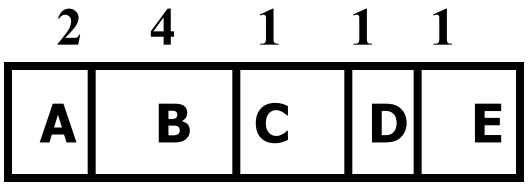
$$(4a + 3b + 2c + d)/4$$

Portanto "a" contribui mais para a média do que os demais.

❑ JOB MAIS CURTO PRIMEIRO

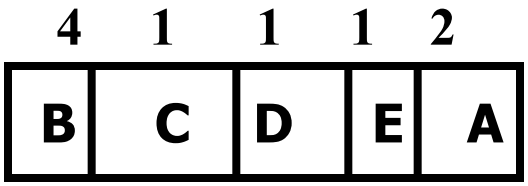
- **O algoritmo do menor job só conduz a resultados ótimos se todos os jobs estão <u>disponíveis ao mesmo tempo</u>.**
- **Por exemplo, considerando 5 jobs, de A a E, com tempos de execução de 2, 4, 1, 1 e 1, respectivamente. Seus tempos de chegada são: 0, 0, 3, 3 e 3.**

Aplicação parcial do algoritmo



Tempo médio de espera para início
(0+2+6+7+8)/5 =4,6 por job

Outra ordem possível



Tempo médio de espera para início
(0+4+5+6+7)/5 =4,4 por job

Escalonamento em Sistemas em Lote

❑ Próximo de Menor tempo restante

- Versão preemptiva do Job mais curto primeiro
- O escalonador sempre escolhe o processo cujo tempo de execução restante seja o menor
- O tempo de execução deve ser previamente estabelecido
- Quando chega um novo job, seu tempo total é comparado com o tempo restante do processo em curso
- Se, para terminar, o novo job precisar de menos tempo que o processo atual, então este será suspenso e o novo job será iniciado
- Este esquema permite que um novo job tenha um bom desempenho

