UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE RUSSAS

Rus0013 - Sistemas Operacionais

Aula 07: Escalonamento

Professor Pablo Soares

2022.2

Sumário

- Escalonamento
- Comportamento do processo
- Quando Escalonar
- Categorias de Algoritmos de Escalonamento
  - Lote
  - Interativo
  - Tempo Real
- Escalonamento de Threads

Escalonamento

- Quando um computador é multiprogramado há vários processos competindo pela CPU num dado instante, quando o no. De processos no estado **pronto** é maior que o de CPU
- O SO deve escolher, através do **escalonador**, qual processo executará, usando um **algoritmo de escalonamento**

Escalonamento

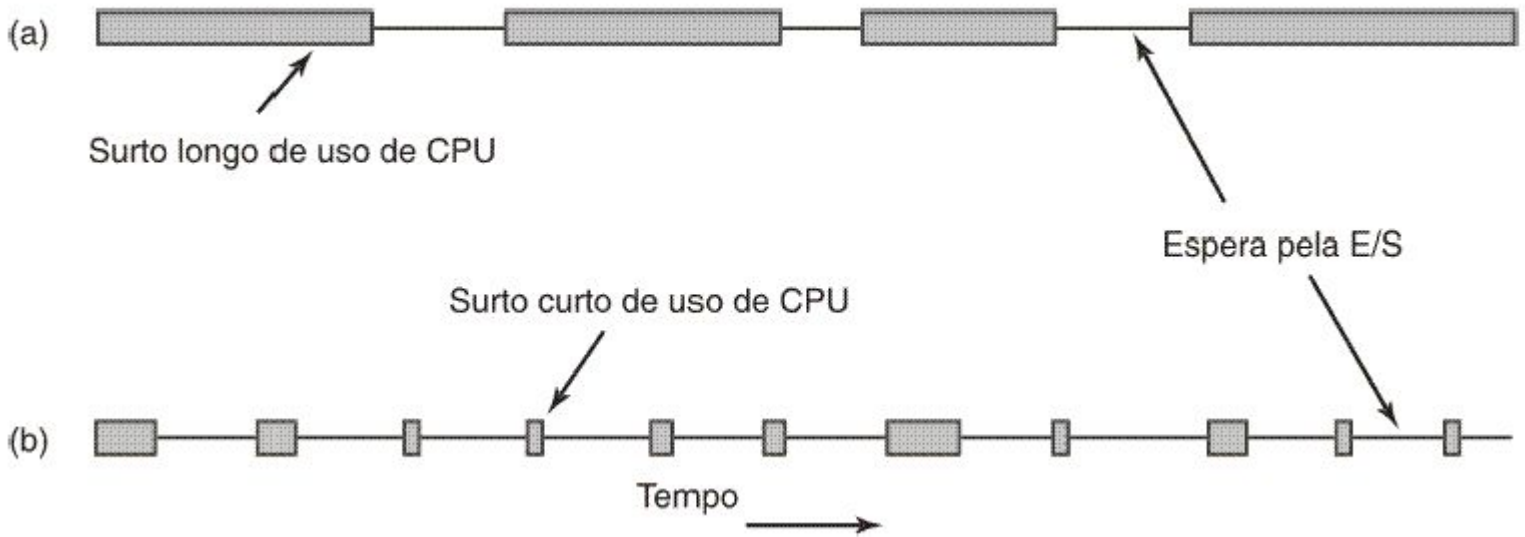
- Além de escolher o processo certo para executar, o escalonador deve se preocupar com o uso eficiente da CPU, pois alternar processos é muito caro
  - De modo usuário para modo núcleo
  - O estado atual do processo deve ser salvo
  - O mapa de bits de memória também deve ser salvo
  - A MMU deve carregar o mapa de bits do novo processo
  - O novo processo deve ser iniciado
  - Invalidação de toda a memória cache

