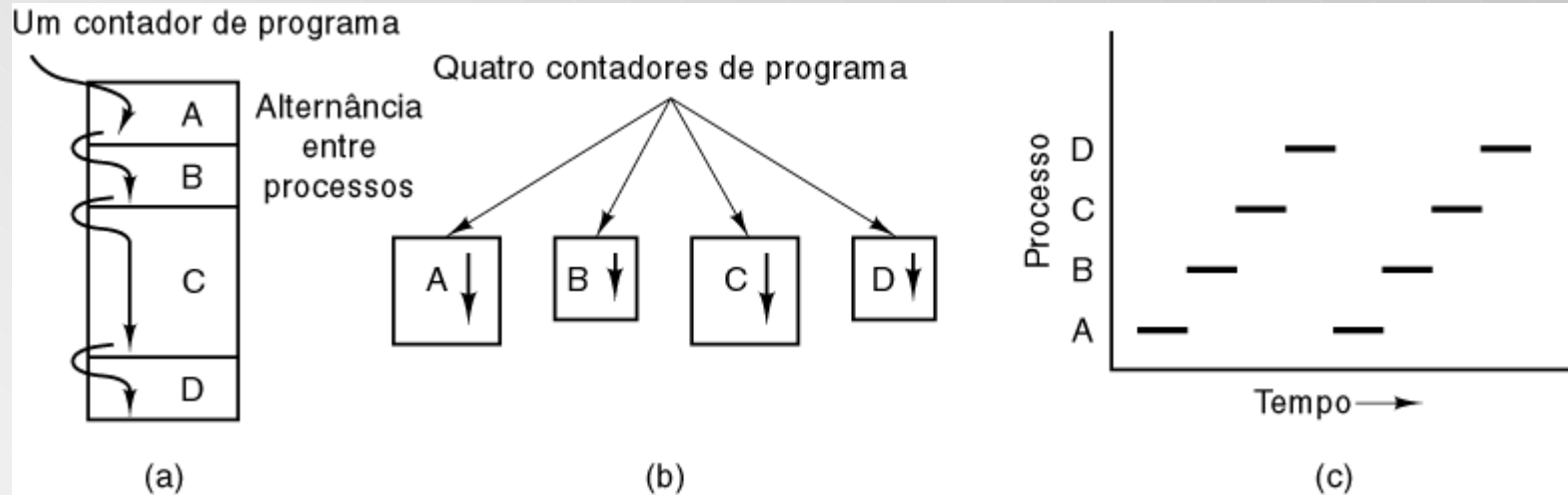


Infra-Estrutura de Software

Escalonamento

Processos Concorrentes

O Modelo de Multiprogramação



- Multiprogramação de quatro programas
- Modelo conceitual de 4 processos sequenciais, independentes
- Somente um programa está ativo a cada momento

Conceitos Básicos

Tipos de S.O.

- Monotarefa
- Multitarefa
- Monousuário
- Multiusuário

Como evitar que um processo monopolize o sistema?

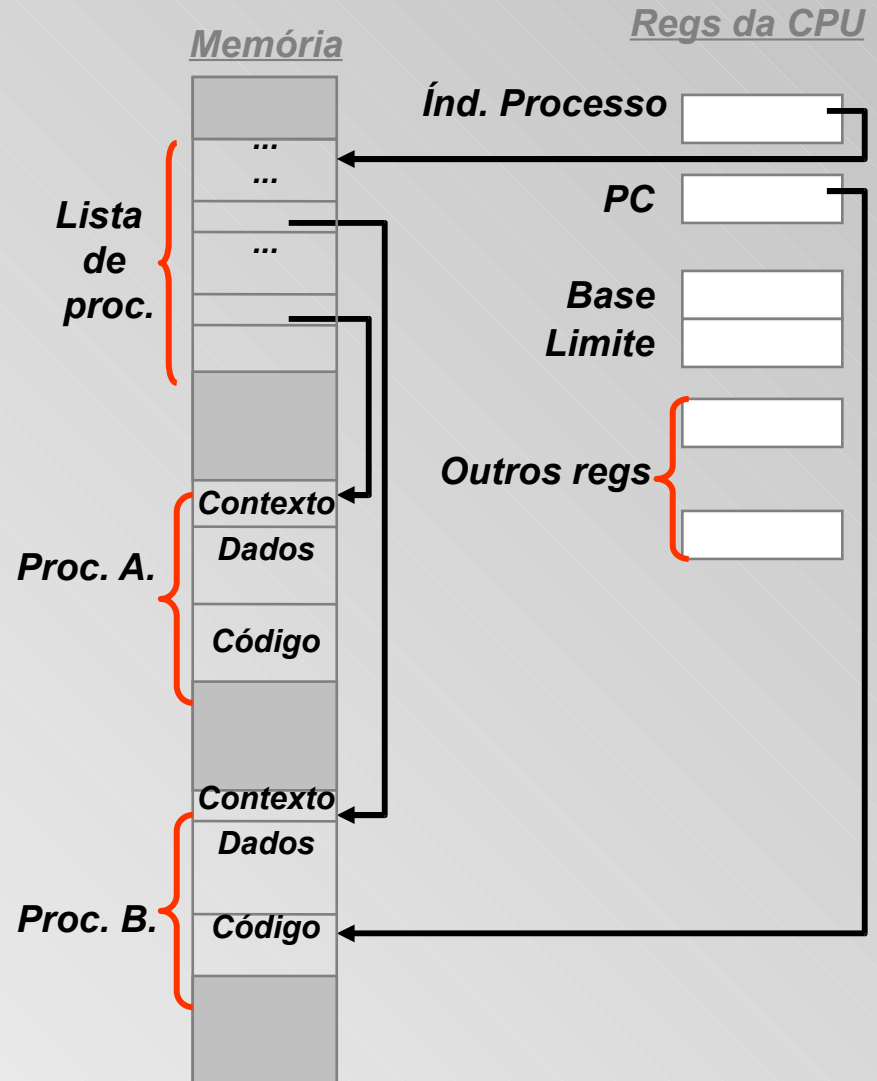
Sistemas de tempo compartilhado (*Time Sharing Systems*)

- Permite sistemas interativos (entrada/saída)
- Requer temporizadores (timers)
- Interrupções

Conceitos Básicos: Tipos de S.O.

