Sistemas Operacionais

Prof. Dr. Helder Oliveira

Plano de Aula

- Escalonamento

Escalonamento

- Computador multiprogramado.
  - Múltiplos processos ou threads competindo pela CPU ao mesmo tempo.
  - Dois ou mais processos prontos.
  - Que processo executará?
- **Escalonador**
  - **Algoritmo de Escalonamento**

Escalonador?

- Escalonamento de processos.
- Escalonamento de threads

- Núcleo gerencia threads – não importa o processo.

Escalonamento

- Antigos sistemas em lote.
  - Simplicidade no escalonamento.
- Multiprogramação.
  - Escalonamento mais complexo.
  - Vários usuários esperando pelo serviço.

- Tempo de CPU é um recurso escasso .
  - Necessidade de bons algoritmos de escalonamento.

Mudanças

- Um processo ativo na maior parte do tempo.
- Evolução dos computadores.
  - CPU não é recurso escasso.
- Dois programas executados ao mesmo tempo.
  - Não importa qual deles vai primeiro.
  - Como consequência, o escalonamento não importa muito em PCs simples.
- Computadores em rede.
- Dispositivos móveis.

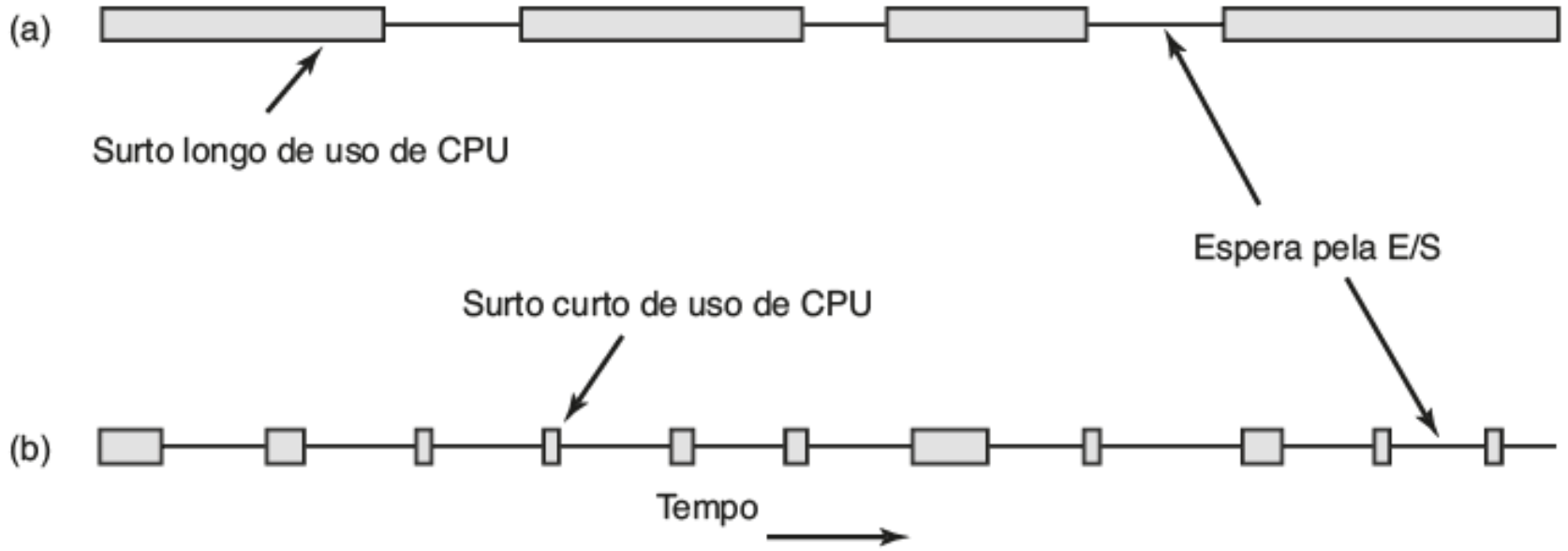
Escalonador

- Chaveamento de processos é algo caro.
  - Troca do modo usuário para o modo núcleo.
  - Salvar estado do processo atual.
  - Um novo processo precisa ser selecionado.

- Troca de processos por segundo pode consumir muito CPU.

