Sistemas Operacionais

4º período

Professora: Michelle Hanne

PUC Minas

Escalonamento
- Revisão –

baseado em http://www.univasf.edu.br/~andreza.leite/aulas/SO/ProcessosEscalonamento.pdf

PUC Minas

Escalonamento

- Definição:
  - O escalonamento consiste em distribuir o acesso aos recursos do sistema entre os processos que o solicitam.
- Objetivo:
  - Otimizar o rendimento dos recursos.
  - Priorizar o acesso aos recursos disponíveis.
- Recursos que necessitam escalonamento:
  - Dispositivos E/S (discos)
  - Processador
  - Memória

Escalonamento

- Multiprogramação:
  - O S.O. gerencia múltiplos processos na memória principal de forma simultânea.
  - Os processos devem compartilhar o acesso ao processador.
- Escalonamento de processos:
  - Decidir sobre:
    - Que trabalhos serão admitidos pelo sistema
    - Que processos serão mantidos na memória principal
    - Que processo utilizará a CPU quando ela estiver livre

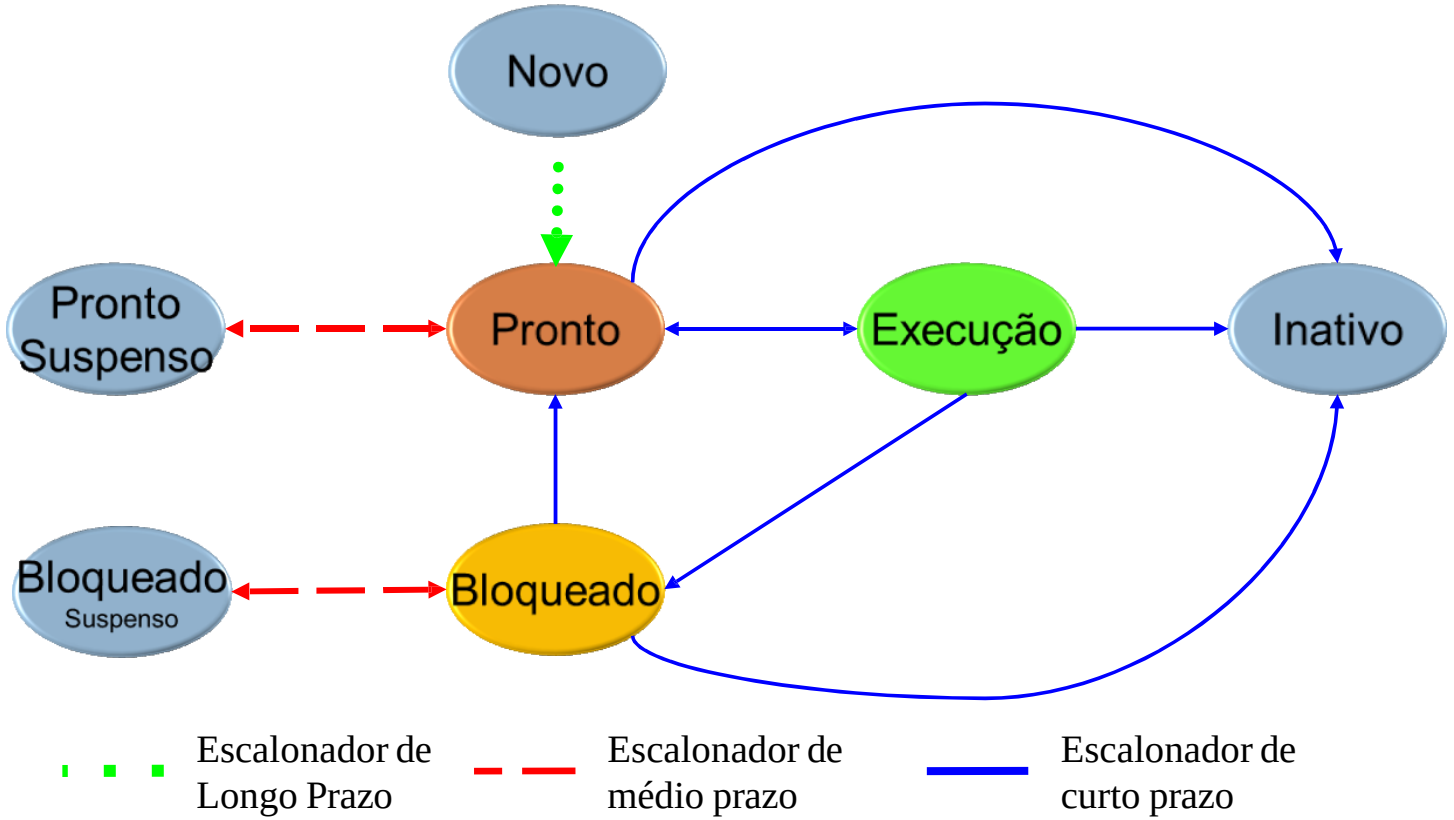
O escalonador de processos é o responsável de tomar estas decisões, repartindo o uso da memória e do processador entre os processos ativos do sistema.