Multiprocessamento

- O índice do processo contém o apontador para a lista de processos
- PC (Program Counter) = contador de programas
- Uma troca de processos consiste em trocar o valor dos registradores de contexto da CPU

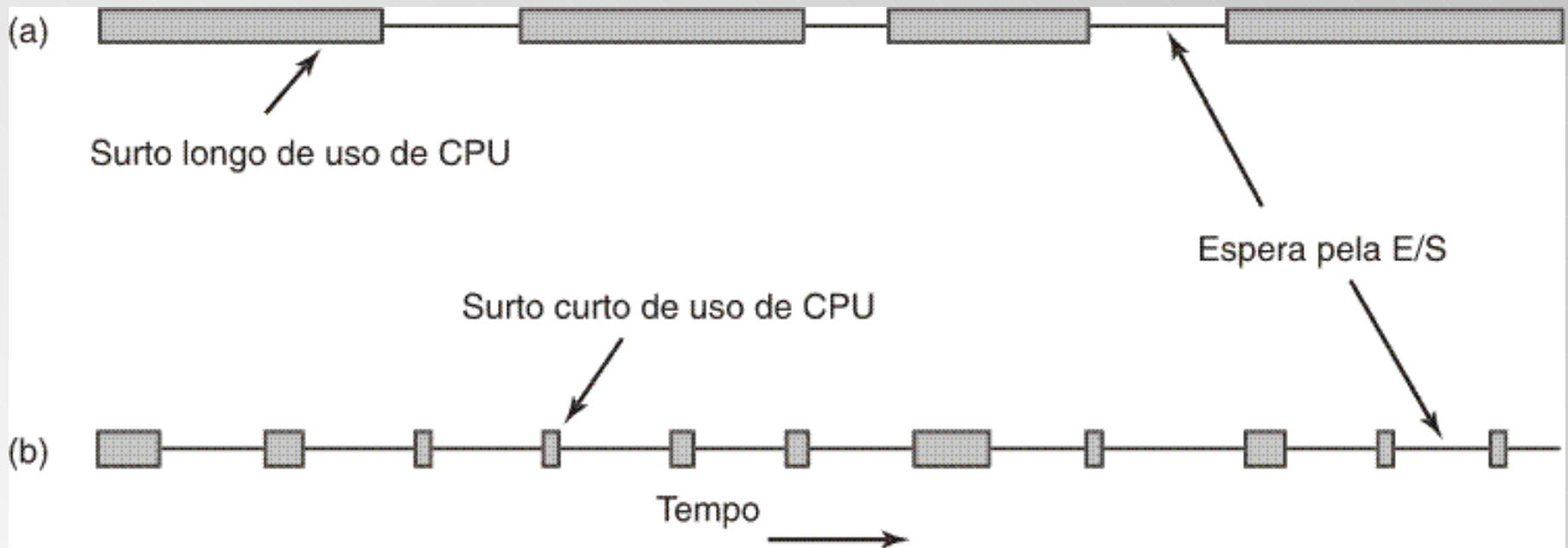


O que é necessário para haver multiprocessamento?

Suporte do Hardware

- Temporizadores (*timers*)
 - Interrupções
 - Proteção de memória
- Suporte do S.O.
 - Escalonamento dos processos
 - Alocação de memória
 - Gerenciamento dos periféricos

Comportamentos de Processos



- Surtos de uso da CPU alternam-se com períodos de espera por E/S
 - um processo limitado por CPU
 - um processo limitado por E/S

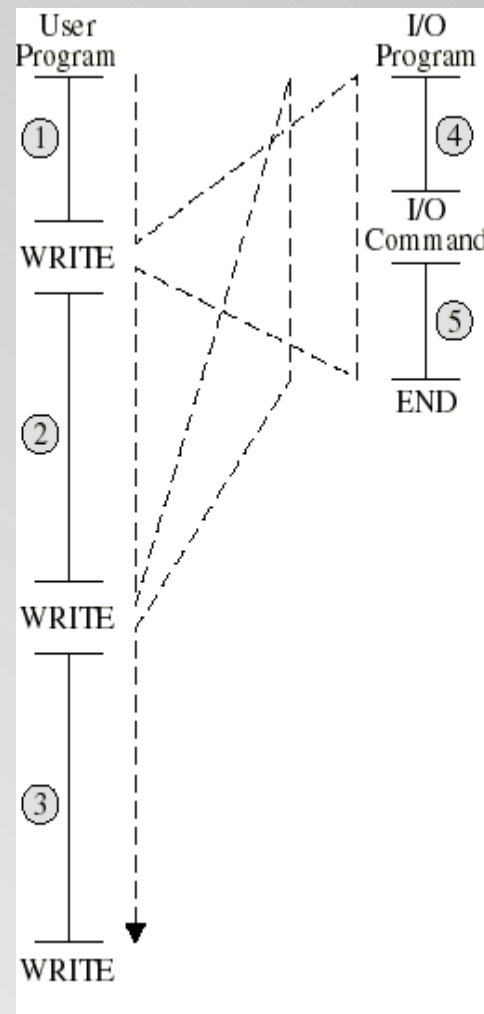
Tipos de Processo

- *CPU-bound*:
 - gasta a maior parte do seu tempo usando a CPU
 - longos tempos de execução e baixo volume de comunicação entre processos
 - ex: aplicações científicas, engenharia e outras aplicações que demandam alto desempenho de computação
- *I/O-bound*:
 - passa a maior parte do tempo esperando por dispositivos de E/S
- processos I/O-bound devem ter prioridade sobre processos CPU-bound
- Batch (lote) x Interativos

A importância da Interrupção

- Num sistema simples, CPU deve **esperar a execução do comando de E/S**
 - A cada chamada do comando *write* a CPU fica esperando o dispositivo executar o comando.