Comportamento de processos

FIGURA 2.39 Surtos de uso da CPU alternam-se com períodos de espera por E/S. (a) Um processo limitado pela CPU. (b) Um processo limitado pela E/S.



Comportamento de Processos

- Processos:
  - Limitados pela computação ou Limitados pela CPU;
  - Limitados pela E/S
- Observe que o fator chave é o comprimento do surto da CPU, não o comprimento do surto da E/S.
- CPUs ficam mais rápidas → processos tendem a ficar mais limitados pela E/S.
- Processos limitados pela E/S ocupa pouco a CPU.

Quando escalonar?

- Há uma série de situações nas quais o escalonamento é necessário!

1. Processo criado - Pai ou o filho, deve ser executado?
2. Ao fim de um processo - Qual o próximo a ser executo?
3. Processo bloqueado - Qual o próximo a ser executo?
4. Interrupção de E/S

Quando escalonar?

- **Como lidar com interrupções de relógio.**

1. **Não preemptivo**
   - Deixa ser executado até que ele seja bloqueado ou liberado pelo CPU.
2. **Preemptivo**
   - Deixa executar por no máximo um certo tempo fixado.

Categorias de algoritmos de escalonamento

1. Lote.
2. Interativo.
3. Tempo real

Objetivos do algoritmo de escalonamento

FIGURA 2.40 Algumas metas do algoritmo de escalonamento sob diferentes circunstâncias.

Todos os sistemas

Justiça — dar a cada processo uma porção justa da CPU

Aplicação da política — verificar se a política estabelecida é cumprida

Equilíbrio — manter ocupadas todas as partes do sistema

Objetivos do algoritmo de escalonamento

FIGURA 2.40 Algumas metas do algoritmo de escalonamento sob diferentes circunstâncias.

Sistemas em lote

Vazão (*throughput*) — maximizar o número de tarefas por hora

Tempo de retorno — minimizar o tempo entre a submissão e o término

Utilização de CPU — manter a CPU ocupada o tempo todo

Objetivos do algoritmo de escalonamento

FIGURA 2.40 Algumas metas do algoritmo de escalonamento sob diferentes circunstâncias.

Sistemas interativos

Tempo de resposta — responder rapidamente às requisições

Proporcionalidade — satisfazer às expectativas dos usuários

Objetivos do algoritmo de escalonamento

FIGURA 2.40 Algumas metas do algoritmo de escalonamento sob diferentes circunstâncias.

Sistemas de tempo real

Cumprimento dos prazos — evitar a perda de dados

Previsibilidade — evitar a degradação da qualidade em sistemas multimídia

Escalonamento em sistemas em lote

- Algoritmos usados em sistemas em lote.
  - Primeiro a chegar, primeiro a ser servido
  - Tarefa mais curta primeiro
  - Tempo restante mais curto em seguida

Primeiro a chegar, primeiro a ser servido

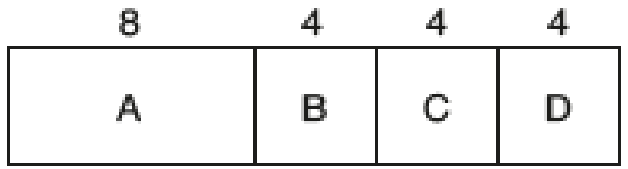
- First-Come, First-Served
- Mais simples de todos os algoritmos de escalonamento.
- Não preemptivo.
- CPU é atribuída aos processos na ordem em que a requisitam.
- Processo bloqueado é colocado no fim da fila.
- Fácil de entender e programar.
- Justo – como quem chega cedo na fila de um ingresso.
- Problema: ter que voltar ao fim da fila sempre que for bloqueado.

Tarefa mais curta primeiro

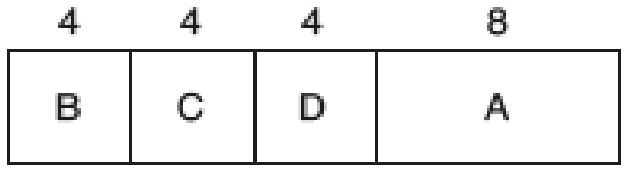
- Shortest job first.
- Tarefa mais curta executa primeiro.
- Não preemptivo.
- Tempos de execução são conhecidos antecipadamente.
- Ex:

Tarefa mais curta primeiro

FIGURA 2.41 Um exemplo do escalonamento tarefa mais curta primeiro. (a) Executando quatro tarefas na ordem original. (b) Executando-as na ordem tarefa mais curta primeiro.