Escalonamento

- Tipos de Escalonadores:
  - Escalonador de médio prazo
    - É o responsável de escolher os processos que serão removidos total ou parcialmente da memória para serem levados ao disco (suspensos)
    - Manter rendimento do sistema
  - Escalonador de curto prazo
    - Responsável por alocar à CPU os processos alocados em memória

PUC Minas

Escalonamento



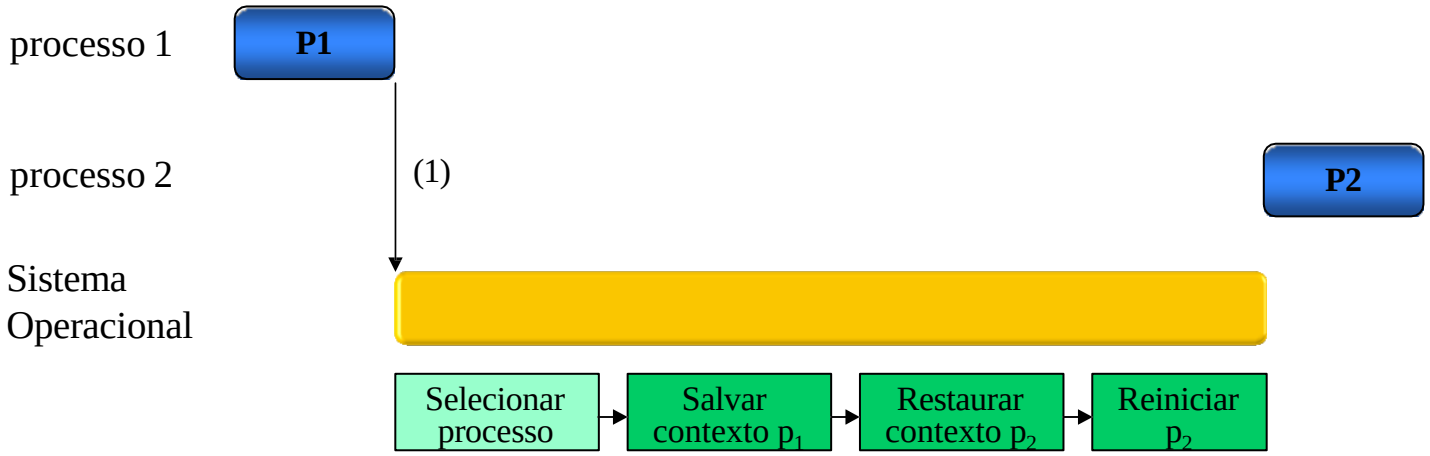
Escalonador:Curto Prazo

- Escalonador
  - Seleciona o processo para sua execução, atendendo a um determinado critério.
- *Dispacher* (despachador)

  É o módulo que dá controle da CPU para o processo selecionado pelo escalonador de curto prazo.
  - Salvar contexto do processo que sai da cpu
  - Restaurar contexto do processo que entra na cpu
  - Reiniciar a execução de processos
    - Alterar para estado pronto.
    - Configurar para o ponto apropiado do programa

Troca de contexto

Troca de processos



(1) Finalização do tempo de execução ou o processo se bloqueia à espera de um recurso que necessita

Escalonador Despachador

Troca de processos

- processos intensivos em CPU

  as etapas de CPU são maiores que as de E/S.

