Escalonamento de tarefas

Sistemas Operacionais
Gerência de processos

Agenda

* Revisão
* Escalonamento
  * Introdução
  * Tipos e níveis de escalonamento
  * Algoritmos de escalonamento
  * Critérios de projeto de algoritmos
* Escalonamento de threads em java

Multiprogramação

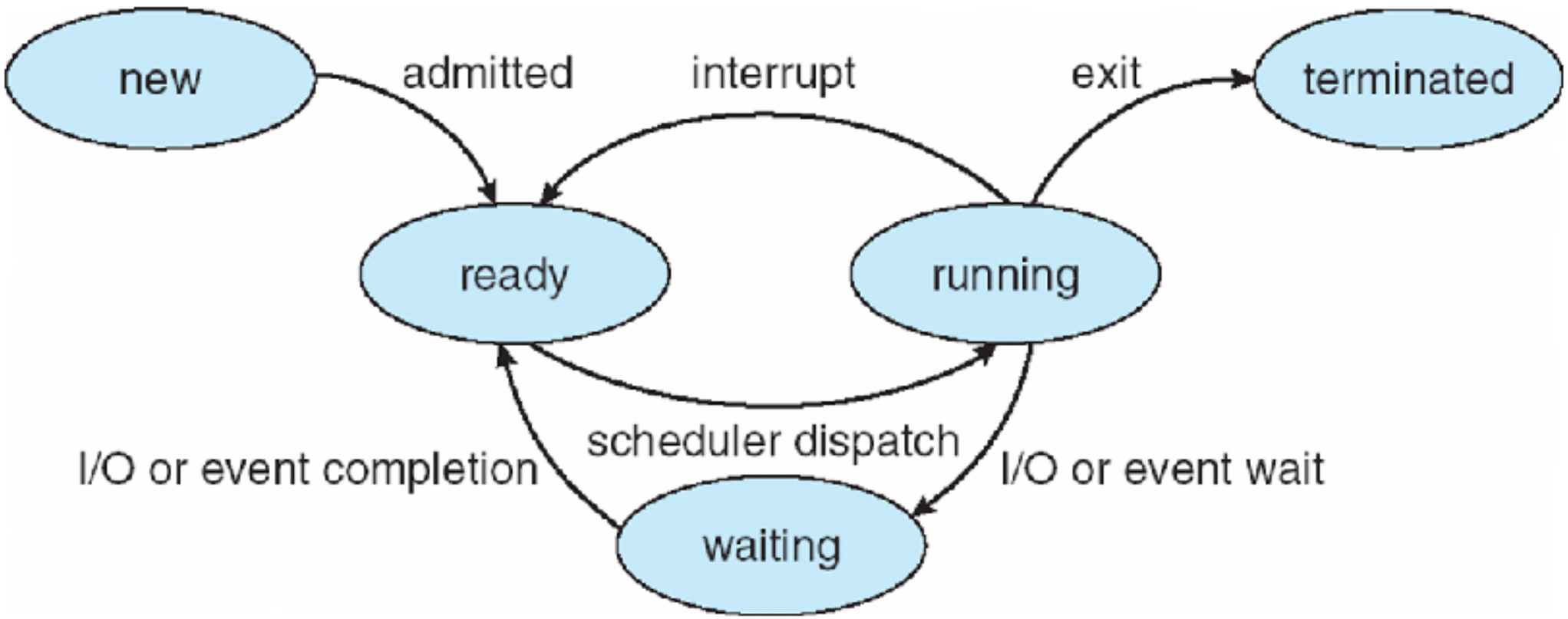
* Multiprogramação pressupõe a existência simultânea de vários processos disputando o processador
* Necessidade de "intermediar" esta disputa de "forma justa"
  * Intermediar
    * Gerênciar o processador
    * Algoritmos de escalonamento
  * Forma justa
    * Critérios do projeto do algoritmo de escalonamento