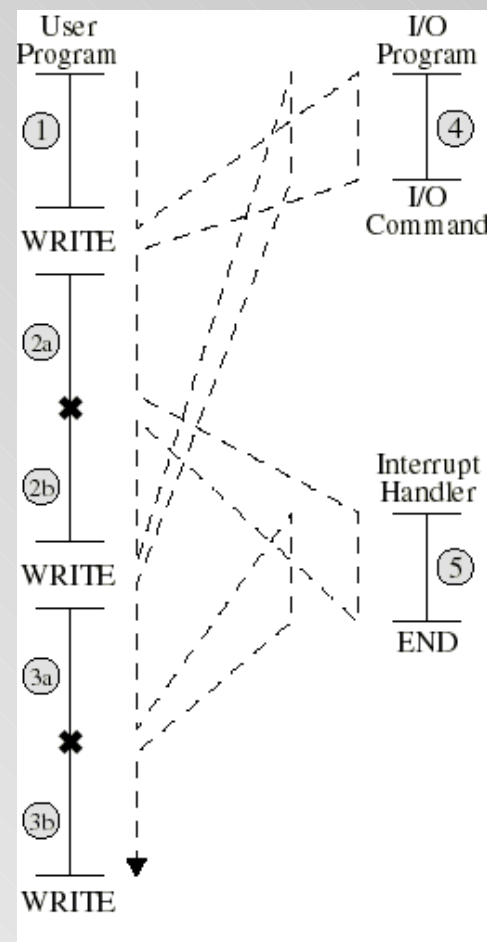
Ex: escrita em disco



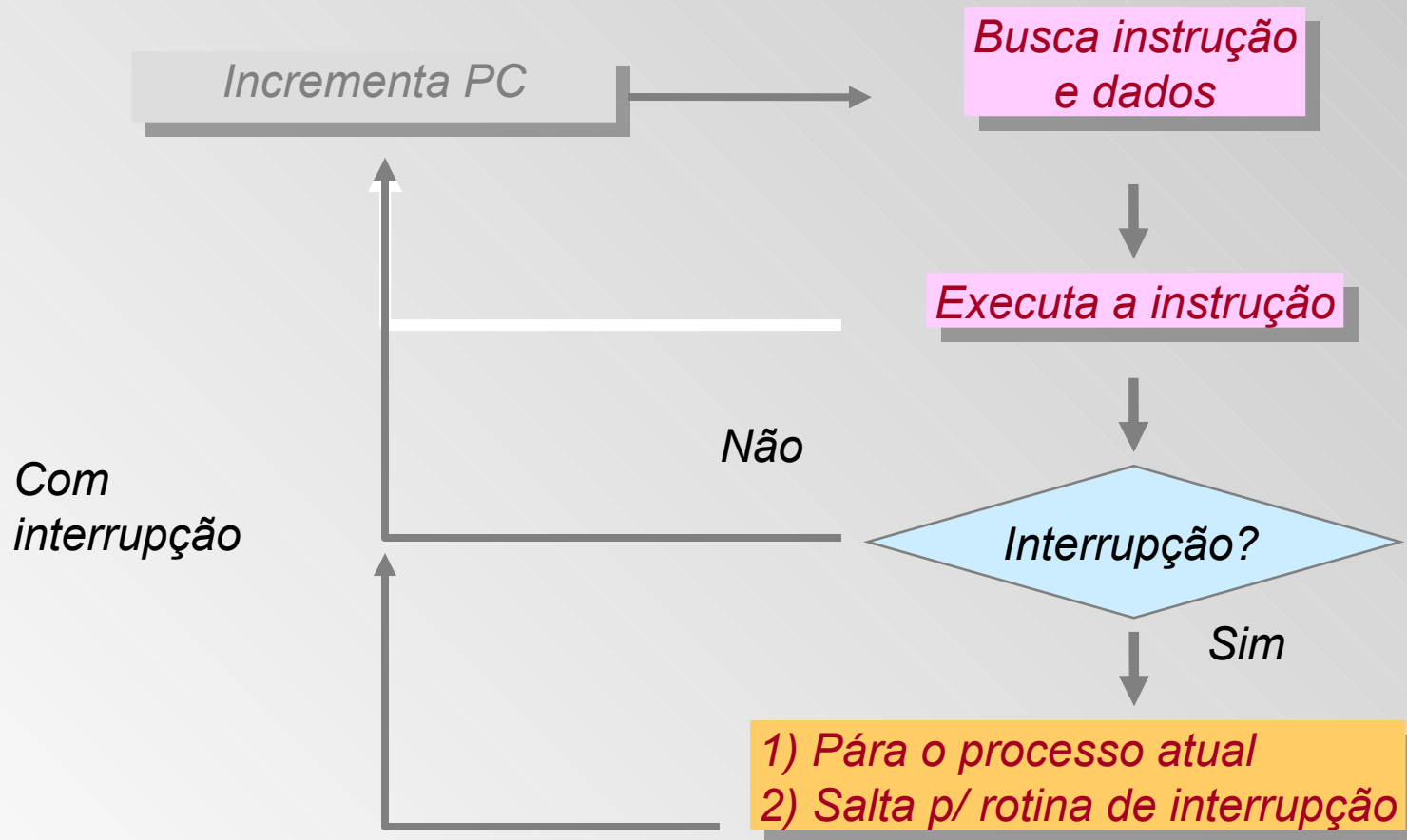
A importância da Interrupção

- Um sistema com interrupção **não fica esperando**
 - A CPU solicita o *write* e fica executando outras tarefas até ser interrompida pelo disco.

Ex: escrita em disco



Operação Básica da CPU



Interrupção do Programa

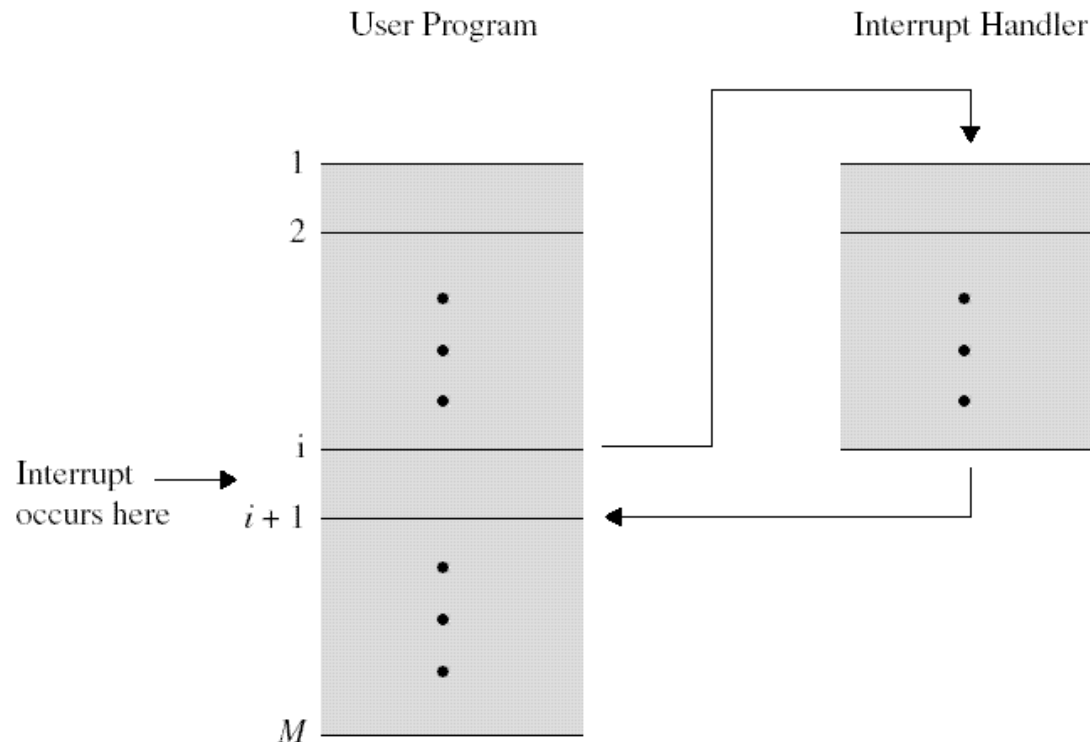
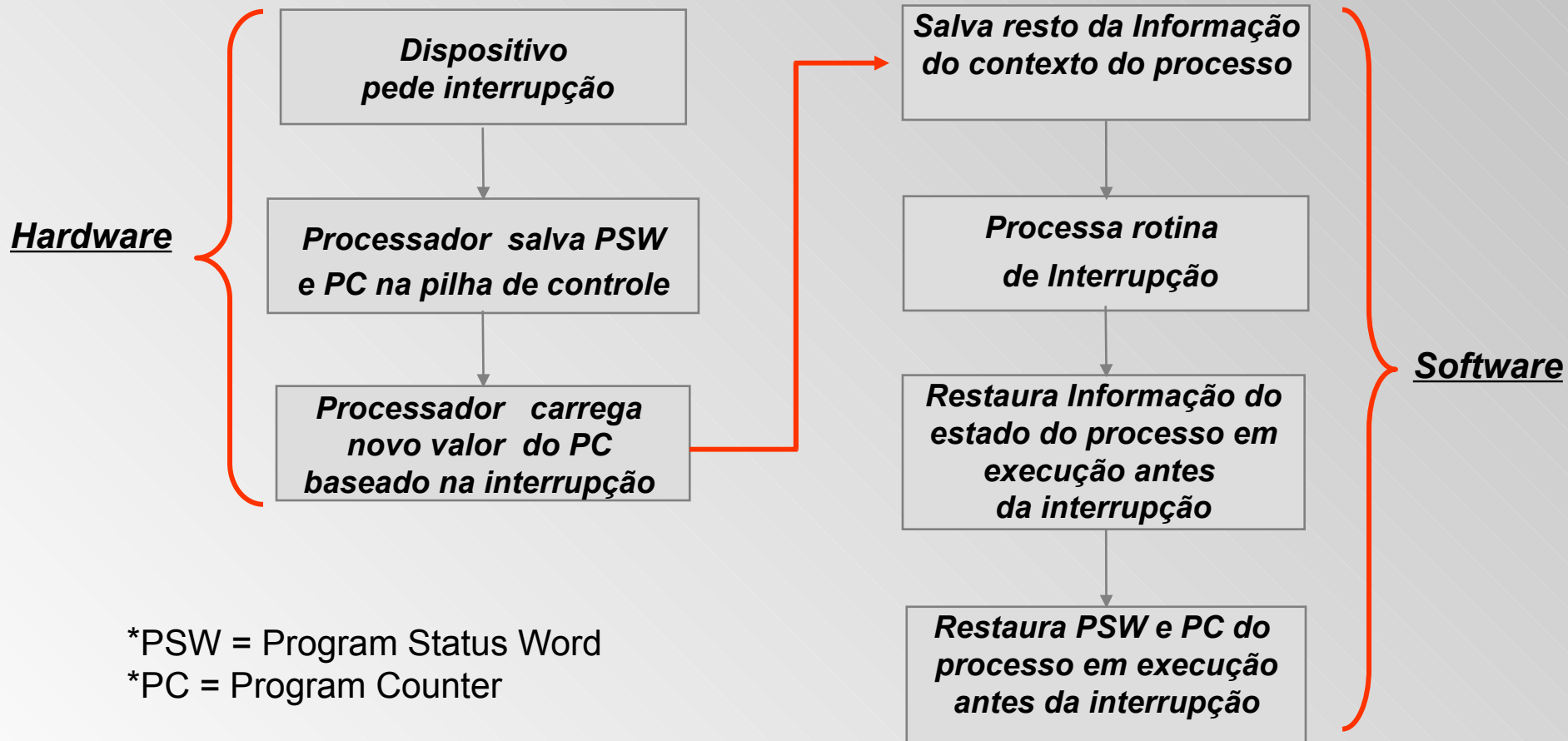


