Processos

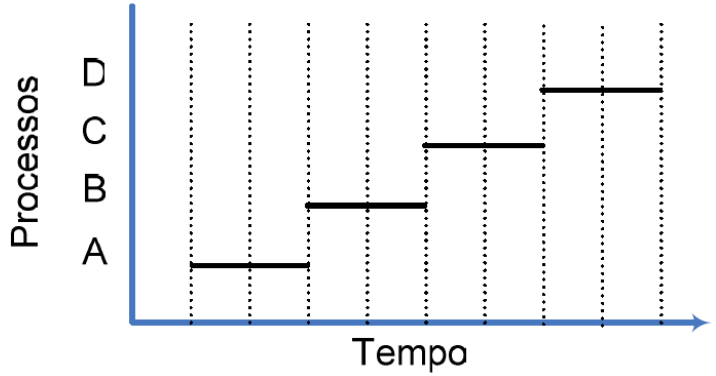
Fontes:

Silberschatz cap 3
Tanenbaum cap 2

- Conceituação
- Gerência de processos
- Estrutura de processos
- Troca de contexto
- Estados de processos

Mono x Multiprogramação

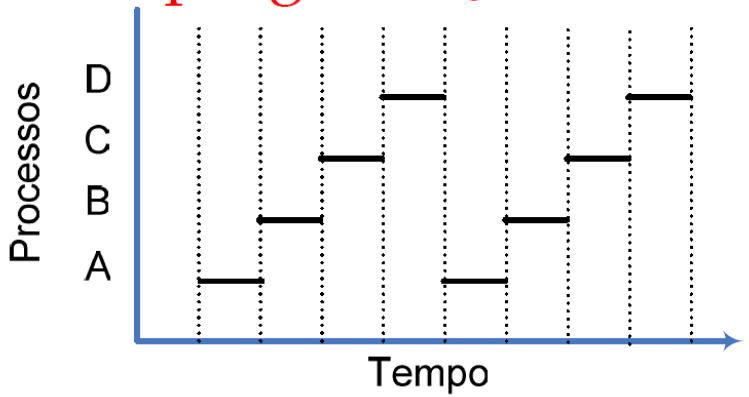
Monoprogramação



É necessário uma fila para escalonar os processos.

Estes são executados por completo antes de ceder a CPU.

Multiprogramação



Antes de terminar todo o processamento, os processos podem ser interrompidos para ceder a CPU a um outro processo.

Isso cria uma falsa impressão de paralelismo.

Multiprogramação

- "*pseudoparalelismo*": coleção de processos sendo executados alternadamente na CPU
    - **execução concorrente**
- ideia: reduzir o desperdício de CPU devido às operações de E/S
- objetivos:
    - aumentar a taxa de uso da CPU
    - melhorar a utilização dos recursos
    - aumentar o *throughput*

Conceito(s) de Processo

- job = processo
- programa em execução
  - "*o processo é uma diferenciação entre o programa e sua execução*"
- entidade ativa que compete por recursos oferecidos pelo SO:
  - acesso a discos, periféricos e, principalmente, CPU
- interage com outros processos
- execução: sequencial

Processo: conceitos relacionados

- Programa – código, sequência de instruções (conceito estático)
- Job – programa batch em execução
- Tarefa (task) – execução de um fluxo de sequencial de instruções. Pode ser implementada como um processo, uma thread, um job ou uma transação
- Processo – programa em execução (conceito dinâmico)
- Thread – processo leve (compartilha código)

Programa x Processo

- Programa
  - Entidade estática e permanente
    - Sequência finita de instruções
    - Passivo sob o ponto de vista do SO (não se altera c/ passar do tempo)
    - Armazenado em disco
- Processo
  - Entidade dinâmica e efêmera
    - Altera seu estado a medida que avança sua execução
  - Um programa pode ter várias instâncias em execução
    → diferentes processos

Modelo de processo (1)

- Organização sequencial
- Processo: programa em execução, acompanhado dos valores atuais de PC, registradores e variáveis

- Obs.:
  - Com a CPU alternando entre processos, a velocidade de execução de cada processo:
    - não será uniforme
    - não será reproduzível se os mesmos processos forem executados novamente

Modelo de processo (2)