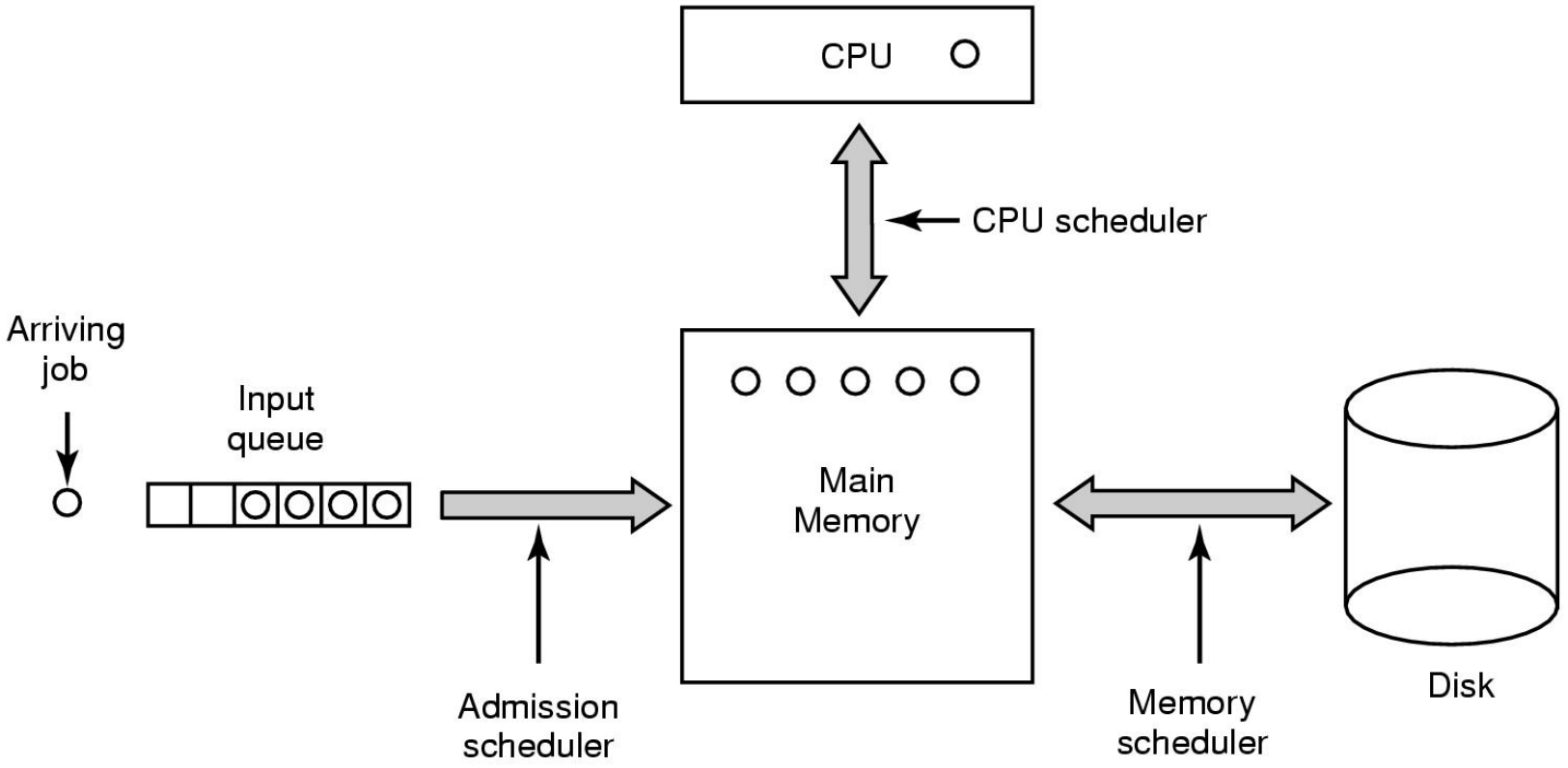
Escalonamento em Sistemas em Lote

❑ Escalonamento em três níveis

- **Existem 3 níveis de escalonadores:**
  - **Nível 1: Escalonador de admissão – decide qual job será admitido no sistema, outros serão mantidos em uma fila**
  - **Nível 2: Escalonador de Memória – decide quais processos ficarão na memória e quais permanecerão no disco**
  - **Nível 3: Escalonador de UCP – escolhe qual dos processos prontos na memória usará a UCP.**

  Qualquer algoritmo (preemptivo ou não) pode ser usado

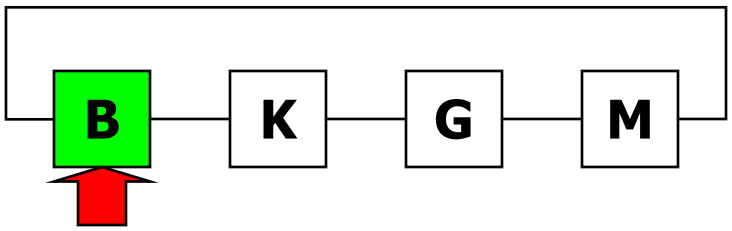
❑ Escalonamento em três níveis



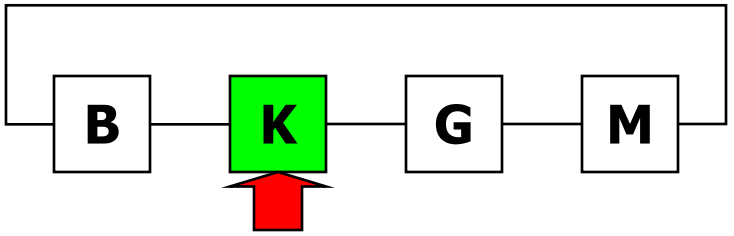
Escalonamento em **Sistemas** Interativos

❑ ESCALONAMENTO ROUND ROBIN

- **Algoritmo antigo, porém justo, simples e muito usado**
- **A cada processo é atribuído um intervalo de tempo idêntico (quantum), ao final do qual é retirado de execução**
- **É de fácil implementação: o escalonador mantém uma lista de processos prontos para executar. Quando o quantum de um processo se esgota, o mesmo é colocado no final da fila.**

