Sistemas Operacionais

Prof. Dr. Helder Oliveira

Plano de Aula

- Introdução a Processos
- Threads vs Processos
- Características dos Processos
- Estados dos Processos

Processos

- Abstração de um programa em execução.
- Para entender Sistema Operacional necessário compreender o que é um processo.
- Sem a abstração de processo, a computação moderna não poderia existir.
- Dão suporte à possibilidade de haver operações (pseudo) concorrentes.

Processos

- Computadores realizam varias tarefas ao mesmo tempo.
- Ex:
  - Servidor da web:
    - Solicitações de páginas da web chegam de toda parte..
    - O servidor confere para ver se a página requisitada está em cache.
    - Se estiver, ela é enviada de volta;
    - Se não, uma solicitação de acesso ao disco é iniciada para buscá-la.
  - Ponto de vista da CPU:
    - Acesso ao disco levam uma eternidade.
    - Outras solicitações podem chegar.
    - Se houver outros discos
      - Algum método é necessário para modelar e controlar essa concorrência.
      - **Processos e Threads**

Multiprogramação

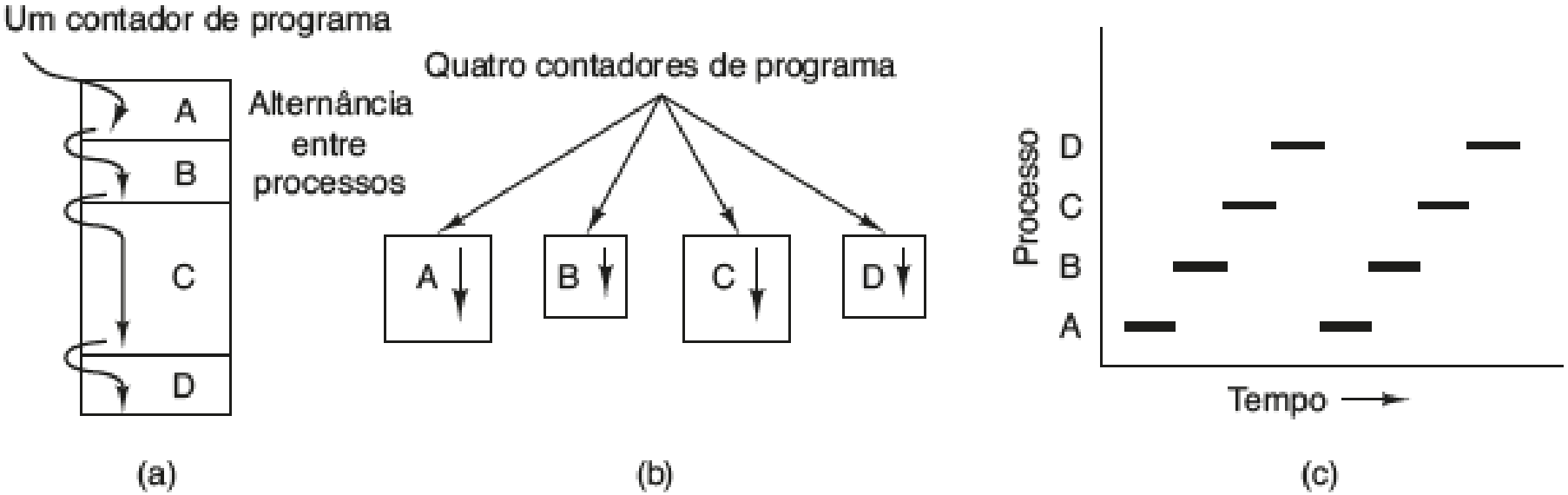
- Por que executar vários programas simultaneamente?
  - Permitir que vários usuários usem uma máquina simultaneamente
  - Melhorar a eficiência do sistema
- A CPU muda de processos rapidamente.
- **Pseudoparalelismo –** Ilusão do paralelismo**.**
- **A ação da CPU de trocar de um programa para o outro rapidamente é chamado de multiprogramação.**

Modelo de processo

- Controlar múltiplas atividades em paralelo – **Difícil de lidar!**
- Executáveis são organizados em uma série de processos sequenciais.
- **Um processo é apenas uma instância de um programa em execução, incluindo os valores atuais do contador do programa, registradores e variáveis.**
- **Processo:**
  - CPU virtual

Multiprogramação

FIGURA 2.1 (a) Multiprogramação de quatro programas. (b) Modelo conceitual de quatro processos sequenciais independentes. (c) Apenas um programa está ativo de cada vez.