CPU BOUND:

processo CPU intensivo:

- processos intensivos en E/S

  as etapas de E/S são maiores que as de CPU.

IO BOUND:

processo E/S intensivo:

Escalonamento

- Escalonar...
  - Divisão equitativa do processador
  - Otimizar alguns critérios:
    - Grau de utilização da CPU.
    - Produtividade (*throughput*).
      - Número de processos terminados por unidade de tempo
    - Tempo de retorno (*Turnaround time*).
      - Tempo transcorrido desde que se lança um processo (entra na fila de prontos) até que finalize sua execução.
      - É a soma do tempo de espera para ir para a memória, tempo de espera na fila dos prontos, tempo em execução na CPU e o tempo de espera por recursos.

PUC Minas

Escalonamento

- O escalonador ideal
  - É aquele que consegue deixar a CPU 100% ocupada.
  - Objetivo
    - Maximizar a produtividade
    - Minimizar o tempo de retorno, resposta e espera.
  - Não existe nenhuma política de escalonamento ótima:
    - Cumprir com todos critérios anteriores
  - A política de escalonamento conveniente depende:
    - Tipo de processo.
    - Critério de otimizarão desejado.

PUC Minas

Escalonamento

- Algoritmos de Escalonamento:
  - Algumas políticas de escalonamento podem funcionar em modo não preemptivo ou em modo preemptivo.
    - Modo não preemptivo:
      - O processo que possui a CPU somente a libera quando quer (quando acaba sua execução)
      - Não necessita suporte de hardware adicional
      - Um processo pode monopolizar a CPU
      - Não são convenientes para ambientes de tempo compartilhado.
        - Exemplo: Windows 3.1 e Apple Macintosh OS

PUC Minas

Escalonamento

- Algoritmos de Escalonamento:
  - Algumas políticas de escalonamento podem funcionar em modo não preemptivo ou em modo preemptivo.
    - Modo preemptivo:
      - O escalonador pode desalocar um processo da CPU em qualquer instante de tempo.
      - Maior custo, porém evita-se que um processo tenha 100% da CPU

PUC Minas

Escalonamento

- Algoritmos de Escalonamento:
  - Não Preemptivos
    - *Fisrt-Came, Fisrt-Served* – FCFS (FIFO)
    - *Shortest-Job-First* – SJF
  - Preemptivos
    - Por prioridades
    - Turno rotativo ou Circular (*Round-Robin*)
    - Filas multi-nivel
    - Tempo Real

Escalonamento - Exemplo

- Para os exemplos dos algoritmos de escalonamento vamos supor a existência de 3 processos com as seguintes características:

Processo	Tempo de chegada	Etapas do proceso
Processo A	0	7_{CPU}
Proceso B	2	4_{CPU}
Processo C	3	2_{CPU}

- OBS: Considere os *delays* dos tempos de chegada de cada processo.

Não Preemptivos

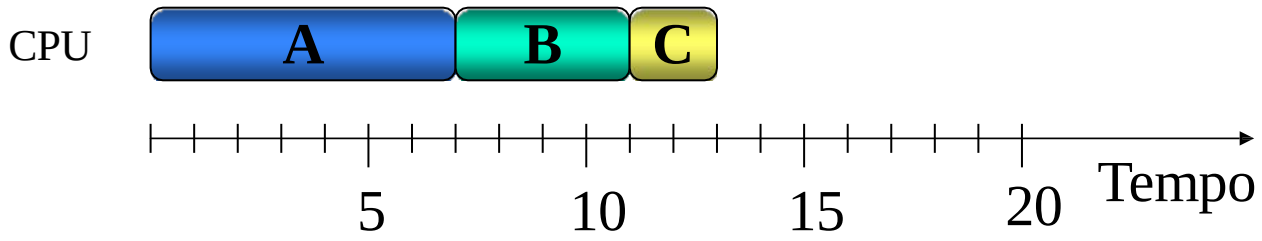
PUC Minas

Algoritmo FCFS

Algoritmo FCFS (*First-Come First-Served*)

- **Funcionamento:**
  - O procesador é alocado seguindo a ordem de chegada dos processos à fila de processos prontos.
  - O processo que tem a CPU não a libera até que acabe sua execução ou até que fique bloqueado por uma operação de E/S.
- **Implementação:**
  - A fila de processos prontos é implementada mediante uma fila FIFO (*First-In First-Out*).