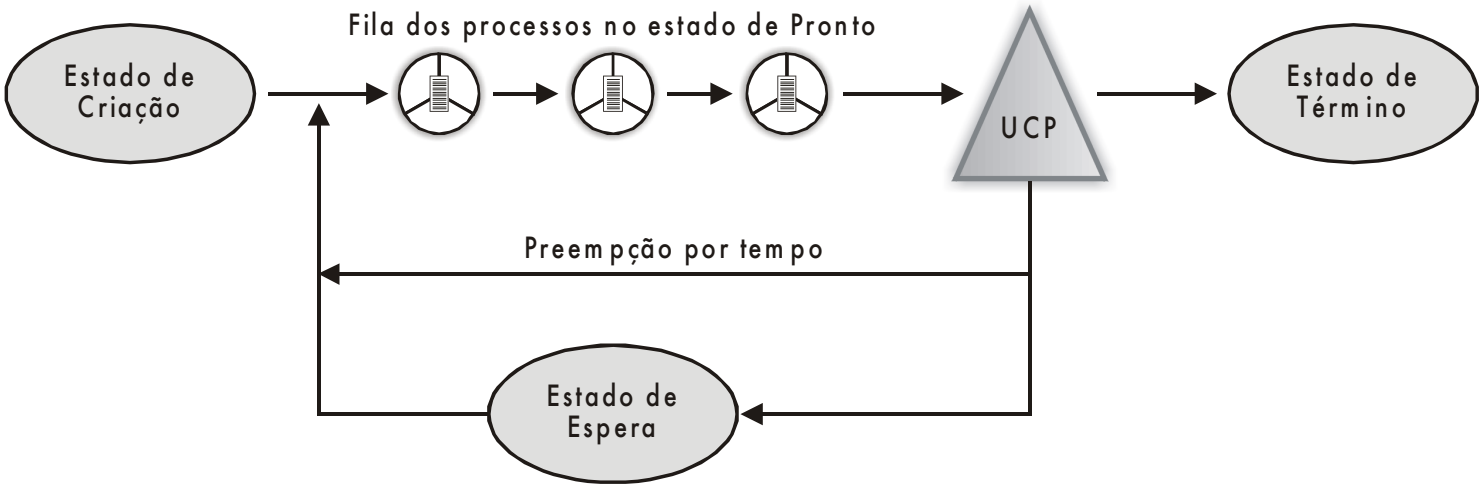


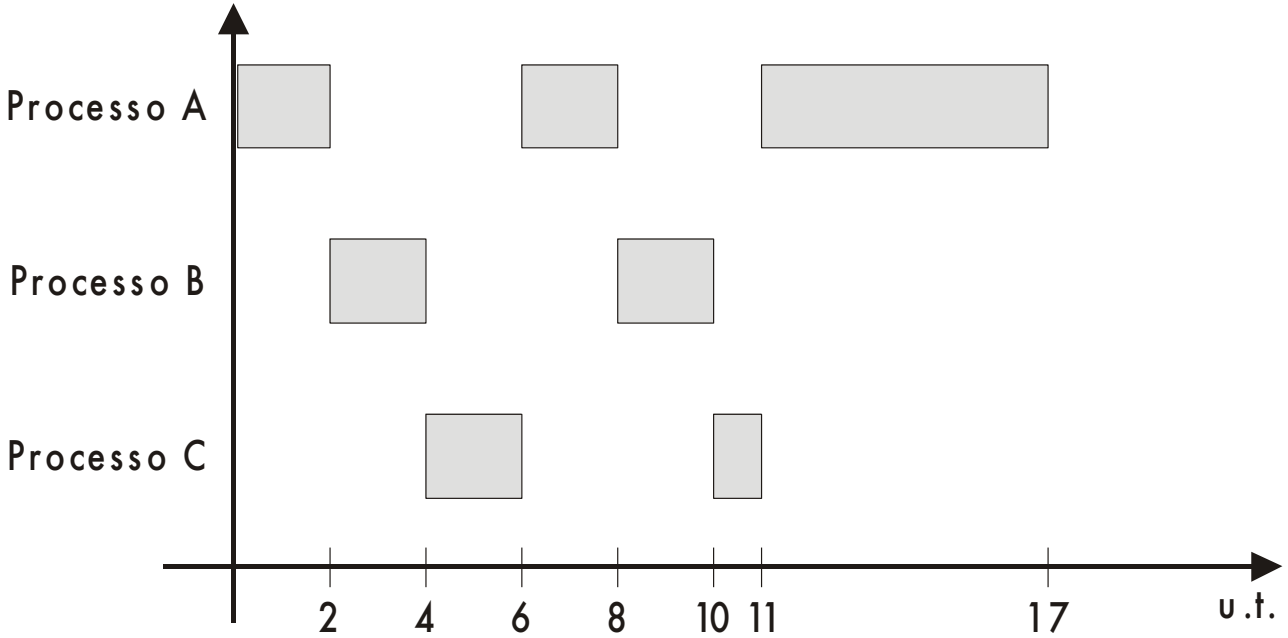
Processo corrente



Processo corrente

❑ ESCALONAMENTO ROUND ROBIN





❑ ESCALONAMENTO ROUND ROBIN

- **Problema do Round Robin: determinação do Quantum**
  - **Quantum muito pequeno: sucessivas trocas de contexto e redução na eficiência do processador**
  - **Quantum muito grande: tempo de resposta não aceitável em sistemas interativos**