Processos

- Em sistema operacional é conveniente diferenciar um programa de sua execução:
- Programa - entidade estática e permanente .
  - Composto por uma sequência de instruções: passivo sob o ponto de vista do sistema operacional.
- Processo - entidade dinâmica.
- Altera seu estado a medida que avança sua execução;
- O processo é uma abstração que representa um programa em execução;

Processos

- Analogia entre um Processo e um Cozinheiro:
  - Imagine um engenheiro com dotes culinários fazendo um bolo:
  - Receita = programa
  - Engenheiro cozinheiro = processador (CPU)
  - Ingredientes = dados de entrada.
  - Processo é a atividade desempenhada pelo cozinheiro em ler a receita, buscar os ingredientes e assar o bolo.

Processos

- Filho chorando picado por uma abelha.
- Registra onde parou = estado do processo atual
- Pega um livro de primeiros socorros = troca de processo

Criação de Processos

- Sistemas operacionais precisam criar processos.
- Quatro eventos principais fazem com que os processos sejam criados:
  1. Inicialização do sistema.
  2. Execução de uma chamada de sistema de criação de processo por um processo em execução.
  3. Solicitação de um usuário para criar um novo processo.
  4. Início de uma tarefa em lote.

Criação de Processos

1. Inicialização do sistema.
   - Processos de Primeiro plano:
     - Processos que interagem com usuários (humanos) e realizam trabalho para eles.
   - Processos de Segundo plano:
     - Não estão associados com usuários em particular, mas em vez disso têm alguma função específica.
     - Ex:
       - Processo pode ser projetado para aceitar e-mails, ficando inativo a maior parte do dia, mas subitamente entrando em ação quando chega um e-mail.
     - **Daemons**

Criação de Processos

2. Execução de uma chamada de sistema de criação de processo por um processo em execução.
   - Muitas vezes, um processo em execução emitirá chamadas de sistema para criar um ou mais processos novos para ajudá-lo em seu trabalho.
   - Ex:
     - Grande quantidade dados buscada na rede.
     - Dados serão processados.

Programas para listar processos

- Unix
  - Programa ps
- Windows
  - Gerenciador de tarefas

Criação de Processos

3. Solicitação de um usuário para criar um novo processo

- Unix
  - Processo executado por um programa ocupa a janela na qual ele foi iniciado.
- Windows
  - Quando um processo é iniciado, ele não tem uma janela, mas ele pode criar uma (ou mais), e a maioria o faz.

Criação de Processos

4. Inicio de uma tarefa em lote
   - Aplica-se somente aos sistemas em lote encontrados em grandes computadores
   - Ex:
     - Gerenciamento de estoque de uma cadeia de lojas.
     - Usuários podem submeter tarefas em lote ao sistema.

Criação de Processos

- Tecnicamente, em todos esses casos, um novo processo é criado por outro já existente executando uma chamada de sistema de criação de processo.
- Execução de uma chamada de sistema para criar o novo processo.
- Essa chamada de sistema diz ao sistema operacional para criar um novo processo e indica, direta ou indiretamente, qual programa executar nele.

Termino de processos

- O processo termina por uma das seguintes condições:
  1. Saída normal (voluntária).
  2. Erro fatal (involuntário).
  3. Saída por erro (voluntária).
  4. Morto por outro processo (involuntário).

Termino de Processos

1. Saída Normal
   - Trabalho concluído
   - Quando um compilador termina de traduzir o programa dado a ele, o compilador executa uma chamada para dizer ao sistema operacional que ele terminou.

Termino de Processos

2. Erro fatal
   - Processo descobre um erro fatal.
   - Ex:
     - Comando cc foo.c
     - Programa não existe

Termino de Processos

3. Saída por erro
   - Um erro causado pelo processo, muitas vezes decorrente de um erro de programa
   - Ex:
     - Executar uma instrução ilegal.
     - Referenciar uma memória não existente.
     - Dividir por zero.

Termino de Processos

4. Morto por outro processo
   - Quando um processo executa uma chamada de sistema dizendo ao sistema operacional para matar outro processo.
   - Em UNIX, essa chamada é kill.
   - No Win32 é TerminateProcess
   - Necessário de autorização.