Escalonamento

- De modo geral muitas alternâncias de processos por segundo pode comprometer boa parte do tempo de CPU com o próprio escalonamento, o que é indesejável

Comportamento do Processo

- Alguns processos gastam a maior parte do tempo
- computando e outros esperando E/S
- Surtos de uso da CPU alternam-se com períodos de espera por E/S
  - a) um processo orientado à CPU
  - b) um processo orientado à E/S



Comportamento do Processo

- Alguns tipos de E/S são considerados computação, como escrever na memória de vídeo
- A medida que as CPUs se tornam mais rápidas, os processos tendem a ficar mais orientados a E/S
  - CPU estão ficando muito mais rápidas que os discos
- Uma regra básica é que se um processo orientado a E/S quiser executar, ele deve ser rapidamente atendido, pois assim ele executará suas requisições de disco, o mantendo ocupado

Quando Escalonar

- Dentre as principais situações que levam ao escalonamento temos

    1. Quando se cria um novo processo, é necessário tomar a decisão entre executar o processo pai e o filho

    2. Quando um processo termina, algum outro processo deve ser escolhido entre os processos prontos

Quando Escalonar

- Dentre as principais situações que levam ao escalonamento temos

  3. Quando um processo bloqueia em uma E/S, na entrada de uma região crítica, ou por outro motivo, outro processo precisa ser escolhido para executar. O motivo do bloqueio pode ser relevante

- Caso um processo “A” bloqueie na entrada de uma região crítica convém escolher o processo que já está nela “B”, para que “A” passe ao estado de pronto

Quando Escalonar

- Dentre as principais situações que levam ao escalonamento temos

  4. Quando ocorre uma interrupção de E/S (de um dispositivo que terminou seu trabalho) um processo irá passar do estado de bloqueado para pronto.

  - O escalonador deve escolher entre continuar executando o **processo atual**, executar o processo que acabou de ficar **pronto** ou um **terceiro processo** qualquer

Quando Escalonar

- Dentre as principais situações que levam ao escalonamento temos

  5. Se o hardware de relógio oferece interrupções periódicas (ex: 50 Hz ou 60 Hz) uma decisão de escalonamento deve ser tomada a cada interrupção

- Os algoritmos de escalonamento podem ser de 2 formas com relação as interrupções de relógio:
  - **Preemptivo**
  - **Não preemptivo**

Quando Escalonar

- **Algoritmo não preemptivo**
  - Escolhe um processo e então o deixa executar até seja bloqueado ou até que deixe a CPU voluntariamente. Mesmo que execute por horas, não será retirado da CPU
- **Algoritmo preemptivo**
  - Escolhe um processo e o deixa executar, no máximo, um tempo previamente fixado. Se ainda estive executando ao final desse tempo, o escalonador escolherá um outro processo para executar
- Se não houver relógio disponível, o escalonamento não preemptivo será a única opção

Categorias de algoritmos de escalonamento

- Três ambientes diferentes
  - Lote
    - Não usuários esperando, então algoritmos não preemptivos ou preemptivos com longos intervalos são aceitáveis
  - Interativo
    - Preempção é essencial para evitar que um processo seaposse da CPU e com isso negue serviços aos outros
  - Tempo real
    - Preempção é desnecessária, pois os processos sabem que não devem executar por longos períodos e em geral fazem seu trabalho e bloqueiam rapidamente

Objetivos do algoritmo de escalonamento

- Um bom algoritmo de escalonamento pode ter seus objetivos alterados em função do ambiente em que ele executa
- Levam em consideração
  - **Vazão:** é o número de jobs por hora que o sistema termina
  - **Tempo de retorno:** estaticamente – o tempo médio do momento em que um job em lote é submetido até o momento em ele é terminado
  - **Tempo de resposta:** tempo entre a emissão de um comando e a obtenção do resultado
  - **Proporcionalidade:** requisições interativas com precedência sobre processos em 2º plano

