Sistemas Operacionais

Multiprogramação

2ª edição

Capítulo 2

Revisão: Fev/2003

Multiprogramação

- Tornar mais eficiente o aproveitamento dos recursos do computador
- Execução simultânea* de vários programas
  - Diversos programas são mantidos na memória
  - Conceitos necessários a multiprogramação
    - Processo
    - Interrupção
    - Proteção entre processos
- Próprio sistema operacional é um programa

O conceito de processo (1)

- Diferenciação entre o programa e sua execução
- Programa:
  - Entidade estática e permanente
    - Seqüência de instruções
    - Passivo sob o ponto de vista do sistema operacional
- Processo:
  - Entidade dinâmica e efêmera
    - Altera seu estado a medida que avança sua execução
  - Composto por programa (código), dados e contexto (valores)

O conceito de processo (2)

- Abstração que representa um programa em execução
- Diferentes instâncias
  - Um programa pode ter várias instâncias em execução, i.e., diferentes processos
  - Mesmo código (programa) porém dados e momentos de execução (contexto) diferentes
- Forma pela qual o sistema operacional “enxerga” um programa e possibilita sua execução
- Processos executam:
  - Programas de usuários
  - Programas do próprio sistema operacional (*daemons*)

Ciclos de vida de um processo

- Criação
- Execução
- Término

Ciclos de vida de um processo: criação

- Momento da execução
- Chamadas de sistemas
  - e.g.: *fork, spawn*, etc
- Podem ser associados a uma sessão de trabalho
  - e.g.: *login* de usuários: login + senha → *shell* (processo)
- Identificado por um número único (PID)

Ciclos de vida de um processo: execução (1)

- Processos apresentam dois ciclos básicos de operação
  - Ciclo de processador
    - Tempo que ocupa a CPU
  - Ciclo de entrada e saída
    - Tempo em espera pela conclusão de um evento (e.g. E/S)
- Primeiro ciclo é sempre de processador
  - Trocas de ciclos por:
    - CPU → E/S: chamada de sistema
    - E/S → CPU: ocorrência de evento (interrupção)

Ciclos de vida de um processo: execução (2)

- Processos
  - CPU *bound*
    - Ciclo de processador >> ciclo de E/S
  - I/O *bound*
    - Ciclo de E/S >> ciclo de processador
- Sem quantificação exata
- Situação ideal:
  - Misturar processos CPU *bound* com I/O *bound*
    - Benefícios a nível de escalonamento

Ciclos de vida de um processo: término

- Final de execução (normal)
- Por erros
  - e.g: proteção, aritméticos, E/S, tentativa de execução de instruções inválidas, falta de memória, exceder tempo de limite
- Intervenção de outros processos (*kill*)
- *Log off* de usuários

Relacionamento entre processos (1)

- Processos independentes
  - Não apresentam relacionamentos com outros processos
- Grupo de processos
  - Apresentam algum tipo de relacionamento
    - e.g. filiação
  - Podem compartilhar recursos
  - Definição de hierárquia

Relacionamento entre processos (2)

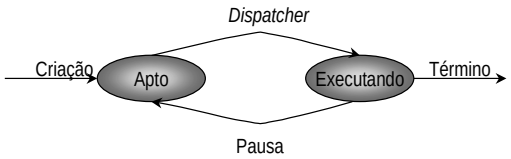
- Hierárquia de processos:
  - Processo criador é processo pai
  - Processo criado é processo filho
- Representação através de uma árvore
  - Evolução dinâmica
- Semântica associada: O que fazer na destruição de um processo?
  - Toda a descendência "morre"
  - A descendência é herdada pelo processo "vô"
  - Postergar a destruição efetiva do processo pai até o final de todos processos filhos

Estados de um processo

- Após criado o processo necessita entrar em ciclo de processador
- Hipotéses:
  - Processador não está disponível
  - Vários processos sendo criados
- Que fazer?
  - Criação de uma fila de aptos (p/ espera pelo processador)

Modelo simplificado a dois estados

- Manter uma fila de processos aptos a executar
  - Esperando pelo processador ficar livre
- Escalonador (*dispatcher*):
  - Atribui o processador a um processo da fila de aptos
  - Pode prevenir um único processo de monopolizar o processador