Figure 1.6 Transfer of Control via Interrupts

Processo de Interrupção



*PSW = Program Status Word

*PC = Program Counter

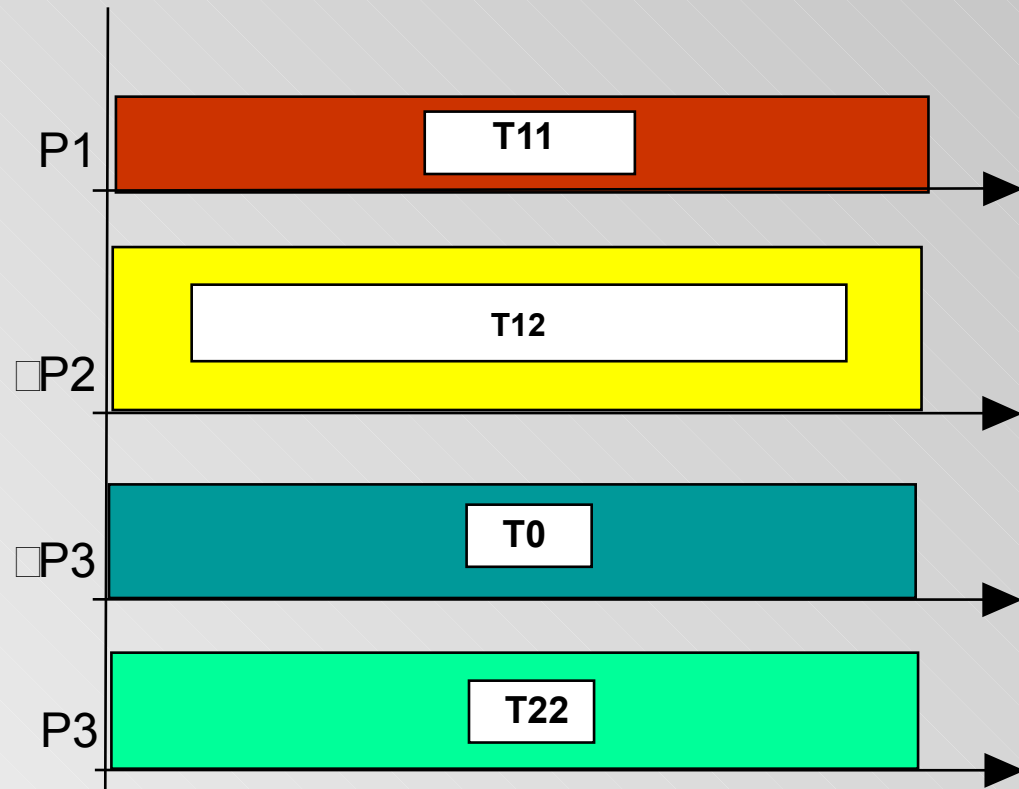
Algoritmos de Escalonamento

Escalonamento de processos

- Quando um ou mais processos estão prontos para ser executados, o SO decide qual executar primeiro
- A parte do sistema operacional responsável por essa decisão é chamada **escalador**
- O algoritmo usado para tal é chamado de **algoritmo de escalonamento**