Objetivos do algoritmo de escalonamento

Todos os sistemas

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

Sistemas em lote

Vazão (throughput) — maximizar o número de jobs por hora
Tempo de retorno — minimizar o tempo entre a submissão e o término
Utilização de UCP — manter a UCP ocupada o tempo todo

Sistemas interativos

Tempo de resposta — responder rapidamente às requisições
Proporcionalidade — satisfazer as expectativas dos usuários

Sistemas de tempo real

Cumprimento dos prazos — evitar a perda de dados
Previsibilidade — evitar a degradação da qualidade em sistemas multimídia

Escalonamento em sistemas de lote

- Primeiro a chegar, primeiro a ser servido (FCFS - FIFO)
  - Processos são colocados em uma fila à medida que são criados
  - O algoritmo de escalonamento simplesmente sempre escolhe o 1º processo da fila
  - Quando um processo bloqueia e retorna ao estado pronto ele também é colocado no final da fila
  - Vantagem se ser fácil de entender e implementar
  - Processos orientados a E/S pode demorar muito

Escalonamento em sistemas de lote

- Job mais curto primeiro
  - Em muitos casos, em um sistema em lote, pode-se prever quanto tempo um job levará para executar
  - Este algoritmo seleciona sempre primeiramente os jobs mais curtos

(a) Execução na ordem original; (b) Execução na ordem *job mais curto primeiro*

  - A média em tempo de retorno (a) é 14 e em (b) é 11

Escalonamento em sistemas de lote

- Contraexemplo (Job Mais curto)

  Considere cinco tarefas A a E, com tempos de execução:
  - 2, 4, 1, 1, 1.
  - Tempos de chegada são 0, 0, 3, 3, 3