(a)



(b)

Tempo restante mais curto em seguida

- Shortest remaining time next.
- Versão preemptiva da tarefa mais curta primeiro.
- Tempo de execução precisa ser antecipado.
- Tarefas curtas novas tem bom desempenho.

Escalonamento em sistemas interativos

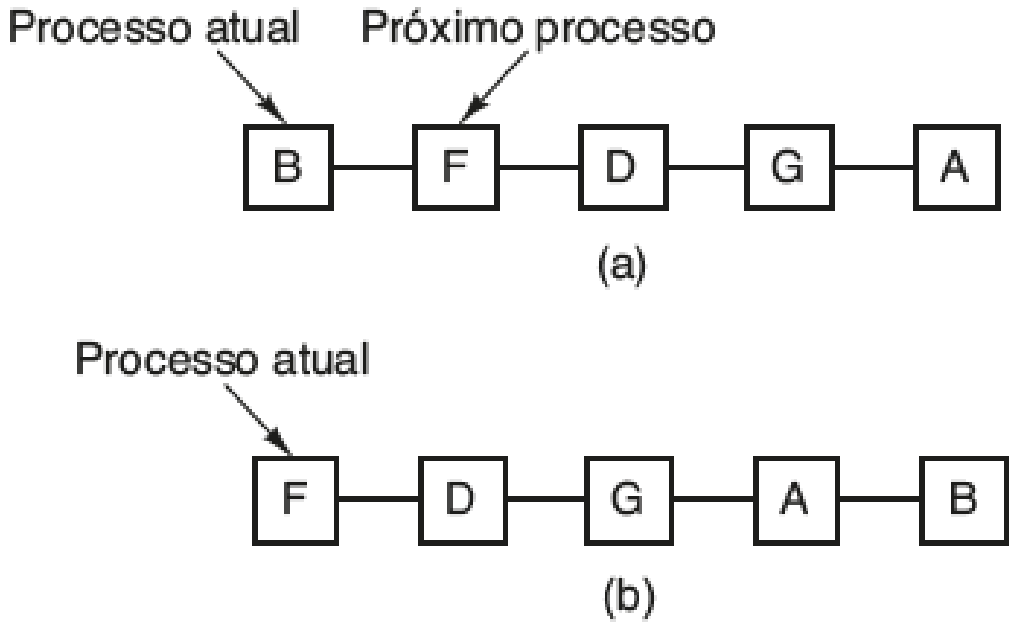
- Comuns em computadores pessoais, servidores e outros tipos de sistemas.
- Algoritmos usados em sistemas interativos.
  - Escalonamento por chaveamento circular.
  - Escalonamento por prioridades.
  - Múltiplas filas.
  - Processo mais curto em seguida.
  - Escalonamento garantido
  - Escalonamento por loteria
  - Escalonamento por fração justa

Escalonamento por chaveamento circular

- Round-robin
- Antigos, simples, justos e amplamente usados.
- Uso do **quantum.**
- Fácil implementar.
- Mantem uma lista de processos executáveis.
- Comprimento do quantum?
  - Chaveamento de processo ou chaveamento de contexto.
  - Um quantum em torno de 20-50 ms é muitas vezes bastante razoável

Escalonamento por chaveamento circular

FIGURA 2.42 Escalonamento circular. (a) A lista de processos executáveis. (b) A lista de processos executáveis após *B* usar todo seu quantum.