Representação de processo

* PCB Process Control Block
* Bloco de processo descritor ou descritor de processo

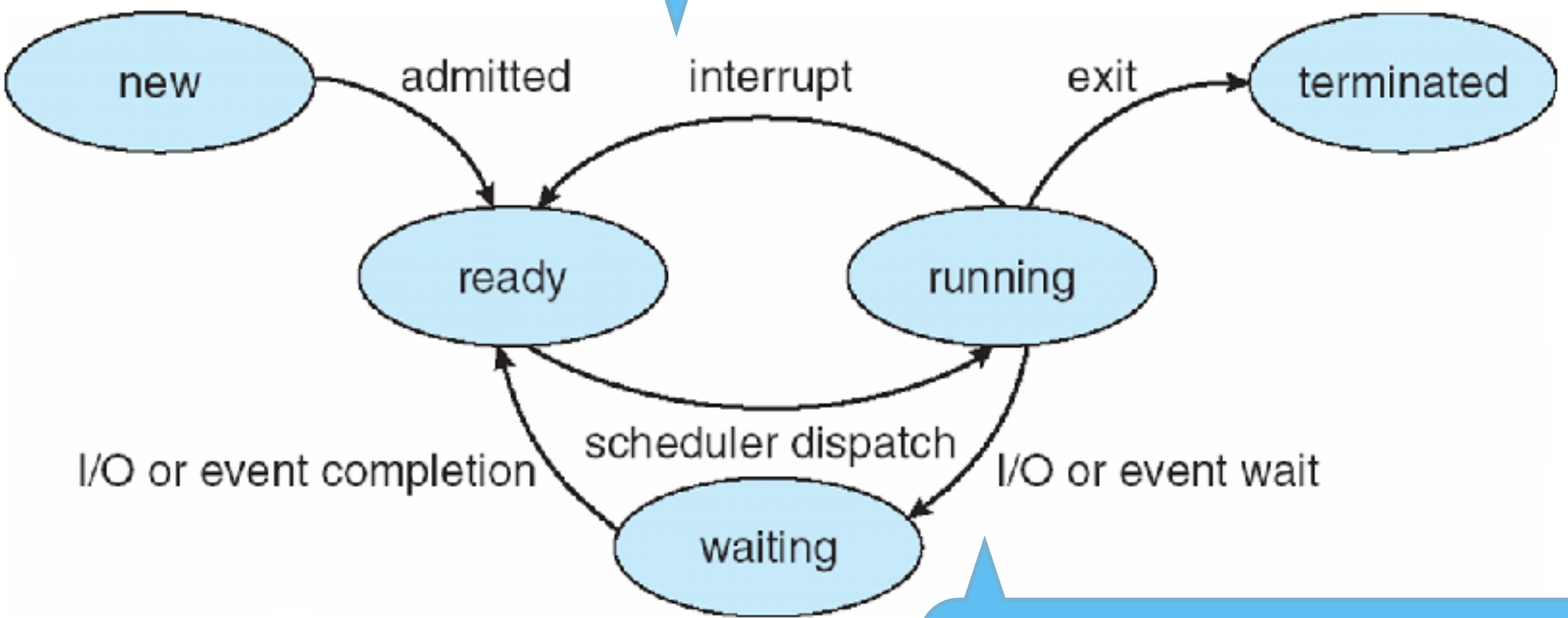
```
struct desc_proc{
        char     estado_atual;
        int                prioridade;
        unsigned inicio_memoria;
        unsigned tamanho_mem;
        struct   arquivos arquivos_abertos[20];
        unsigned tempo_cpu;
        unsigned proc_pc;
        unsigned proc_sp;
        unsigned proc_acc;
        unsigned proc_rx;
        struct   desc_proc *proximo;
}
```

Estados de um processo



Estados de um processo

Quantos processos em cada estado?



Como organizar esses processos?