Diagrama de Gant FCFS



$$Utilização_{cpu} = \frac{TCPU_{ocupada}}{Tempo} = \frac{13}{13} = 1 = 100\%$$

$$\text{Produtividade} = \frac{nº\ processos}{tempo} = \frac{3}{13} = 0{,}23$$

$$TEspera_{medio} = \frac{TEspera_A + TEspera_B + TEspera_C}{nº\ processos} = \frac{0 + (7-2) + (11-3)}{3} = \frac{0+5+8}{3} = 4{,}3$$

$$T\,\text{Retorno}_{medio} = \frac{T\,\text{Retorno}_A + T\,\text{Retorno}_B + T\,\text{Retorno}_C}{nº\ processos} = \frac{7+11+13}{3} = 10{,}33$$

PUC Minas

Algoritmo SJF

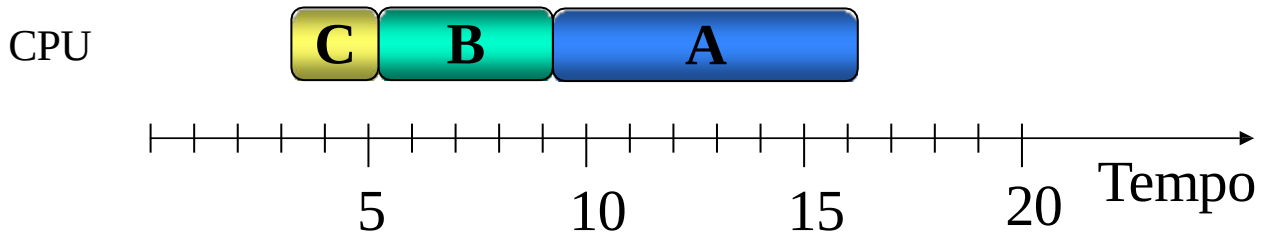
PUC Minas

Algoritmo SJF (*Shortest Job First*)

- Funcionamento:
  - O processador é alocado ao processo com etapa de CPU mais breve.
  - Em caso de empate se aplica outro algoritmo (normalmente o FIFO).
  - Não preemptivo
    - O processo que possui a CPU somente a libera quando quando termina sua execução ou quando se bloqueia
  - Com preempção
    - Se um outro processo chegar pico de CPU menor do que o restante do processo atual, há preempção. Esse esquema é conhecido como "*Shortest Remaining Time First*" (SRTF).

- Implementação:
  - Ordena a fila de processos prontos em função do tempo das seguintes etapas de CPU dos processos.

Diagrama de Gant SJF



$$Util._{cpu} = \frac{TCPU_{ocupada}}{Tempo} = \frac{13}{16} = 0{,}8125 = 81{,}25\%$$

$$\Pr odutividade = \frac{n^o\ processos}{tempo} = \frac{3}{16} = 0{,}17$$

$$TEspera_{medio} = \frac{TEspera_A + TEspera_B + TEspera_C}{n^o\ procesos} = \frac{(9-0)+(5-2)+(3-3)}{3} = \frac{9+3+0}{3} = 4$$

$$T\,\mathrm{Re}torno_{medio} = \frac{T\,\mathrm{Re}torno_A + T\,\mathrm{Re}torno_B + T\,\mathrm{Re}torno_C}{n^o\ procesos} = \frac{16+9+5}{3} = 10$$

PUC Minas

Características SJF

Reduz o tempo de espera médio

Minimiza o efeito de priorizar processos do tipo *cpu-bound*

É difícil determinar a priori qual será a duração da seguinte etapa de CPU dos processos.

Preemptivos

PUC Minas

Algoritmo Por Prioridades

PUC Minas

Algoritmo por prioridades

- **Funcionamento:**
  - Cada processo tem associado um valor inteiro que representa sua prioridade de execução
  - O escalonador escolhe o processo da fila de processos prontos que tenha a maior prioridade.

- **Implementação:**
  - A fila de processos prontos é ordenada pela prioridade dos processos.

- **Opções:**
  - A Política pode ser preemptiva ou não.
  - As prioridades podem ser definidas de forma interna (pelo SO) ou de forma externa (pelo usuário).
  - Prioridades estáticas ou dinâmicas.