- Job mais curto A, B, C, D, E

  Média 4,6
- Outra composição B, C, D, E, A

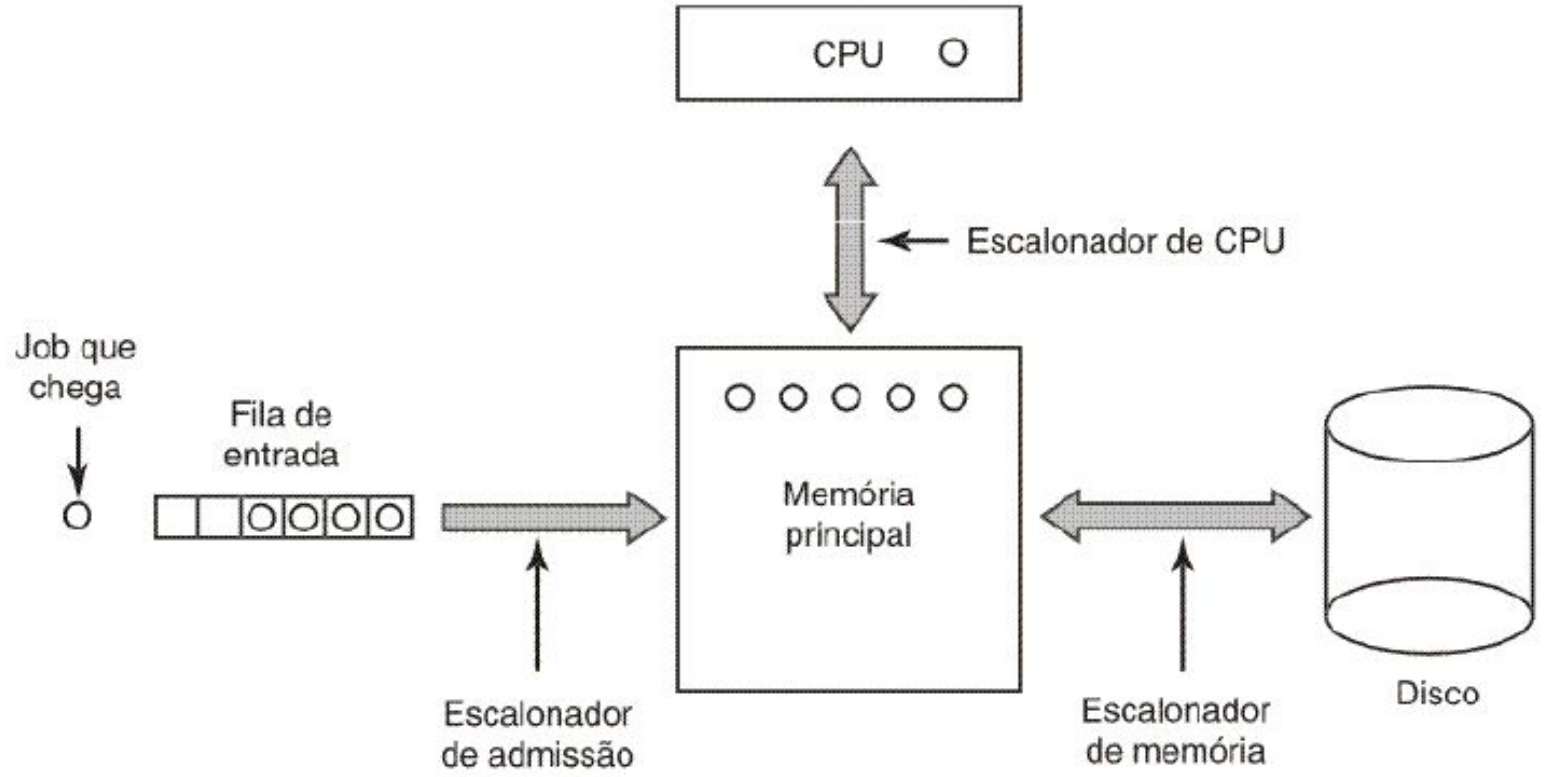
  Média 4,4

Escalonamento em sistemas de lote

- Próximo de menor tempo restante
  - É uma variação do job mais curto primeiro onde o escalonador escolhe sempre o job cujo tempo de execução restante ao seu término seja o menor
  - O tempo tem que ser previamente conhecido
  - Quando chega um novo job, seu tempo total é comparado ao tempo restante do processo em curso

Escalonamento em sistemas de lote

- Escalonamento em 3 níveis



Escalonamento em sistemas interativos

- Os algoritmos para sistemas interativos também podem ser utilizados em sistemas de lote
  - Escalonamento *round-robin*
  - Escalonamento por prioridade
  - Escalonamento garantido
  - Escalonamento por loteria
  - Escalonamento fração justa

Escalonamento em sistemas interativos

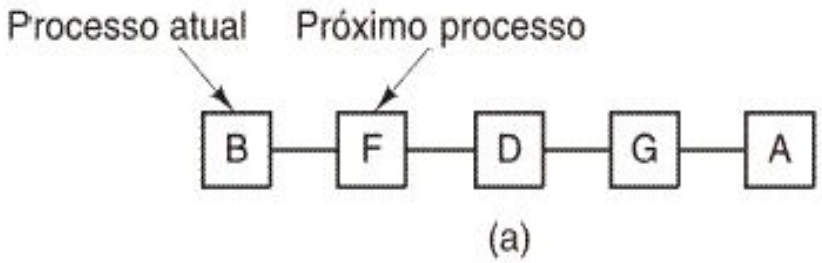
- Escalonamento *round-robin*
  - Bastante antigo, simples, justo e muito usado
  - Cada processo recebe um intervalo de tempo máximo (**quantum**) durante o qual pode executar
  - Se ao final de seu **quantum** o processo ainda estiver executando ele é escalonado
  - O SO mantém uma lista de processos executáveis (estado pronto), quando um processo é escalonado ele é colocado no final da fila