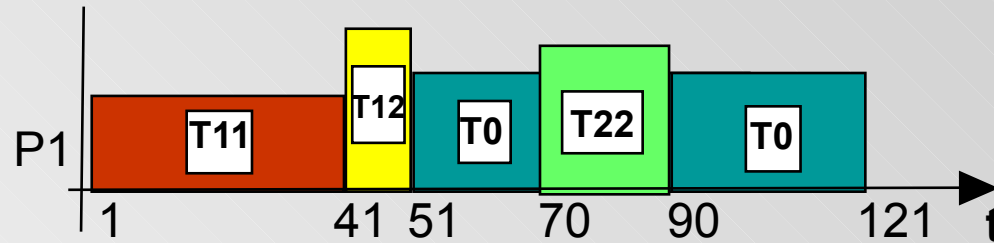
Abstração

- Uma máquina para cada processo
- Paralelismo real



Realidade

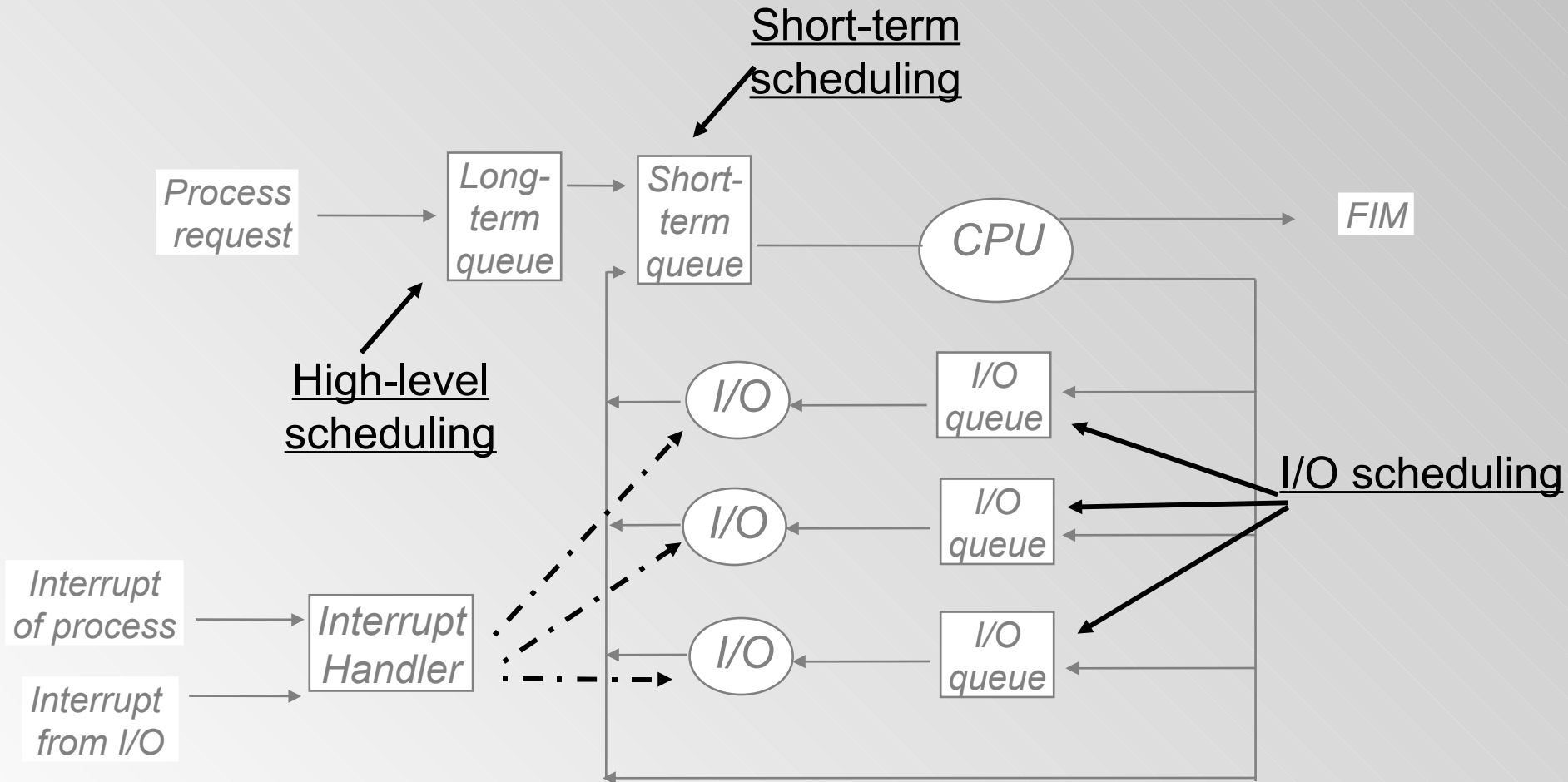
- Compartilhamento do tempo
- Pseudo-paralelismo



Filas de Escalonamento

- High-level
 - Decide quantos programas são admitidos no sistema
 - Aloca memória e cria um processo
- Short-term
 - Decide qual processo deve ser executado
- I/O
 - Decide qual processo (com I/O) pendente deve ser tratado pelo dispositivo de I/O

Filas de Escalonamento



Categorias de Algoritmos de Escalonamento

- Em lote (batch)
- Interativo
- Tempo-real

Introdução ao 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

Objetivos do algoritmo de escalonamento

Características de Escalonamento

- Justiça (fairness)
 - Todos os processos têm chances iguais de uso dos processador
- Eficiência
 - Taxa de ocupação do processador ao longo do tempo
- Tempo de Resposta
 - Tempo entre a ocorrência de um evento e o termino da ação correspondente
- *Turnaround* (tempo de retorno)
 - “Tempo de resposta” para usuários em batch
 - Minimizar o tempo que usuários batch devem esperar pelo resultado
- *Throughput* (vazão)
 - No. de “jobs” (processos) executados por unidade de tempo

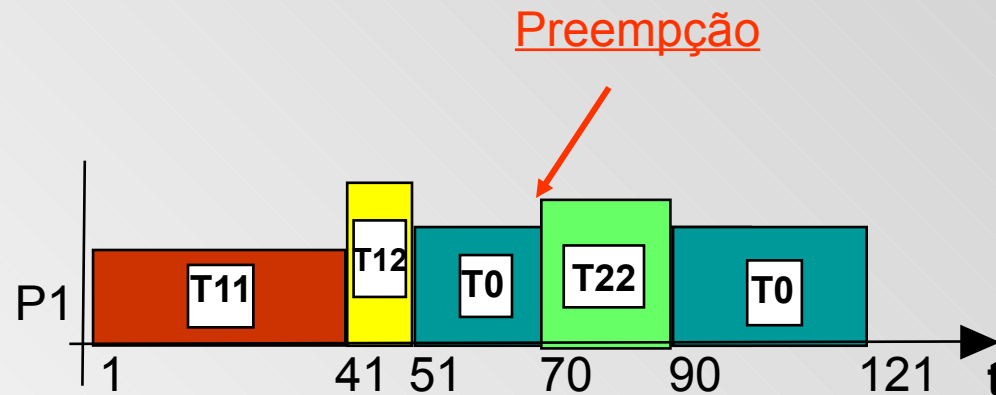
Tipos de Escalonamento

- Mecanismos de Escalonamento
 - Preemptivo x Não-preemptivo
- Políticas de Escalonamento
 - Round-Robin
 - FIFO (First-In First-Out)
 - Híbridos
 - Partições de Lote (Batch)
 - MFQ - Multiple Feedback Queue
 - SJF – *Shortest Job First*
 - SRJN – *Shortest Remaining Job Next*

Diz-se que um algoritmo/sistema operacional é **preemptivo** quando um processo entra na CPU e o mesmo pode ser retirado da CPU antes do término da sua execução

Escalonamento Preemptivo

- Permite a suspensão temporária de processos
- *Quantum* ou *time-slice*: período de tempo durante o qual um processo usa o processador a cada vez



Problema das trocas de processos

- Mudar de um processo para outro requer um certo tempo para a administração
 - salvar e carregar registradores e mapas de memória, atualizar tabelas e listas do SO, etc
- Isto se chama **troca de contexto**
- Suponha que esta troca dure 5 ms e que o *quantum* está ajustado para 20 ms
- Com esses parâmetros, **20% do tempo de CPU** (5 ms a cada 25 ms) será gasto com trocas de contexto

Solução?

- Para melhorar a eficiência da CPU, poderíamos ajustar o *quantum* para 500 ms
 - Agora o tempo gasto com troca de contexto é menos do que 1% - “desprezível”...
- O que aconteceria se dez usuários apertassem a tecla <ENTER> exatamente ao mesmo tempo, disparando cada um processo?
 - Dez processos serão colocados na lista de processo aptos a executar
 - Se a CPU estiver ociosa, o primeiro começará imediatamente, o segundo não começará cerca de $\frac{1}{2}$ segundo depois, e assim por diante
 - O último processo somente começará a executar 5 segundos depois do usuário ter apertado <ENTER>
 - Muitos usuários vão achar que o tempo de resposta de 5s é demais

“Moral da estória”

- Um *quantum* muito pequeno causa muitas trocas de contexto e diminui a eficiência da CPU ...
- Já um muito alto causa um tempo de resposta inaceitável para pequenas tarefas interativas

Quantum grande:

Diminui número de mudanças de contexto e *overhead* do S.O., mas...