PUC Minas

Algoritmo *Round Robin*

PUC Minas

Algoritmo ***Round-Robin***(turno rotativo)

- Atribui-se a cada processo durante um intervalo de tempo um valor pré fixado de forma rotativa, denominado *quantum*.
- Funcionamento:
  - Semelhante ao FCFS
  - Fila de prontos é uma fila FIFO circular
  - Escalonador percorre fila alocando, para cada processo, até 1 quantum
- Implementação:
  - Neste algorimo é requerido um valor temporal de troca de contexto.
- Características:
  - Permite esgotar ao máximo o tempo de resposta dos processos.
  - Algoritmo ideal para sistemas de tempo compartilhado.

Algoritmo ***Round-Robin***(turno rotativo)

- Se processo não deixar a CPU dentro do quantum, é preemptado
- Se houverem n processos e o quantum for q, cada processo possui 1/n tempo de CPU, executado em porções de tempo de tamanho até q
- Nenhum processo espera mais do que (n-1)q para utilizar CPU
  - Não ocorre **starvation (estagnação)**
- Desempenho
  - Quantum muito grande: execução FCFS (FIFO)
  - Quantum muito pequeno: muitas trocas de contexto
    - Alto custo
  - Quantum deve ser pequeno suficiente para garantir o tempo compartilhado
  - Quantum deve ser grande bastante para compensar trocas de contexto
  - Bom desempenho: 80% dos picos de CPU devem ser menores que quantum

Filas Multiníveis

PUC Minas

Filas Multiníveis

- Fila de prontos é dividida em várias filas
  - Ex.: 2 filas
    - Processos em primeiro plano (interativos/foreground);
    - Processos em segundo plano (background/batch);
- Cada fila possui seu próprio algoritmo de escalonamento:
  - Ex.:
    - Processos em primeiro plano: RR (para manter tempo compartilhado);
    - Processos em background: FCFS;
- É necessário haver escalonamento entre as filas:
  - Para escolher o processo de qual fila será executado;
  - Se usar algoritmo de prioridade fixa de uma fila sobre outra: starvation;
  - Outra opção: dividir o tempo de execução entre as filas:
  - Foreground fica com 80% e background com 20% do tempo de CPU.

Filas Multiníveis com Retroalimentação

- Sem retroalimentação: processo nunca é trocado de fila;
- Com retroalimentação: processo pode ser trocado de fila;
  - Permite separar processos com características de picos de CPU semelhantes;
    - Um processo que usa muito tempo de CPU é movido para fila de mais baixa prioridade;
    - Dessa forma: processos IO-bound e interativos (dependem da interação do usuário) ficam nas filas com mais prioridade;
    - Processos que ficam aguardando muito tempo por CPU podem ser movidos para filas de mais alta prioridade: evita starvation (estagnação)