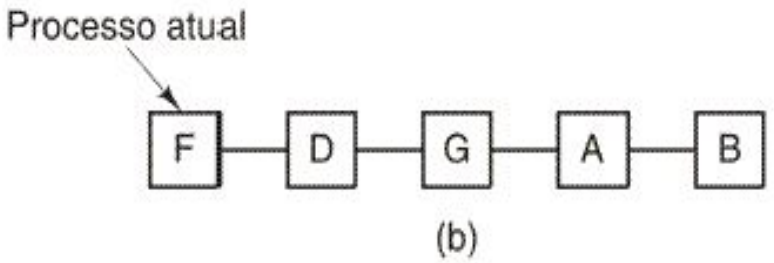
Escalonamento em sistemas interativos

- Escalonamento por alternância circular (*round-robin)*

  a) lista de processos executáveis

  b) lista de processos executáveis depois que B usou todo o seu quantum





Escalonamento em sistemas interativos

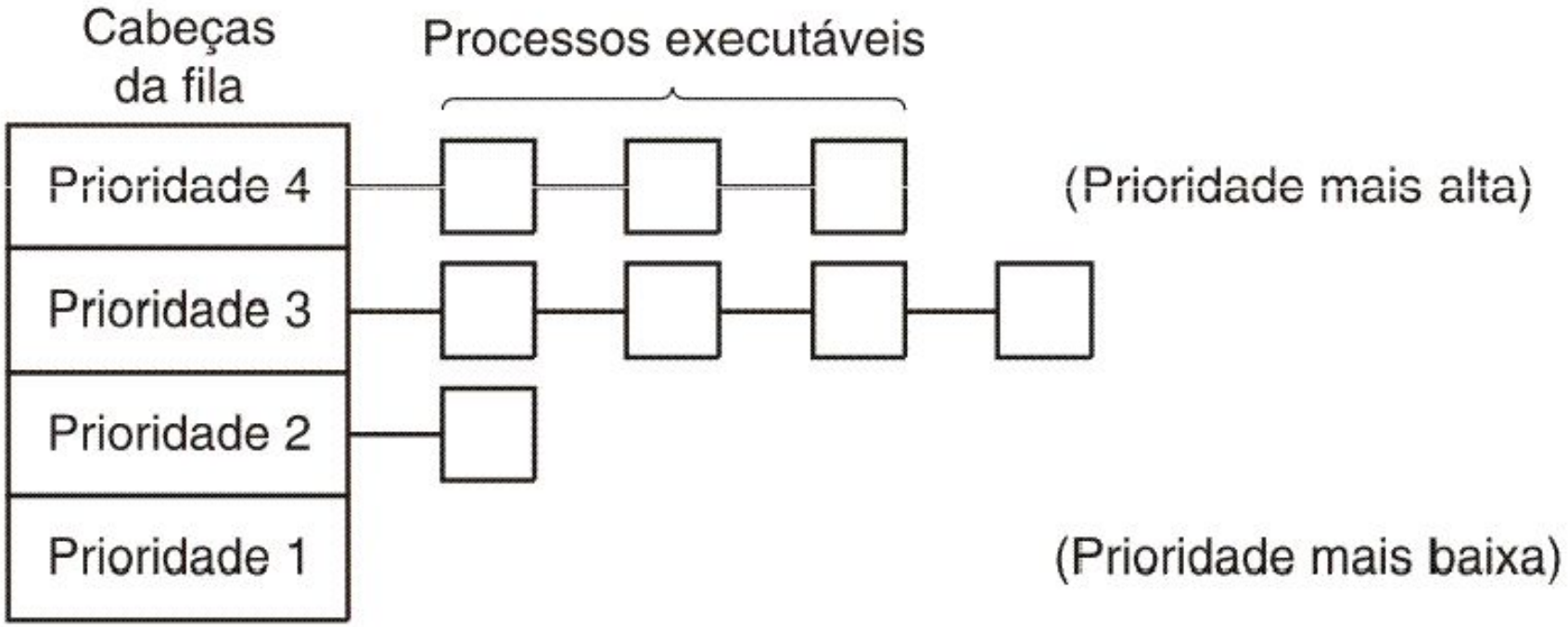
- Escalonamento *round-robin*
  - Suponha chaveamento de contexto = 1ms
    - quantum = 4 ms
      - 20% do tempo em administração
    - quantum = 100ms
      - 1% do tempo em administração
      - Suponha 50 solicitações
        - O último cara vai esperar 5s
- Conclusão
  - Para este algoritmo é fundamental a escolha de um valor de quantum adequado. Implementações reais usam valores de 20 a 50 ms