Ruim para processos interativos

Categorias de Algoritmos de Escalonamento

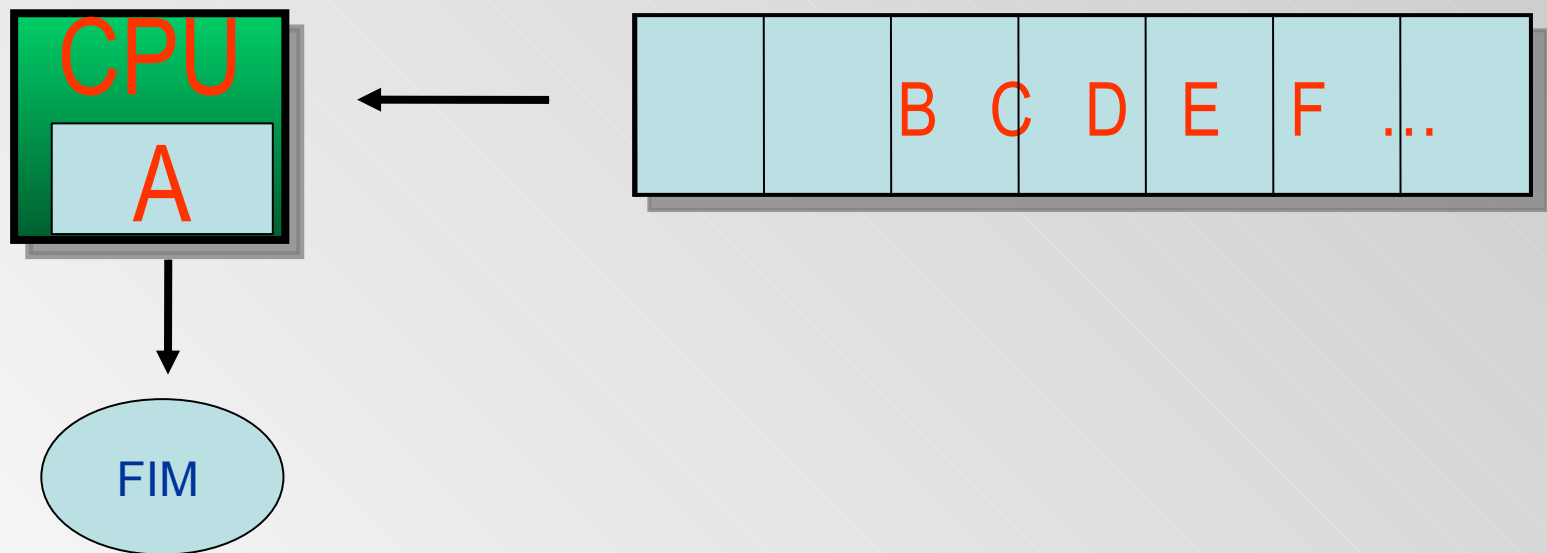
- Em lote (batch)
- Interativo
- Tempo-real
- Híbrido

Escalonamento em Sistemas em Lote

- First-come first-served (ou FIFO)
- Shortest job first (*job mais curto primeiro*)
- Shortest remaining Time/Job next

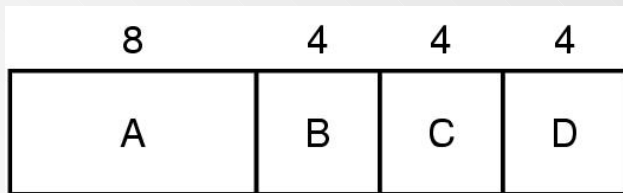
First-In First-Out (FIFO)

- Uso de uma lista de processos sem prioridade
- Escalonamento não-preemptivo
- Simples e justo
- Bom para sistemas em batch (lote)

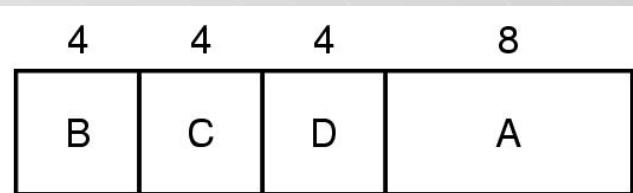


Escalonamentos baseados no tempo de execução

- Shortest Job First (não-preemptivo)
- Shortest Remaining Job Next (preemptivo)
- Melhora o tempo de resposta
- Não é justo: pode causar estagnação (*starvation*)
 - Pode ser resolvida alterando a prioridade dinamicamente



(a)



(b)

Exemplo de escalonamento *job mais curto primeiro* (**Shortest Job First – SJF**)

Escalonamento em Sistemas Interativos

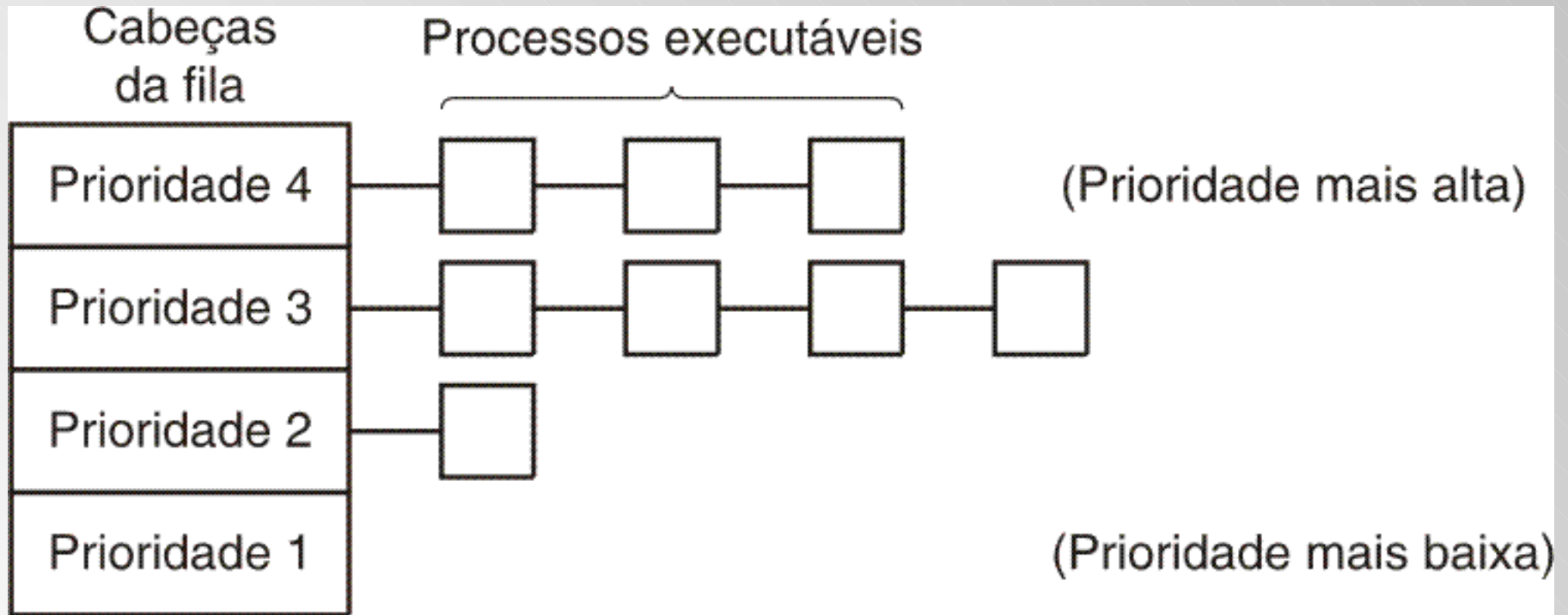
- Round-robin
- Prioridade
- Multiple queues
- Shortest process next
- Guaranteed scheduling
- Lottery scheduling
- Fair-share scheduling