Limitação do modelo simplificado

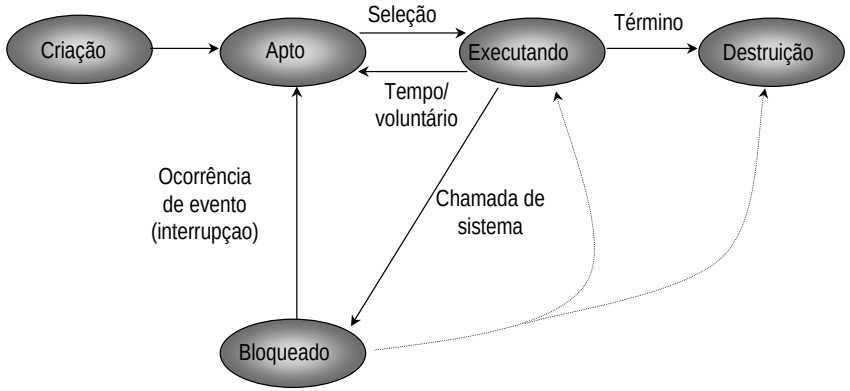
- Causas para um processo não executar
  - Esperando pelo processador
    - Aptos para executar
  - Esperando pela ocorrência de eventos externos
    - Bloqueado
- Escalonador não pode selecionar um processo bloqueado, logo modelo a dois estados não é suficiente
  - Criação de novos estados

Modelo de 5 estados (1)

- Executando (*Running*)
- Apto (*Ready*)
- Bloqueado (*Blocked*)
- Criação (*New*)
- Destruição (*Exit*)

Modelo a 5 estados (2)

- Necessidade de filas



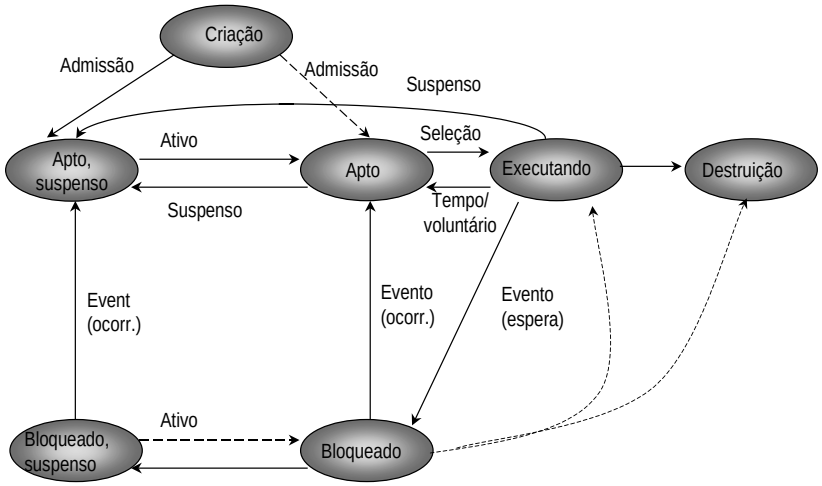
Processos suspensos

- Processador é mais rápido que operações de E/S
  - Possibilidade de todos processos estarem bloqueados esperando por E/S
- Liberar memória ocupada por estes processos
  - Transferidos para o disco (*swap*)
- Estado bloqueado assume duas situações:
  - Bloqueado com processo em memória
  - Bloqueado com processo no disco
- Necessidade de novos estados
  - Bloqueado, suspenso (*Blocked, suspend*)
  - Apto, suspenso (*Ready, suspend*)

Razões para suspender um processo

- *Swapping*:
  - SO necessita liberar memória para executar um novo processo
- Solicitação do usuário
  - Comportamento típico de depuradores
- Temporização:
  - Processo deve ter sua execução interrompida por um certo período de tempo
- Processo suspender outro processo
  - e.g. sincronização