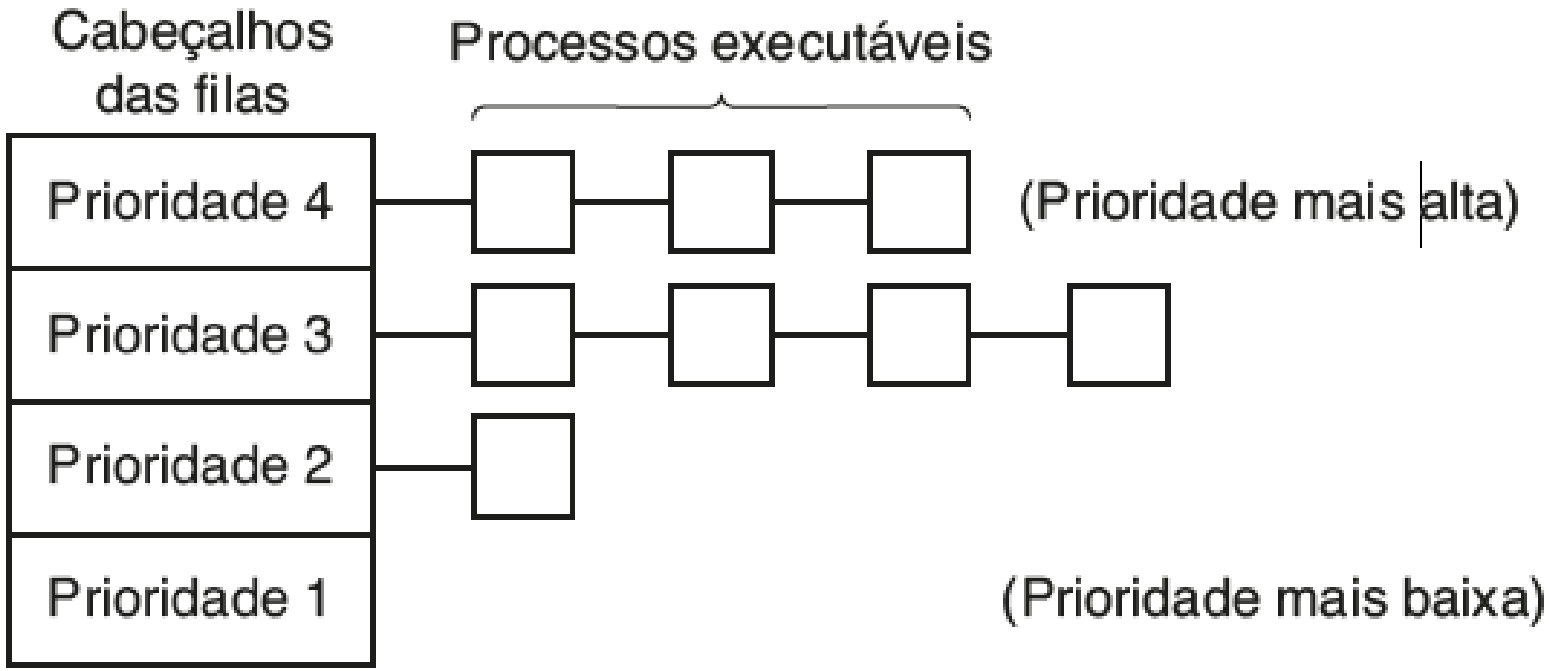


Escalonamento por prioridades

- A cada processo é designada uma prioridade, e o processo executável com a prioridade mais alta é autorizado a executar.
- Escalonador pode diminuir a prioridade do processo que está sendo executado em cada tique do relógio (interrupção do relógio).
  - Alternativa usar um quantum de tempo máximo.

Escalonamento por prioridades

FIGURA 2.43 Um algoritmo de escalonamento com quatro classes de prioridade.



Múltiplas Filas

- Dá aos processos limitados pela CPU um grande quantum de vez em quando.
- Processos na classe mais alta seriam executados por dois quanta.
- Processos na classe seguinte seriam executados por quatro quanta etc.
- Processo consumia todos os quanta alocados para ele, era movido para uma classe inferior.

Processo mais curto em seguida

- Executar a mais curta primeiro.
- Problema é descobrir qual dos processos atualmente executáveis é o mais curto.
- Uma abordagem é fazer estimativas baseadas no comportamento passado e executar o processo com o tempo de execução estimado mais curto.
- Envelhecimento (aging).

Escalonamento garantido

- Faz promessas reais para os usuários a respeito do desempenho e então cumpri.
- Ex: Se n usuários estão conectados enquanto você está trabalhando, você receberá em torno de 1/n da potência da CPU.