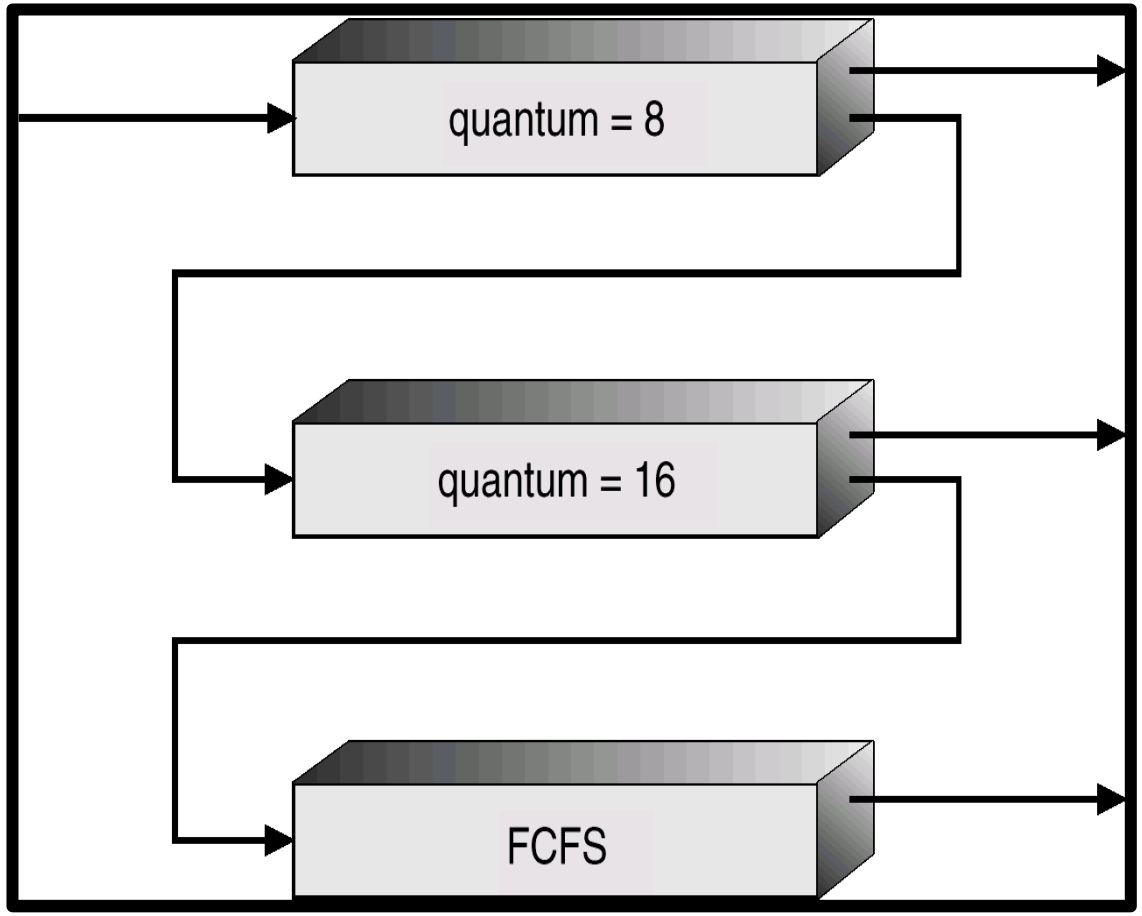
Filas Multiníveis com Retroalimentação

- Escalonador é definido pelos seguintes parâmetros:
  - número de filas;
  - algoritmos de escalonamento para cada fila;
  - método usado para determinar quando elevar um processo;
  - método usado para determinar quando rebaixar um processo;
  - método usado para determinar em que fila um processo entrará quando esse processo precisar de atendimento;

Filas Multiníveis com Retroalimentação

- EX:
- Três filas (com prioridade fixa):
  - Q0 – quantum de tempo 8 milissegundos
  - Q1 – quantum de tempo 16 milissegundos
  - Q2 – FCFS
- Escalonamento
  - Uma nova tarefa entra na fila Q0 , que é atendida com base no RR. Quando ganha a CPU, a tarefa recebe 8 milissegundos. Se não terminar nesse tempo, a tarefa é movida para a fila Q1.
  - Em Q1, a tarefa é atendida novamente com base no RR e recebe 16 milissegundos adicionais. Se ainda não estiver completa, a tarefa é apropriada e movida para a fila Q2.

Filas Multiníveis com Retroalimentação



PUC Minas

Escalonamento em Tempo Real

PUC Minas

Escalonamento

- Escalonador de TEMPO REAL
  - Tipos de aplicações
    - Industriais
    - Automóveis
    - Multimídia
  - Tipos de sistemas tempo real
    - Sistemas críticos (*Hard Real-Time*)
    - Sistemas não críticos (*Soft Real-Time*)

Escalonamento

- Escalonador de TEMPO REAL
  - Sistemas críticos (*Hard Real-Time*)
    - É necessário garantir que a(s) tarefa(s) consideradas críticas terminem antes de um determinado tempo (*deadline*), caso contrário o seu não cumprimento pode resultar em graves danos para o sistema.
    - Exemplos:
      - Aplicações aeroespaciais
      - ABS de um carro
      - Sistema de automação

PUC Minas

Escalonamento

- Escalonador de TEMPO REAL
  - Sistemas não críticos (*Soft Real-Time*)
    - O funcionamento do sistema é apenas ligeiramente afetado caso não seja possível cumprir um determinado *deadline*.
    - Exemplos:
      - Aplicações multimídia
      - Jogos de computador