Agenda

* Revisão
* Escalonamento
  * Introdução
  * Tipos e níveis de escalonamento
  * Algoritmos de escalonamento
  * Critérios de projeto de algoritmos
* Escalonamento de threads em java

Escalonamento

* O escalonador é a entidade do SO responsável por selecionar um processo pronto para executar em um processador
* O objetivo é dividir o tempo do processador de "forma justa" entre os processos de estato "pronto" (aptos a executar)
  * Típico de sistemas multiprogramados : batch, time-sharing, multiprogramado ou tempo real
  * Cada tipo tem seus requisitos e restrições (critérios) diferentes em relação a utilização da CPU

Escalonamento

* O escalonamento é dividido em duas partes:
  1. Dispatcher (Despachador ou despachante)
     * Responsável por efetuar a troca de contexto
  2. Escalonador
     * Define a política de seleção
     * Usualmente atráves de uma algoritmo

Despachador (Dispatcher)

* Objetivo
  * Fornece o controle da CPU ao processo selecionado pelo escalonador

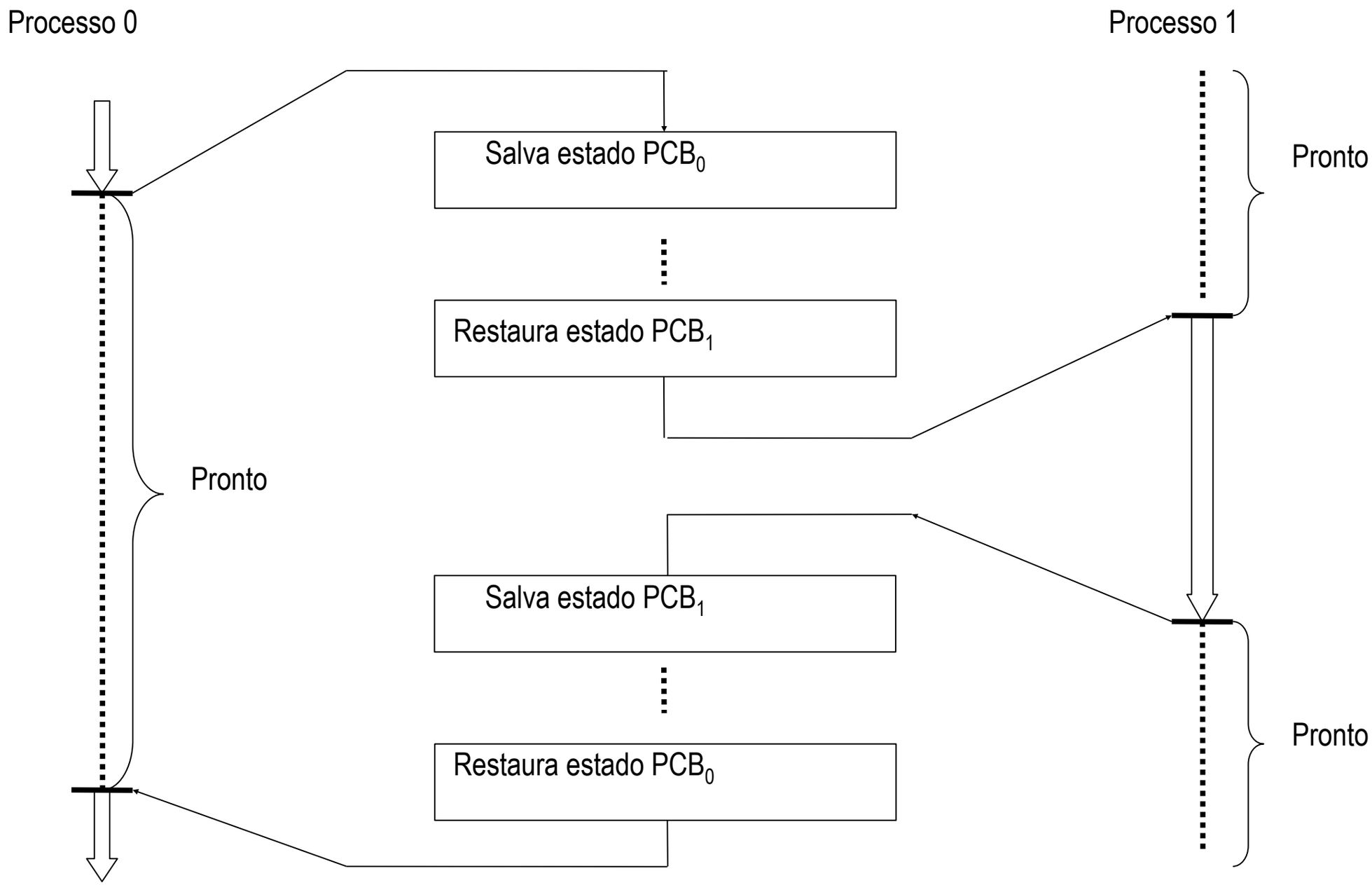
Despachador

* Tarefas do despachador :
  * Alterna para modo supervisor
  * Executa a troca de contexto
  * Salta para o local apropriado do PC
  * Alterna troca para o modo usuário
  * Reinicia a execução

Latência de despacho

* Tempo para que o despachante pare a execução de um processo e inicie a execução de outro

Despachador



Escalonador

* Objetivo
  * Seleciona dentre os processos na memória que estejam prontos para executar, e aloca a CPU a um deles