Hierarquia de processos

- O processo pai e o processo filho continuam a ser associados de certas maneiras.
- O processo filho pode criar mais processos – **Hierarquia de processos.**
- Em UNIX, um processo e todos os seus filhos e demais descendentes formam juntos um grupo de processos.
  - Quando um usuário envia um sinal do teclado, o sinal é entregue a todos os membros do grupo de processos associados com o teclado no momento.
  - Cada processo pode pegar o sinal, ignorá-lo, ou assumir a ação predefinida, que é ser morto pelo sinal.
  - Não podem deserdar seus filhos.

Hierarquia de processos

- No Windows não tem conceito de hierarquia de processo.
- Todos os processos são iguais.
- O único indício de uma hierarquia ocorre quando um processo é criado e o pai recebe um identificador especial (chamado de handle) que ele pode usar para controlar o filho.
- No entanto, ele é livre para passar esse identificador para algum outro processo, desse modo invalidando a hierarquia.

Composição de processos

- Um processo é composto por:
  - Programas.
  - Dados.
  - Contexto (valores).

Características/Propriedades de um processo

- Um programa pode gerar (criar) vários processos.
- Um processo tem duas partes:
  - Ativa - fluxo de controle.
  - Passiva - espaço de endereçamento (memória, registradores, arquivos).

Estados de processos

- Um processo pode estar nos estados em execução, bloqueado ou pronto.
  1. Em execução (realmente usando a CPU naquele instante).
  2. Pronto (executável, temporariamente parado para deixar outro processo ser executado).
  3. Bloqueado (incapaz de ser executado até que algum evento externo aconteça).

Estados de processos

- Processos muitas vezes precisam interagir entre si.
- Entrada de um processo pode ser a saída de outro.
- No comando no Shell temos 2 processoss:
  - cat chapter1 chapter2 chapter3 | grep tree
- Pode acontecer do grep está pronto antes de ter uma entrada.
- Processo é bloqueado porque logicamente não pode continuar.
  - Falta de entrada.
- Processo também pode ser bloqueado por decisão do SO

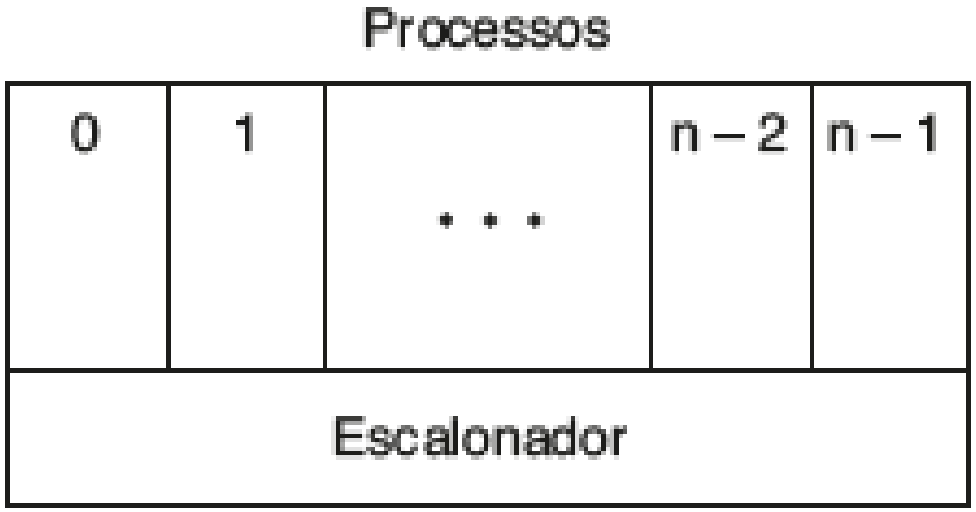
Estados de processos

FIGURA 2.2 Um processo pode estar nos estados em execução, bloqueado ou pronto. Transições entre esses estados ocorrem como mostrado.