Escalonamento em Sistemas Interativos

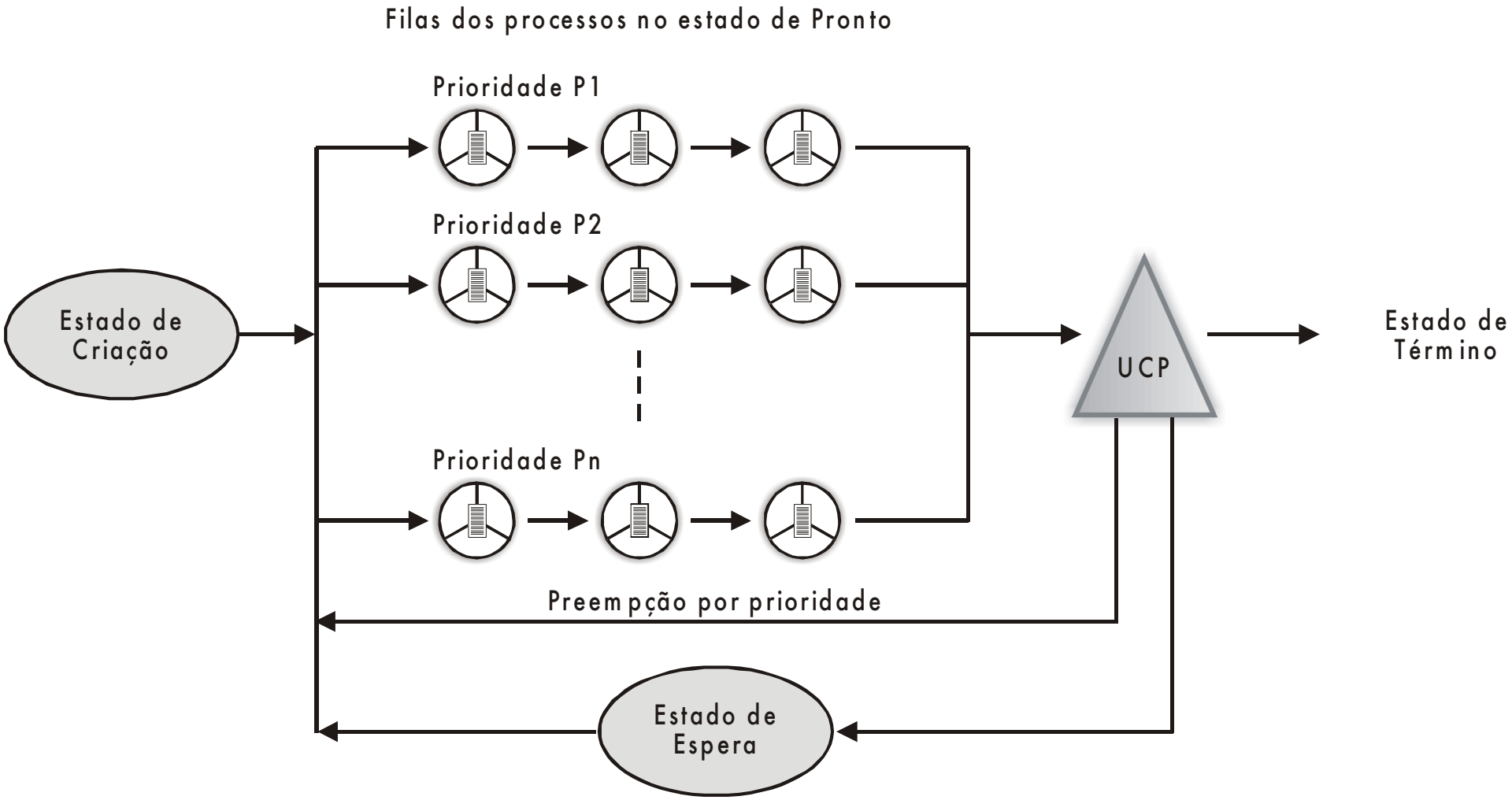
❑ ESCALONAMENTO POR PRIORIDADE

- **Nem sempre todos os processos têm a mesma importância: existência de prioridade**