Situações típicas para escalonar

* São situações típicas
  * Processos que alternam seu estado
  * Sempre que a CPU estiver livre
* Eventos externos ao processo
  * Interrupção de relógio (clock)
  * Interrupção de dispositivos de E/S
* Eventos internos
  * Interrupção por erros

Situações típicas para escalonar

* De uma forma geral, por alternância de estado de um processo

A. Passa do estado executando para esperando

B. Passa do estado executando para pronto

C. Passa de esperando para pronto

D. Termina

Agenda

* Revisão
* Escalonamento
  * Introdução
  * Tipos e níveis de escalonamento
  * Algoritmos de escalonamento
  * Critérios de projeto de algoritmos
* Escalonamento de threads em java

Tipos de escalonadores

* 2 tipos
  * preemptivo
    * SO pode interromper um processo em execução
* não preemptivo ou cooperativo
  * Somente o processo pode "deixar" o processador

Tipos de escalonadores

	Preemptivo	Não preemptivo
Término de execução	XX	XX
Requisição dispositivo de E/S	XX	XX
Liberação voluntária	XX	XX
Interrupção de relógio	XX	
Processo de mais alta prioridade em estado de pronto	XX	

Níveis de escalonadores

- Longo prazo
  - Executado quando um novo processo é criado
  - Determina quando um processo novo passa a ser considerado no sistema, isto é, quando após sua criação ele passa a ser apto
  - Controla o grau de multiprogramação do sistema
    - Quanto maior o número de processos ativos, menor a porcentagem de tempo de uso do processador por processo
- Médio prazo
- Curto prazo