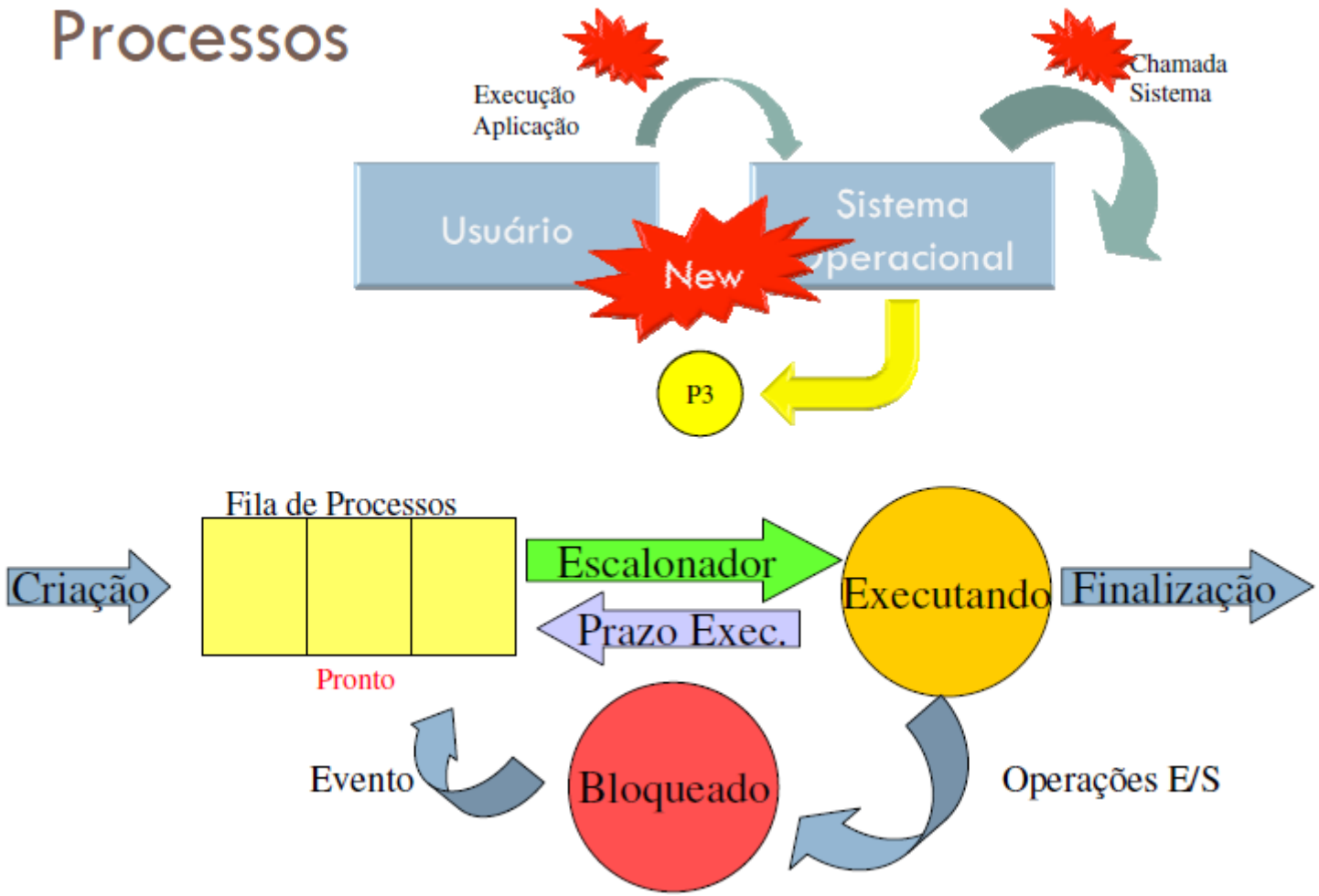
1. O processo é bloqueado aguardando uma entrada
2. O escalonador seleciona outro processo
3. O escalonador seleciona esse processo
4. A entrada torna-se disponível

Escalonador

FIGURA 2.3 O nível mais baixo de um sistema operacional estruturado em processos controla interrupções e escalonamento. Acima desse nível estão processos sequenciais.



Funcionamento



Processos

- **Thread**
  - Denota um fluxo de controle (Processo leve).
  - Por questão de eficiência, processos podem ter múltiplas threads que compartilham o espaço de endereçamento do processo.
- **Escalonador**
  - Programa que controla/decide que thread deve ser executada a cada instante.

Resumo:

- Um processo é uma abstração de um programa em execução
- Funções do S.O.
  - Alocar recursos a processos
  - Suportar criação de processos pelo usuário
  - Suportar comunicação entre processos

É função do Escalonador de Processos:
Dividir tempo de CPU para diferentes processos de forma a maximizar a utilização da CPU, fornecendo um tempo de resposta razoável!

Leitura

- SISTEMAS OPERACIONAIS MODERNO 4ª edição
  - 2.1 Processos

Dúvidas?