Diagrama de estados de processos



Suporte de hardware à multiprogramação

- O compartilhamento de recursos comuns implica em garantir que a execução incorreta de um programa não influencie a execução de outro programa
- A implementação da multiprogramação explora características do hardware dos processadores
- Mecanismos básicos:
  - Dois modos de operação
  - Interrupção
  - Proteção de periféricos, memória e processador

Modos de operação do processador

- Arquitetura de processadores oferecem mecanismos para diferenciar pelo menos dois modos diferentes de operação
  - Modo supervisor (privilegiado/protegido)
    - Possibilita a execução de todas as instruções do processador
    - Modo de execução sistema operacional
  - Modo usuário
    - Certas instruções (privilegiadas) não podem ser executadas
    - Modo de execução dos processos usuários
- Chaveamento de modos:
  - Interrupção (modo usuário → modo protegido)
  - Instrução (modo protegido → modo usuário)

Mecanismo de interrupção (1)

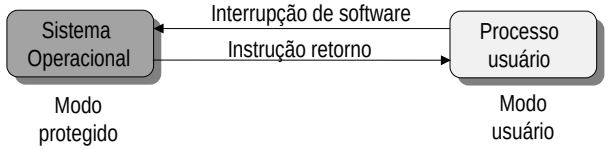
- Sinaliza a ocorrência de algum evento
- Provoca a execução de uma rotina especial
  - Tratador de interrupção
- Ciclo de execução de uma interrupção
  - Prepara a transferência de controle para o tratador (salvamento do contexto de execução)
  - Desvia controle para tratador
  - Retorna execução (restaura contexto de execução)

Mecanismo de interrupção (2)

- Tipos de interrupção
  - Hardware: ocorrência de evento externo
  - Software: execução de uma instrução específica
  - Exceção: erros de execução (*overflow*, *undeflow*...)
- Identificadas por um número
  - Vetor de interrupção
- Prioridades
- Instruções privilegiadas

Proteção de periféricos

- Instruções de E/S são privilegiadas
- Como processos usuários realizam operações de E/S já que estas são instruções privilegiadas?
  - Chamadas de sistema



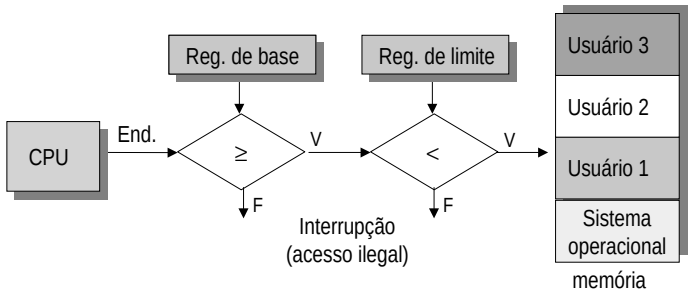
Chamada de sistema

- Método empregado para um processo usuário solicitar serviços ao sistema operacional.
  - Normalmente baseada em interrupções de software (*traps*)
  - Aciona a rotina de tratamento de interrupção
    - Identifica serviço requisitado
    - Verifica validade dos parâmetros
    - Executa o serviço
    - Retorna ao processo do usuário

Proteção de memória (1)

- Necessário para evitar que usuário corrompa espaços de memória não-pertencentes a seus processos
- Baseado em facilidades da arquitetura do processador:
  - Registrador de base
  - Registrador de limite
- Faixa de endereçamento fora da área delimitada pelos registradores base e limite é protegida
- Possível proteger dispositivos de E/S quando a técnica E/S mapeada em memória é empregada

Proteção de memória (2)



Proteção do processador

- Para garantir a execução do sistema operacional uma interrupção de tempo (*timer*) ocorre periodicamente
- Interrupção de tempo:
  - Empregada para implementar multiprogramação
  - Mantém contabilização de tempo para o sistema operacional (relógio)
- Instruções relacionadas com a programação do tempo são privilegiadas

Leituras complementares

- R. Oliveira, A. Carissimi, S. Toscani *Sistemas Operacionais* Editora Sagra-Luzzato, 2001.
  - Capítulo 2.
- A. Silberchatz, P. Galvin *Operating System Concepts*. 4th edition. Addison-Wesley.
  - Seções 2.1, 2.2, 2.5, 4.1, 4.2 e 4.3