Escalonamento em sistemas interativos

- Escalonamento **por prioridades**
  - *Round-robin* assume que todos os processos possuem uma mesma importância ou prioridade
  - Este modelo mantém a idéia de quantum máximo de execução, adicionando a cada processo uma prioridade
  - No momento do escalonamento, o processo pronto com maior prioridade é escolhido para executar
  - Existem diversas técnicas que ajustam a prioridade de cada processo de forma dinâmica

Escalonamento em sistemas interativos

- Escalonamento **por prioridades**



Escalonamento em sistemas interativos

- Escalonamento **garantido**
  - Neste método se houver *n* usuários conectados em uma máquina, cada um deles receberá 1/*n* do tempo total da CPU
  - De forma semelhante, se houver *n* processo em uma máquina monousuário, cada processo receberá 1/*n* da CPU
  - O sistema mantém um controle do tempo de CPU que cada processo recebeu desde sua criação

Escalonamento em sistemas interativos

- Escalonamento **por loteria**
  - Este método se baseia na idéia de distribuir "**bilhetes**" ao processos
  - Quando há um escalonamento um "**bilhete**" é sorteado, e o processo que o detém ganha acesso ao recurso (ex: cada "**bilhete**" pode representar o direito a um quantum de CPU)
  - Processos podem receber número diferentes de **bilhetes**, de forma a se ter diferentes probabilidade de escolha
  - Também há implementações onde existem as ações como compra, venda, empréstimo e troca de bilhetes

Escalonamento em sistemas interativos

- Escalonamento **fração justa** (*fair-share*)
  - Um outro fator importante é que há uma série de propriedades dos processos que devem ser levadas em conta no momento do escalonamento
  - Ex: se temos 2 usuários conectados em uma máquina, um com 9 e outro com 1 processo, não é justo que o 1º ganhe 90% do tempo da CPU
  - Vale salientar que um sistema real sempre irá se utilizar de um subconjunto ou de todas estas técnicas ao mesmo tempo

Escalonamento em sistemas de tempo real

- É aquele no qual o tempo tem uma função essencial
- Sistemas de tempo real são em geral categorizados como
  - **Tempo real crítico**: há prazos absolutos que devem ser cumpridos
  - **Tempo real não crítico**: o descumprimento ocasional de um prazo é indesejável, contudo tolerável

Escalonamento em sistemas de tempo real

- Nestes sistemas, 1 ou mais dispositivos geram estímulos, e o computador deve reagir a eles dentro de um intervalo de tempo máximo garantido

- Ter uma resposta correta mas em um tempo tardio é tão ruim como não ter nada

Escalonamento em sistemas de tempo real

- Eventos podem ser periódicos ou aperiódicos (modo imprevisível)
  - O trabalho do escalonador é escalonar os processos de tal maneira que todos os prazos sejam cumpridos
  - Existem diversas regras a serem seguidas para classificar um STR como escalonável
  - Isto garante que mesmo no pior caso, os prazos de todos os eventos serão cumpridos

Escalonamento em sistemas de tempo real

- Um sistema de tempo real que satisfaça esse critério e chamado de **escalonável**
-

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

- Considere três eventos periódicos
  - 100, 200 e 500 ms
  - Cada evento requer 50, 30 e 100 ms de tempo de CPU
  - Teríamos 0,5 + 0,15 + 0,2 < 1