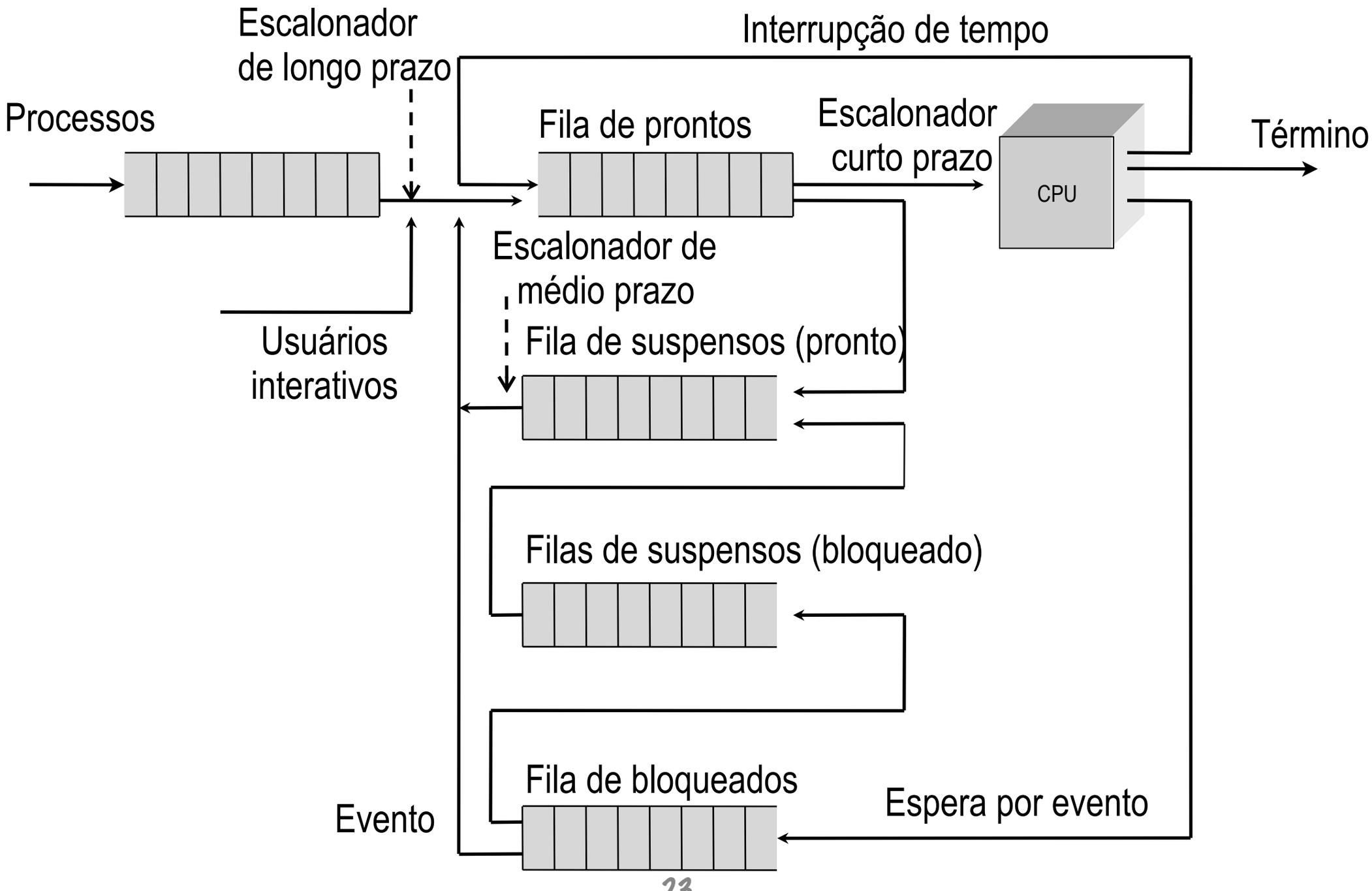
Níveis de escalonadores

* Longo prazo
* Médio prazo
    * Associado a gerência de memória
        * Participa do mecanismo de swapping
    * Suporte adicional a multiprogramação
        * Grau de multiprogramação efetiva (diferencia prontos dos prontos-suspensos)
* Curto prazo

Níveis de escalonadores

* Longo prazo
* Médio prazo
* Curto prazo
  * Determina qual processo apto deverá utilizar o processador
  * Executado sempre que ocorre eventos importantes:
    * Interrupção de relógio, interrupção de E/S, chamadas de sistemas, sinais (interrupção software)

Diagrama de níveis de escalonamento



Agenda

* Revisão
* Escalonamento
  * Introdução
  * Tipos e níveis de escalonamento
  * Algoritmos de escalonamento
  * Critérios de projeto de algoritmos
* Escalonamento de threads em java