Escalonamento por loteria

- Mais simples que o escalonamento garantido.
- A ideia básica é dar bilhetes de loteria aos processos para vários recursos do sistema, como o tempo da CPU.
- Processos cooperativos podem trocar bilhetes se assim quiserem.

Escalonamento por fração justa

- Qual usuário é dono de um processo?
- Cada usuário é alocada alguma fração da CPU e o escalonador escolhe processos de uma maneira que garanta essa fração.

Escalonamento em sistemas de tempo real

- Um sistema de tempo real é aquele em que o tempo tem um papel essencial.
- Algoritmos de escalonamento de tempo real podem ser
  - Estáticos.
  - Dinâmicos.

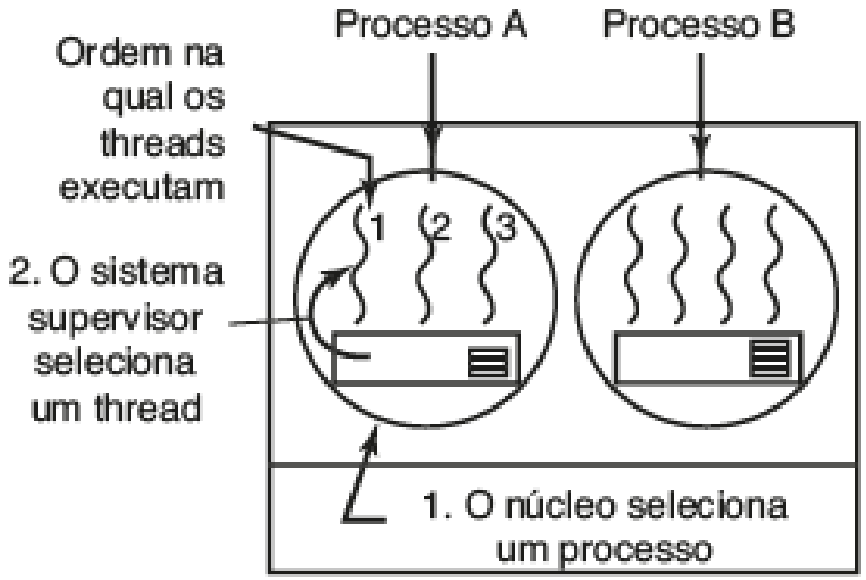
Política versus mecanismo

- Processos filhos disputando CPU.
- Processo pai sabe a importância.
- Processo pai pode estabelecer e mudar prioridades dos filhos.
- Pai pode controlar como seus filhos são escalonados.
- Mecanismo está no núcleo, mas a política é estabelecida por um processo do usuário.

Escalonamento de threads

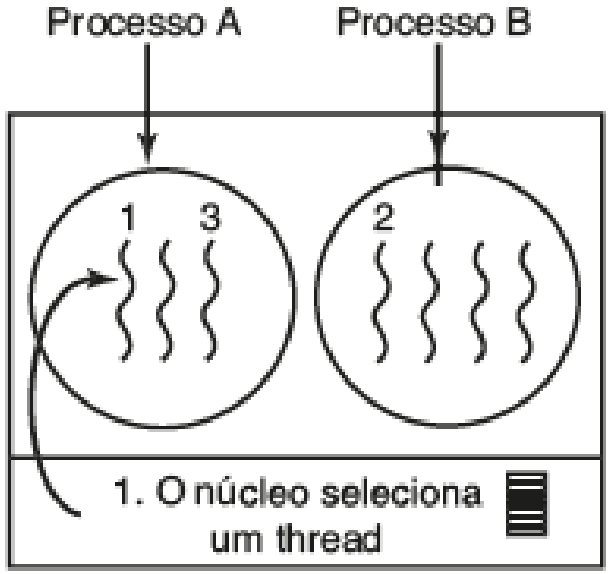
- Processos com múltiplos threads.
- Recebem suporte:
  - Threads de usuário
    - Núcleo não tem ciência da existência das threads.
    - Formas de tratar:
      - Escalonador de thread dentro de A decide qual thread executar, digamos, A1 e concluirá A1 antes da próxima thread.
      - Cada thread executa por um tempo, então cede a CPU de volta para o escalonador de threads. Isso pode levar à sequência A1, A2, A3, A1, A2, A3, A1, A2, A3, A1, antes que o núcleo chaveie para o processo B.
    - Comportamento da thread não afetará outro processo.
  - Threads de núcleo
    - Não precisa levar em conta a qual processo o thread pertence, porém ele pode.