Escalonamento de threads

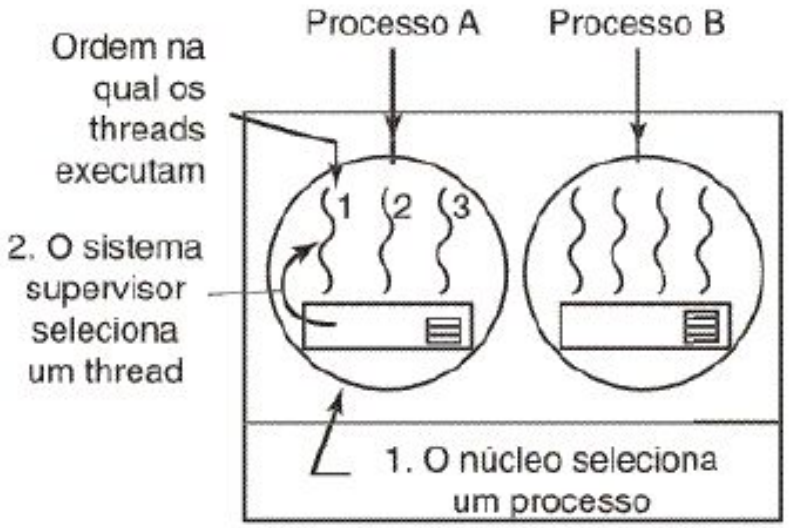
- O escalonamento difere
  - Threads usuário
  - Threads núcleo
- No caso de **threads de usuário**, o SO escolhe um processo "A" para executar dando a ele o seu quantum
  - O sistema supervisor do processo "A" escolhe qual thread deve executar (ex: A1, depois A3, depois A2, etc)
  - O sistema supervisor pode se utilizar de qualquer uma das técnicas anteriormente descritas

Escalonamento de threads

- No caso de **thread de núcleo** o SO escolhe um thread para executar durante um quantum
- O SO pode levar em conta ou não a qual processo cada thread pertence
- Esta informação pode ser importante
  - Se a thread bloquear antes do fim do seu quantum
- Neste caso se houver outras threads do mesmo processo prontas para executar, será bem mais rápido escolher uma delas, que uma thread de um outro processo

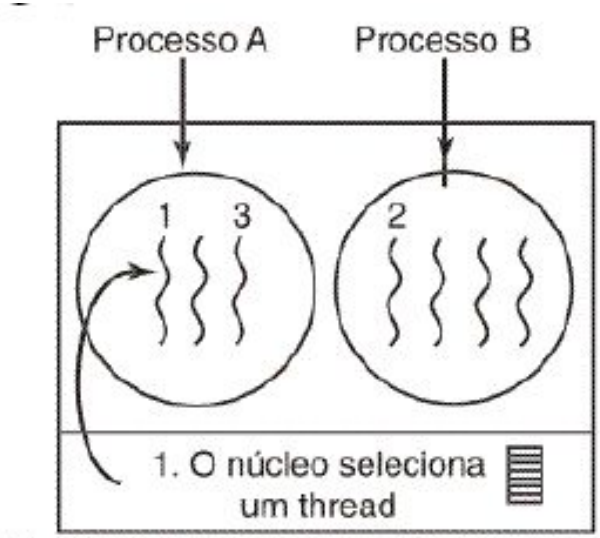
Escalonamento de threads

- Possível escalonamento de threads de usuário e núcleo
  - processo com quantum de 50 *m*seg
  - threads executam 5 *m*seg por surto de CPU



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)

Referências

- Andrew S. Tanenbaum. “Sistemas Operacionais Modernos”. 3ª Edição, Prentice Hall, 2010.

UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE RUSSAS

Rus0013 - Sistemas Operacionais

Aula 07: Escalonamento

Professor Pablo Soares

2022.2