Algoritmos de escalonamento

* First-In First-Out (FIFO) ou First-Come First-Served (FCFS)
* Shortest Job First (SJF) ou Shortest Process Next (SPN)
* Round robin (circular)
* Baseado em prioridades
* Múltiplas filas
* High Response Ratio Next (HRRN)
* Shortest Remaining Time (SRT)

Classificação de Algoritmos

* Algoritmos não preemptivos (cooperativos)
  * First-In First-Out (FIFO) ou First-Come First-Served (FCFS)
  * Shortest Job First (SJF) ou Shortest Process Next (SPN)
* Algoritmos preemptivos
  * Round robin (circular)
  * Baseado em prioridades
  * Múltiplas filas
  * High Response Ratio Next (HRRN)
  * Shortest Remaining Time (SRT)

R. S. Oliveira, A. S. Carissimi, S. S. Toscani. Sistemas Operacionais. Ed Bookman (Série livros didáticos de informática da UFRGS), 4a Ed.

First-In First-Out

* Funcionamento:
    1. Processos modificam seu estado para pronto
    2. Processos são inseridos no final da fila
    3. Processo que está no início da fila é o próximo a executar
    4. Processo executa até que:
        4.1. Libere explicitamente o processador
        4.2. Realize uma chamada de sistema
        4.3. Termine sua execução

First-In First-Out

* Características
  * Implementação simplificada
  * É utilizado a estrutura de dados de fila para organizar os processos em estado de pronto
  * Algoritmo (na sua definição original) cooperativo
  * Prejudica processos E/S (I/O bound)

Shortest job first

* Características
  * Associe a cada processo a extensão de seu próximo tempo de CPU
  * Use essas extensões para escalonar o processo com o menor tempo
  * Processos E/S são favorecidos
  * Porém, difícil o cálculo de tempo do ciclo de CPU para os processos

Shortest job first

* 2 esquemas (preemptivo e cooperativo)
  * Cooperativo (original) - uma vez a CPU dada ao processo, ele não pode ser apropriado até que termine seu uso de CPU
  * Preemptivo (SRTF) - se um novo processo chega com tamanho de uso de CPU menor que o tempo restante do processo atualmente em execução, apropria
    * Algoritmo Shortest-Remaining-Time-First (SRTF)

Round robin

* Características
  * Similar ao algoritmo FIFO, uso de fila
  * Cada processo recebe um tempo limite máximo (time-slice, quantum) para executar um ciclo de CPU
* A fila de processos em estado de pronto é uma fila circular
* Necessidade de um relógio para delimitar as fatias de tempo (ciclo de processamento)
  * Interrupção de relógio do hardware