Escalonamento em Sistemas Interativos (1)



- Escalonamento por alternância circular (*round-robin*)
 - lista de processos executáveis
 - lista de processos executáveis depois que B usou todo o seu quantum

Escalonamento em Sistemas Interativos (2)



Um algoritmo de escalonamento com quatro classes de prioridade

Políticas de Escalonamento

Híbridos

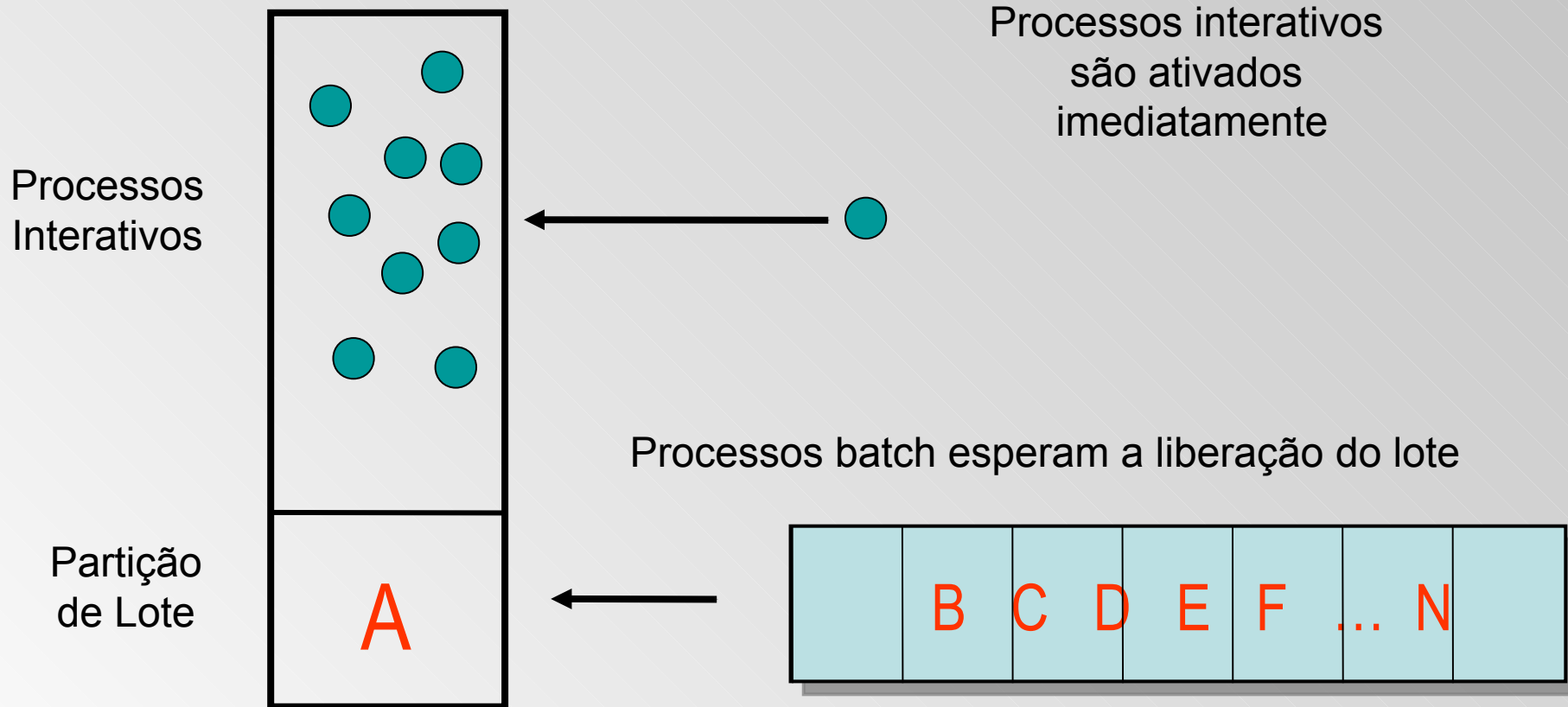
- Como combinar processos batch com interativos?
- Uso de Partições de Lote (batch)
 - O sistema aceita tantos processos batch quantas forem as partições de lote
 - O sistema aceita todos os processos interativos
 - Escalonamento em dois níveis

Segue...

Escalonamentos Híbridos

Partições de Lote

Memória



Escalonamentos Híbridos

Multiple Feedback Queue

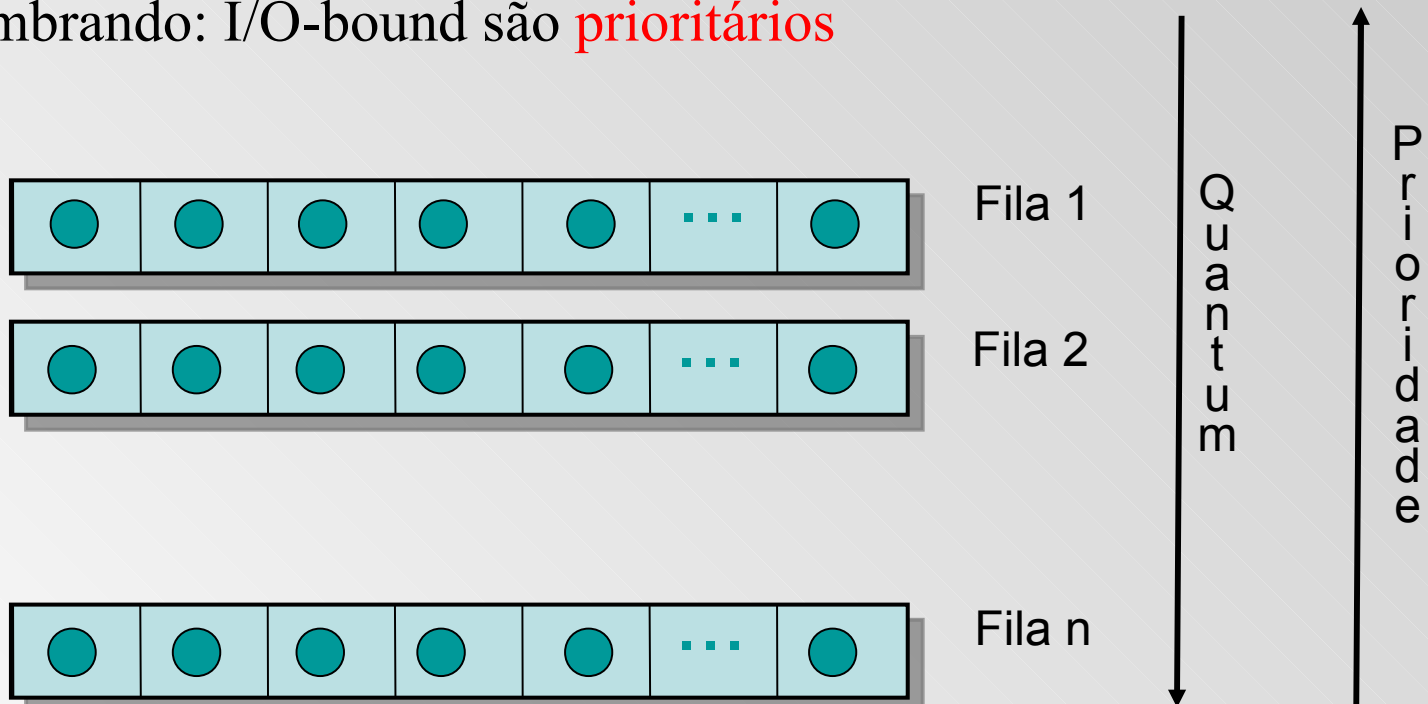
- Como saber *a priori* se o processo é CPU-bound ou I/O-bound?
- MFQ usa abordagem de prioridades dinâmicas
- Adaptação baseada no comportamento de cada processo
- Usado no VAX / VMS...