- **Idéia do algoritmo é simples: a cada processo é associada uma prioridade, e o processo pronto com a maior prioridade será executado primeiro**

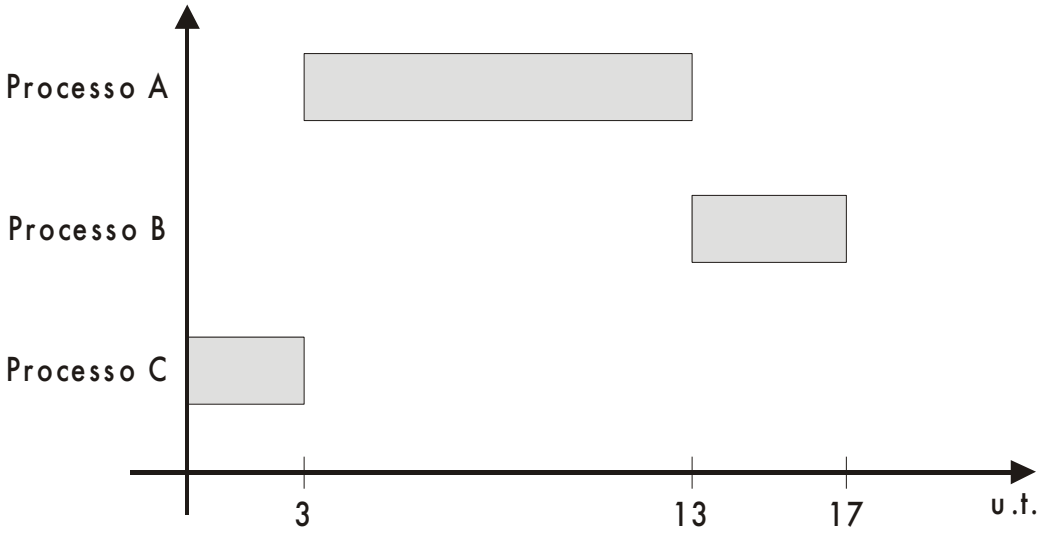
Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO POR PRIORIDADE



Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO POR PRIORIDADE



Processo	Tempo de processador (u.t.)	Prioridade
A	10	2
B	4	1
C	3	3

Escalonamento em **Sistemas** Interativos

❑ ESCALONAMENTO POR PRIORIDADE

As Prioridades podem ser:

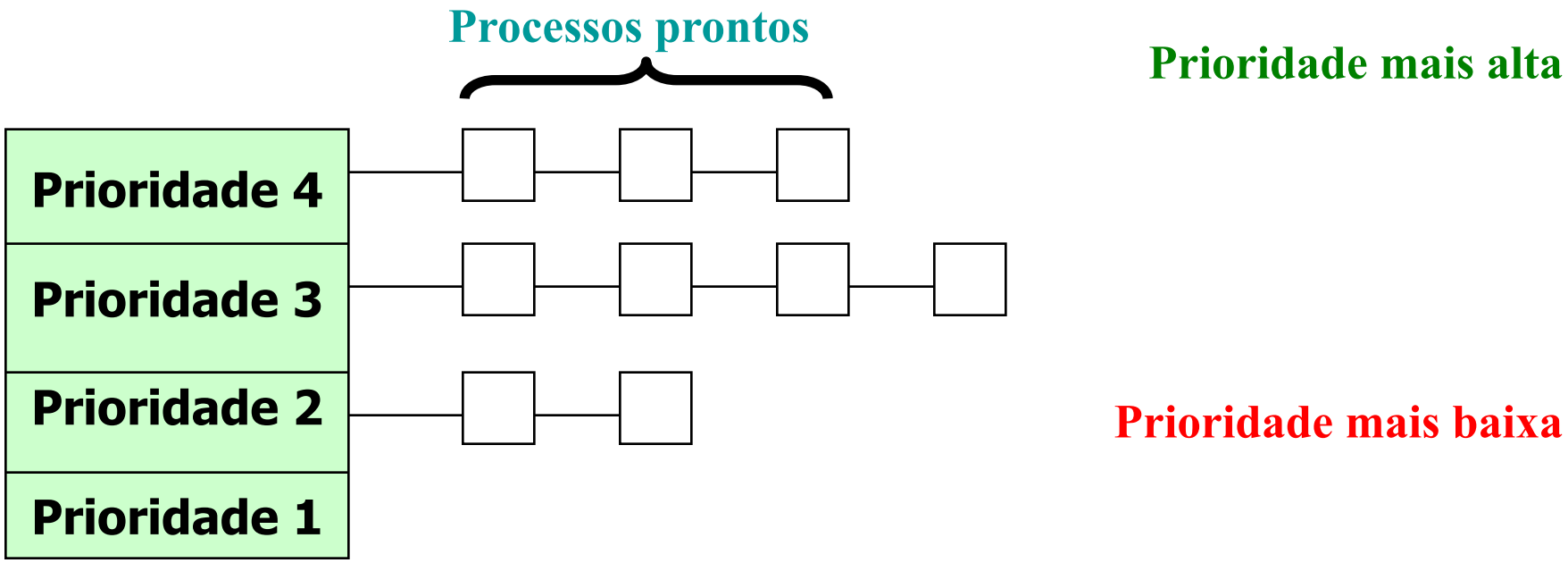
- **Prioridades Estáticas: podem representar uma hierarquia fixa (ex. área militar, acadêmica, etc.)**