Escalonamento de threads



Possível: A1, A2, A3, A1, A2, A3
Impossível: A1, B1, A2, B2, A3, B3

(a)



Possível: A1, A2, A3, A1, A2, A3
Também possível: A1, B1, A2, B2, A3, B3

(b)

Escalonamento de threads

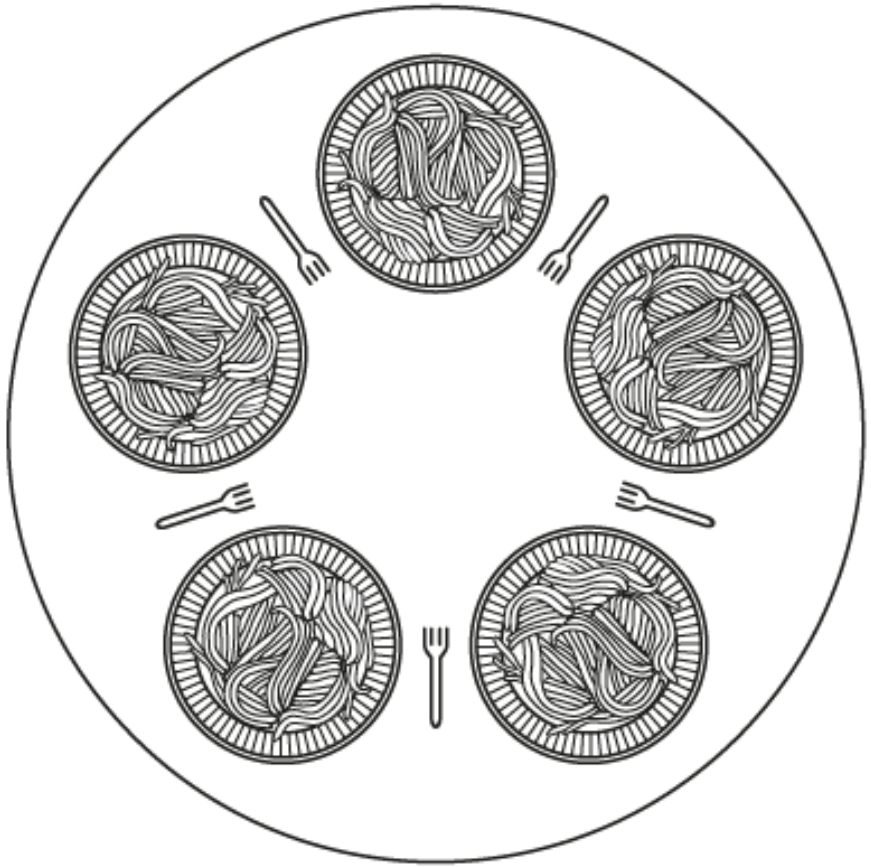
- Realizar um chaveamento de thread com threads de usuário exige um punhado de instruções de máquina.
- Com threads de núcleo é necessário um chaveamento de contexto completo, mudar o mapa de memória e invalidar o cache, o que representa uma demora de magnitude várias ordens maior.
- Com threads de núcleo, ter um bloqueio de thread na E/S não suspende todo o processo como ocorre com threads de usuário.
- Threads de usuário podem empregar um escalonador de thread específico de uma aplicação.

Problemas clássicos de IPC

- O problema do jantar dos filósofos
- O problema dos leitores e escritores

Problema do jantar dos filósofos

FIGURA 2.45 Hora do almoço no departamento de filosofia.



Problema dos leitores e escritores

- Um sistema de reservas de uma companhia aérea, com muitos processos competindo entre si desejando ler e escrever. É aceitável ter múltiplos processos lendo o banco de dados ao mesmo tempo, mas se um processo está atualizando (escrevendo) o banco de dados, nenhum outro pode ter acesso, nem mesmo os leitores. A questão é: como programar leitores e escritores?

Leitura

- SISTEMAS OPERACIONAIS MODERNO 4ª edição
  - 2.4 Escalonamento

Dúvidas?