Segue...

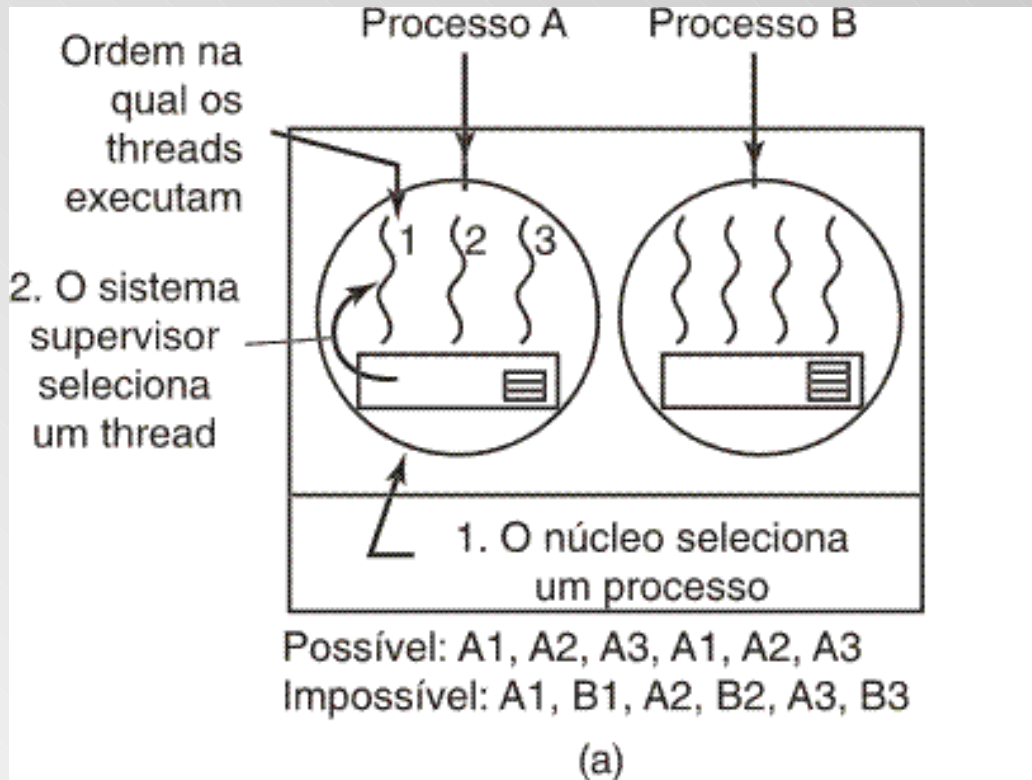
Escalonamentos Híbridos

Multiple Feedback Queue

- Novos processos entram na primeira fila (prioridade mais alta)
- Se acabar o quantum **desce** um nível
- Se requisitar E/S **sobe** um nível
 - Lembrando: I/O-bound são **prioritários**



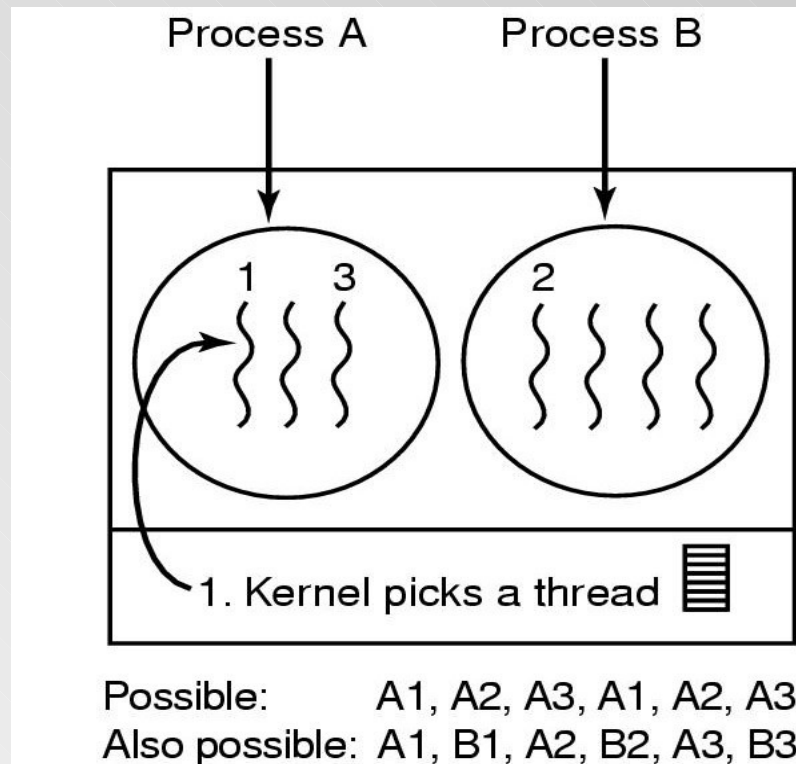
Escalonamento de Threads (1)



Possível escalonamento de **threads de usuário**

- processo com quantum de 50-mseg
- threads executam 5 mseg por surto de CPU

Escalonamento de Threads (2)



Possível escalonamento de **threads de núcleo**

- Muitas vezes, apenas *threads* são levadas em consideração
 - Independentemente dos processos

Conclusões

Como funcionam dois ou mais programas ao mesmo tempo?

- Conceitos
 - Processos x *Threads* (processos leves)
- Interrupção
 - Cooperação hardware-software
- Escalonamento
 - Tipos de processos
 - CPU-bound x I/O-bound
 - Lote (batch) x interativo
 - Filas de escalonamento
 - Long-term (admissão)
 - Short-term
 - I/O
- Escalonamento (cont.)
 - Objetivos
 - Justiça
 - Eficiência
 - Tempo de Resposta
 - Conceitos
 - Preempção
 - *Quantum (time-slice)*
 - Troca de contexto
 - Algoritmos
 - Propósito x Complexidade x Eficiência