Round robin

* Características
  * Por ser preemptivo, um processo "perde" CPU

    (A)Libera explicitamente o processador

    (B)Realize uma requisição a dispositivos de E/S

    (C)Termina sua execução

    (D)Quando sua fatia de tempo é esgotada

* Se quantum $\rightarrow \infty$ obtém-se o comportamento do algoritmo FIFO

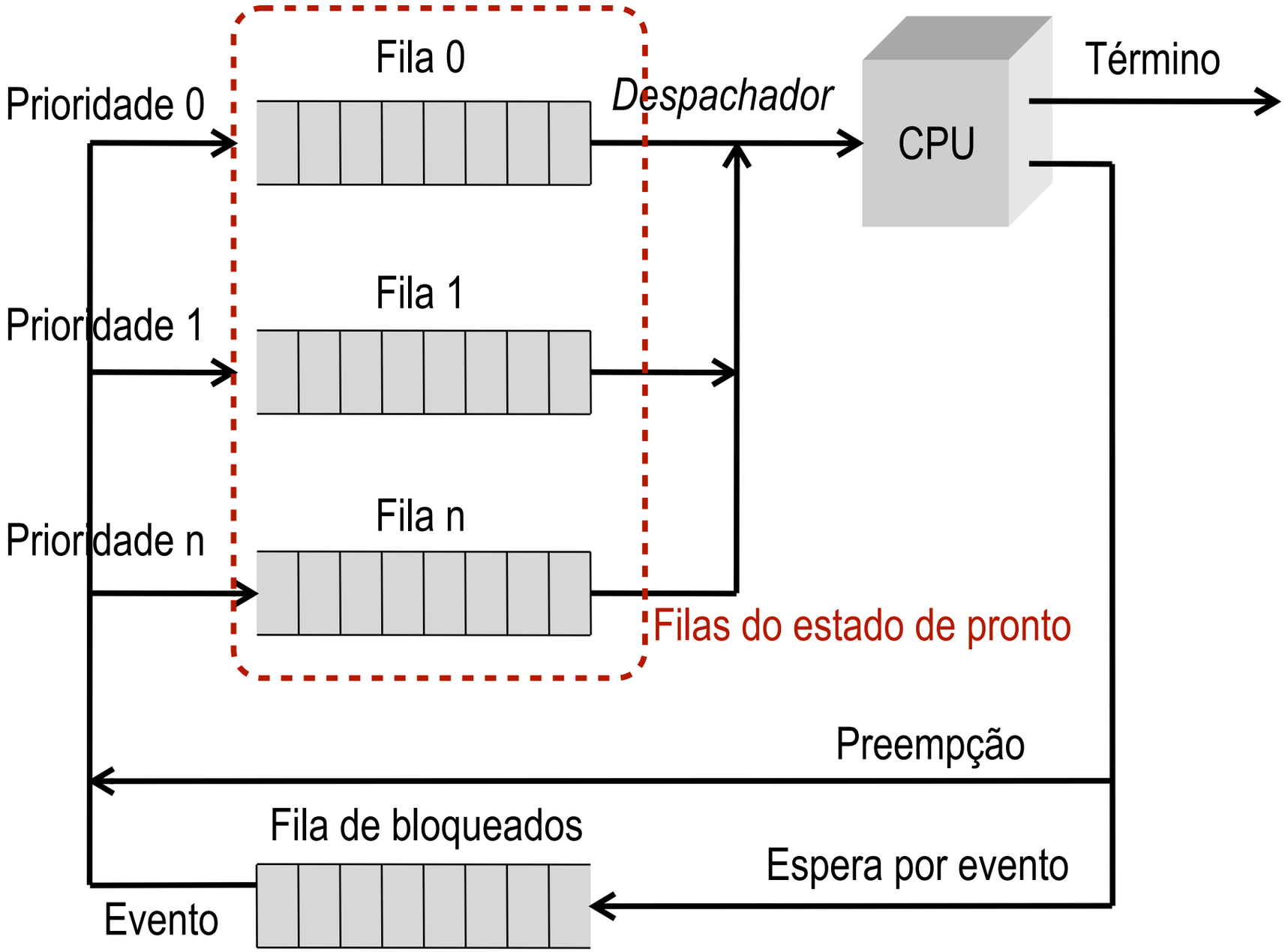
Round robin Problemas

* Problema 1 : dimensionamento do quantum
  * Compromisso entre sobrecarga (latência) e tempo de resposta em função do número de usuários (processos)
  * Compromisso entre tempo de troca de contexto (latência) e tempo do ciclo de processador (quantum)
* Problema 2 : processos E/S são prejudicados
  * Esperam da mesma forma que processos CPU, porém muito provavelmente não utilizam todo o seu quantum
  * Solução : uso de prioridade
    * Associar prioridades mais altas aos processos E/S para compensar o tempo gasto no estado de espera (pronto)

Prioridade

* Objtetivo
  * Sempre que um processo de maior prioridade que o processo atualmente em execução entrar no estado pronto deve ocorrer uma preempção
* Características
  * A existência de prioridades pressupõem a preempção
  * Escalonador deve sempre selecionar o processo de mais alta prioridade segundo uma política:
    * Round-Robin, FIFO, SJF