- **Prioridades Dinâmicas: por exemplo, para contemplar os processos com maior taxa de E/S.**
  - **Algoritmo simples: associar uma prioridade igual a `1/f` , onde f é a fração do quantum que o processo usou no último escalonamento.**
    Ex: usou 50 de um quantum de 100, prioridade = 1/50/100 = 2

Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO POR PRIORIDADE

- Para evitar monopólio do processador pelos processos com maior prioridade: sistema decrementa a prioridade a cada quantum.
- Pode ser conveniente, em algumas vezes, agrupar os processos em classes de prioridades, e usar o **escalonamento com prioridades entre as classes** e o **round robin dentro da mesma classe.**



Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO POR MÚLTIPLAS FILAS

- **Um dos mais antigos escalonadores com prioridade foi projetado para o sistema CTSS**
- **Problema com o CTSS: só mantinha um processo em sua pequena memória principal**
- **Solução dos projetistas: quantum longo aos processos que usavam muita UCP como forma de reduzir o swap**
- **Para evitar o tempo de resposta ruim desta solução, dividiram os processos em classes de prioridade: processos com maior prioridade, rodavam 1 quantum; os da classe seguinte, 2 quanta; o da próxima, 4 quanta, e assim por diante.**

Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO POR MÚLTIPLAS FILAS

Exemplo: considerando que um processo precisa rodar 100 quanta

1a. Vez que rodar: receberá 1 quantum
2a. Vez que rodar: receberá 2 quanta
3a. Vez que rodar: receberá 4 quanta
4a. Vez que rodar: receberá 8 quanta
5a. Vez que rodar: receberá 16 quanta
6a. Vez que rodar: receberá 32 quanta
7a. Vez que rodar: receberá 64 quanta
(dos quais precisará apenas 37 quanta)

} **Apenas 7 Swaps**

Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO POR MÚLTIPLAS FILAS

- **Outro algoritmo que atribui classes de prioridades aos processos: o utilizado no sistema XDS 940**
- **Usava quatro classes de prioridades:**

Terminal
Entrada/Saída
Quantum curto
Quantum longo

Maior Prioridade

Menor Prioridade

Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO PRÓXIMO PROCESSO MAIS CURTO (*Shortest Process Next*)

A política do JOB MAIS CURTO PRIMEIRO pode ser usada em sistemas interativos, na tentativa de se obter um melhor tempo de resposta.

Cada comando interativo corresponderá a um job.

Cada comando terá sua duração estimada a partir do seu comportamento passado.

Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO PRÓXIMO PROCESSO MAIS CURTO (*Shortest Process Next*)

- Suponha que o tempo estimado para comandos em um Terminal seja T_0. Agora na rodada seguinte, o tempo medido seja T_1. Podemos atualizar nossa estimativa considerando:

$$aT_0 + (1 - a)T_1$$

- Uma escolha fácil de implementar, é fazer a = 1/2

- Esta técnica de estimativa é chamada de "aging".

Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO PRÓXIMO PROCESSO MAIS CURTO (*Shortest Process Next*)

- **Após algumas estimativas sucessivas, teremos:**

$$T_0$$

$$T_0/2 + T_1/2$$

$$T_0/4 + T_1/4 + T_2/2$$

$$T_0/8 + T_1/8 + T_2/4 + T_3/2$$

- **Nota-se que o peso da estimativa T_0 caiu para 1/8.**

Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO GARANTIDO

- **Consiste em fazer promessas ao usuário a respeito do desempenho e cumprí-las de alguma forma.**
- **Uma promessa fácil de fazer e de cumprir é a seguinte: se houver n usuários ativos, cada um irá receber cerca de 1/n da capacidade do processador.**
- **Para manter esta promessa, o S.O. deve manter o controle da quantidade de UCP que cada processo recebe, desde sua criação**

Escalonamento em Sistemas Interativos

ESCALONAMENTO POR LOTERIA

- Idéia básica: dar bilhetes de loteria aos processos, cujos prêmios são recursos do sistema, como tempo de UCP
- Se houver uma decisão de escalonamento, um bilhete de loteria será escolhido aleatoriamente e o processo que tem o bilhete conseguirá o recurso
- O sistema pode fazer um sorteio 50 vezes por segundo, cada vencedor terá 20 ns de UCP como prêmio
- Quantos mais bilhetes um processo tiver, maior será sua chance de ser sorteado

Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO POR FRAÇÃO JUSTA

- Até agora, não foi considerado quem o dono do processo a ser escalado, o qual é escalado independentemente
- Portando, se o usuário1 iniciar 9 processos e, o usuário 2, iniciar 1 processo, o usuário 1 obterá 90% do tempo de UCP
- Para evitar isso, alguns sistemas consideram a propriedade do processo, antes de escalá-lo
- Nesse modelo, cada a cada usuário é alocado uma fração de tempo de UCP, e o escalonador escolhe os processos de modo a garantir esta fração

Escalonamento em Sistemas Interativos

❑ ESCALONAMENTO POR FRAÇÃO JUSTA

- **Por exemplo:**
  - **existem 2 usuários, e a cada um é prometido 50% do tempo de UCP**
  - **se o usuário1 iniciar 4 processos (A,B,C,D) e, o usuário 2, iniciar 1 processo (E) e, se for usado o algoritmo Round Robin, uma sequência possível será:**

    AEBECEDEAEBECEDE...

  - **se ao usuário1 for destinado o dobro de tempo de UCP, do que para o usuário2, uma sequência possível será:**

    ABECDEABECDE...

Escalonamento em Sistemas de Tempo Real

- Sistemas em que o tempo tem uma função especial
- Existem 2 categorias de sistemas de tempo real:
  - **Tempo real crítico:** há prazos absolutos que devem ser cumpridos
  - **Tempo real não crítico:** o descumprimento ocasional de um prazo é indesejável, porém, tolerável

- Os processos têm, geralmente, vida curta
- Quando detectado um evento externo, o trabalho do escalonador é escalonar processos de maneira a cumprir os prazos

Escalonamento em Sistemas de Tempo Real

- **Os eventos que o sistema pode precisar responder são classificados como:**
  - **Periódicos: ocorrem em intervalos regulares**
  - **Aperiódicos: acontecem de modo imprevisível**
- **Um sistema é escalonável se, dado:**
  - ***m* eventos periódicos**
  - **O evento i ocorre dentro de um período Pi e requer Ci segundos**
- **Então a carga pode ser manipulada se**

$$\sum_{i=1}^{m} \frac{C_i}{P_i} \leq 1$$

Escalonamento em Sistemas de Tempo Real

Ex. Em um sistema de tempo real não crítico, com 3 eventos periódicos, com, respectivamente, períodos (Pi) de:

100, 200 e 500 ms

Esses eventos requerem 50, 30 e 100ms de tempo de UCP (Ci), nesta ordem, o sistema é escalonável porque:

0,5 + 0,15 +0,2 <1

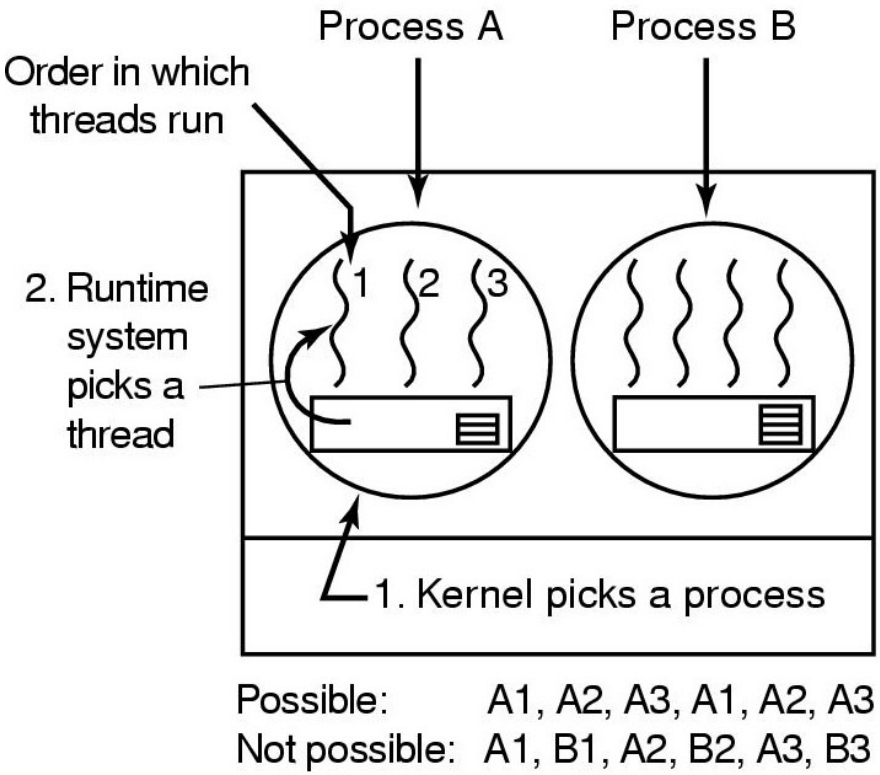
POLÍTICA X MECANISMO

- Algumas vezes é importante separar o mecanismo de escalonamento da política de escalonamento.

- Por exemplo, quando um processo-pai possui vários processos-filhos, pode ser importante deixar o processo-pai interferir no escalonamento dos filhos.

- Para isto, deve se ter um escalonador parametrizado, cujos parâmetros são passados pelos processos de usuário.
- Desse modo, o mecanismo de escalonamento está no núcleo, mas a política é estabelecida por um processo de usuário

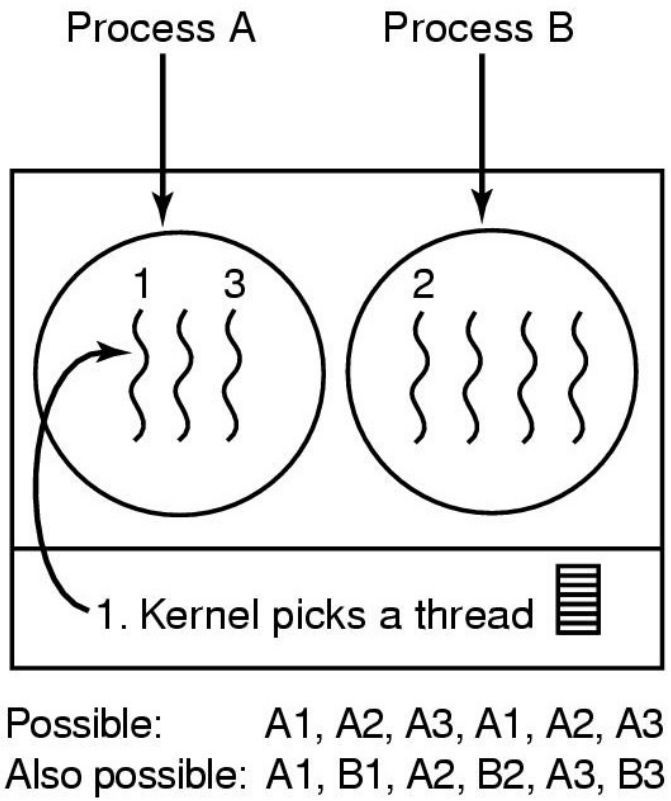
ESCALONAMENTO DE THREADS



Possível escalonamento de threads ao nível do usuário

- Quantum do processo é 50-msec threads
- Executam por surto de 5 msec/CPU

ESCALONAMENTO DE THREADS



Possível escalonamento de threads ao nível do núcleo

- Quantum do processo é 50-msec threads
- Executam por surto de 5 msec/CPU