Prioridade com múltiplas filas



Prioridades Problemas

* Um processo de baixa prioridade pode não ser executado
  * Postergação de execução indefinida (starvation)
* Processo com prioridade estática pode ficar mal classificado e ser penalizado ou favorecido em relação aos demais
  * Típico de processos que durante sua execução trocam de padrão de comportamento (CPU a E/S e vice-versa)
* Solução:
  * Múltiplas filas com realimentação e prioridade dinâmica

Prioridades Problemas

* A mudança de prioridade pode ser considerada
* Em função do tempo de uso da CPU a prioridade do processo aumenta e diminui
* Sistema de envelhecimento (agging) evita postergação indefinida

Agenda

* Revisão
* Escalonamento
  * Introdução
  * Tipos e níveis de escalonamento
  * Algoritmos de escalonamento
  * Critérios de projeto de algoritmos
* Escalonamento de threads em java

Alguns critérios de escalonamento

* Maximizar a utilização do processador
* Maximizar a produção do sistema (throughput)
  * Número de processos executados por unidade de tempo
* Minimizar o tempo de execução (turnaround)
  * Tempo total para executar um determinado processo
* Minimizar o tempo de espera
  * Tempo que um processo permanece na lista de pronto
* Minimizar o tempo de resposta
  * Tempo decorrido entre uma requisição e a sua realização

Critérios de otimização de escalonamento

* Estatística sobre processos e SO
  * Utilização máxima de CPU
  * Throughput máximo
  * Tempo de turnaround máximo
  * Tempo de espera mínimo
  * Tempo de resposta mínimo

Exemplos de critérios

* SO com múltiplos processadores
  * Compartilhamento de carga entre processadores
* Multiprocessamento assimétrico
  * Somente um processador acessa as estruturas de dados do sistema
  * Diminui a complexidade por não necessitar compartilhamento

Exemplos de critérios

* SO de tempo real
    * Tempo real rígido
        * Exigidos para completar uma tarefa crítica dentro de um período de tempo garantido
    * Tempo real flexível
        * Exige que processos críticos recebam prioridade em relação aos menos favorecidos

Agenda

* Revisão
* Escalonamento
  * Introdução
  * Tipos e níveis de escalonamento
  * Algoritmos de escalonamento
  * Critérios de projeto de algoritmos
* Escalonamento de threads em java

Escalonamento de thread em java

* Política de escalonamento baseada em prioridades
* Algoritmo de escalonador implementado pela MV
  * Algumas MV podem fornecer suporte à preempção
  * Relacionamento entre thread de nível de usuário e de kernel

Tempo de CPU

* Um thread executa (tem tempo de CPU) até:

(A)Seu quantum de tempo expirar

(B)Ele for bloqueado por uma solicitação de E/S

(C)Ele sair do seu método run()

Prioridade de Threads em Java

* Prioridade definida por inteiro, variando entre 1-10
* Constantes predefinidas
  * Thread.MIN_PRIORITY = 1
  * Thread. NORM_PRIORITY = 5
  * Thread. MAX_PRIORITY = 10

Códigos

Bibliografia

Escalonamento de tarefas
Gerência de processos
Sistemas operacionais

Bibliografia

* **The Java Tutorials: concurrency**. Disponível em http://docs.oracle.com/javase/tutorial/essential/concurrency/ (acessado em 18/01/2013)
* SILBERSCHATZ, G.; GAGNE, G. Sistemas Operacionais com Java. Campus, 7a Ed, 2007.
* R. S. Oliveira, A. S. Carissimi, S. S. Toscani. Sistemas Operacionais. Ed Bookman (Série livros didáticos de informática da UFRGS), 4a Ed.

Escalonamento de tarefas

Sistemas Operacionais
Gerência de processos