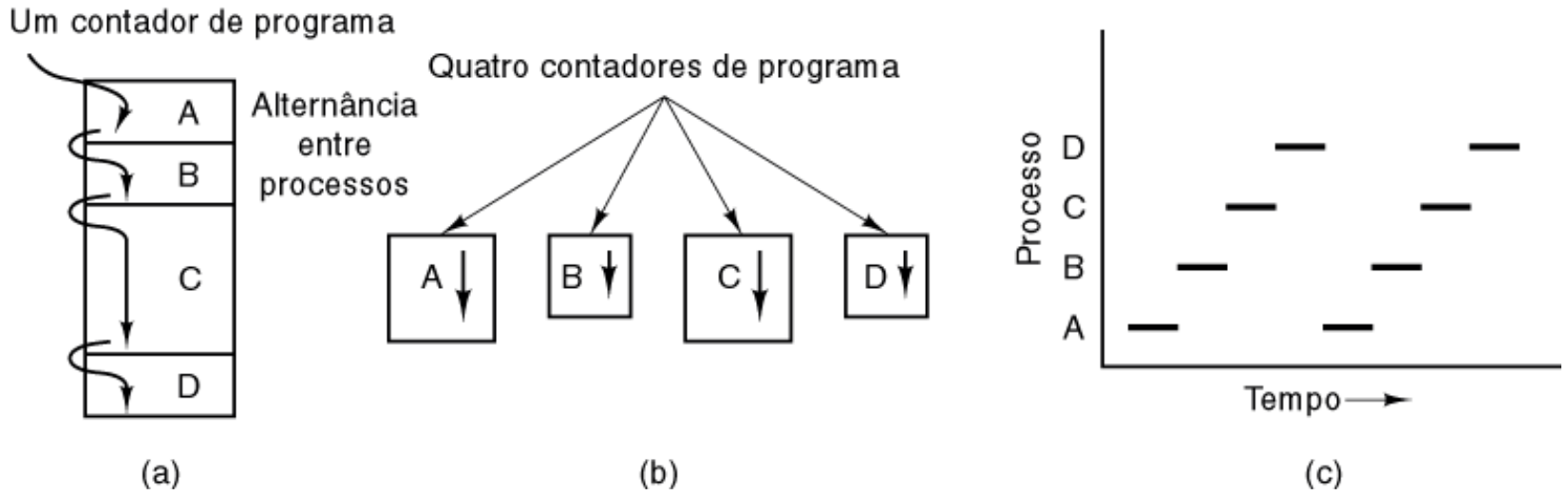
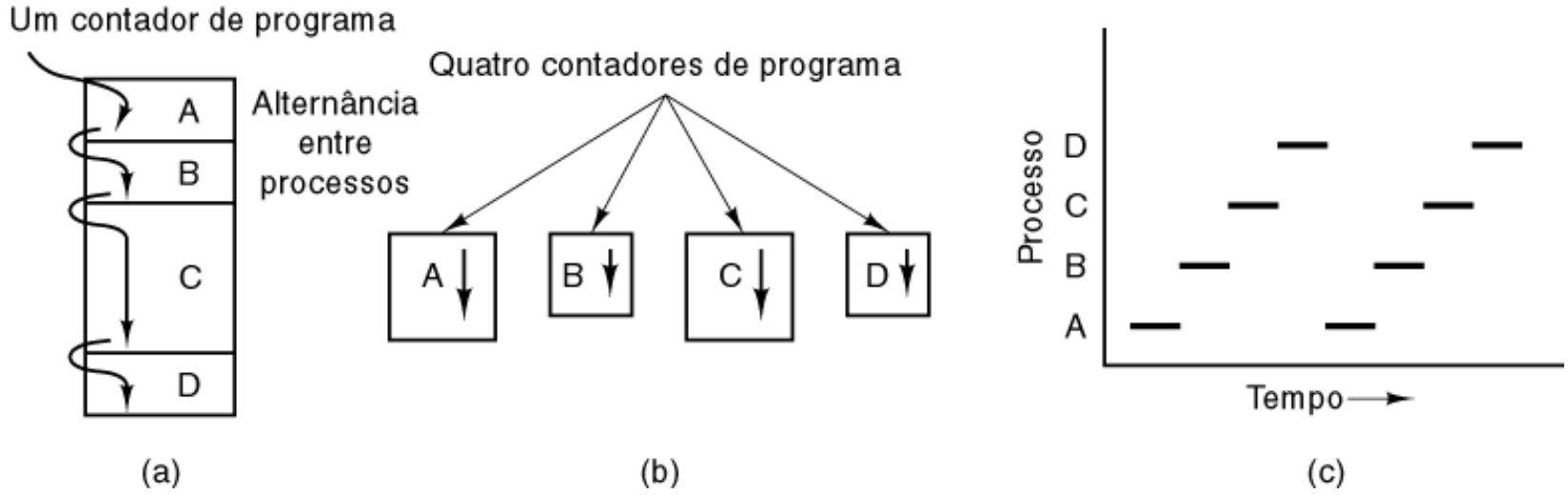


Figura (a): 4 processos na memória

Figura (b): cada processo possui seu PC

Figura (c): sequência de execução dos processos

Modelo de processo (3)



- Realidade: existe apenas 1 PC
  - Ao executar, o PC do sistema é carregado com o endereço do processo “selecionado”
  - Ao terminar o tempo de CPU do processo “selecionado”, o PC “físico” é salvo no PC “lógico” do processo na memória

Processo: o que esperar do SO?

- Alternar a execução de processos
  - Maximizar o uso da CPU
  - Fornecer tempo de resposta razoável
- Alocar recursos a processos
- Suportar a criação de processos pelo usuário
- Suportar a comunicação entre processos

Controle de processos

- Para gerenciar processos o SO precisa conhecer:
  - onde o processo está localizado
  - os atributos do processo

- Como um processo é representado?
  - *Imagem do processo* = programa + dados + pilha(s) + atributos
  - Atributos: informações necessárias pelo SO
  - Imagem está na MP

Representação da imagem de um processo

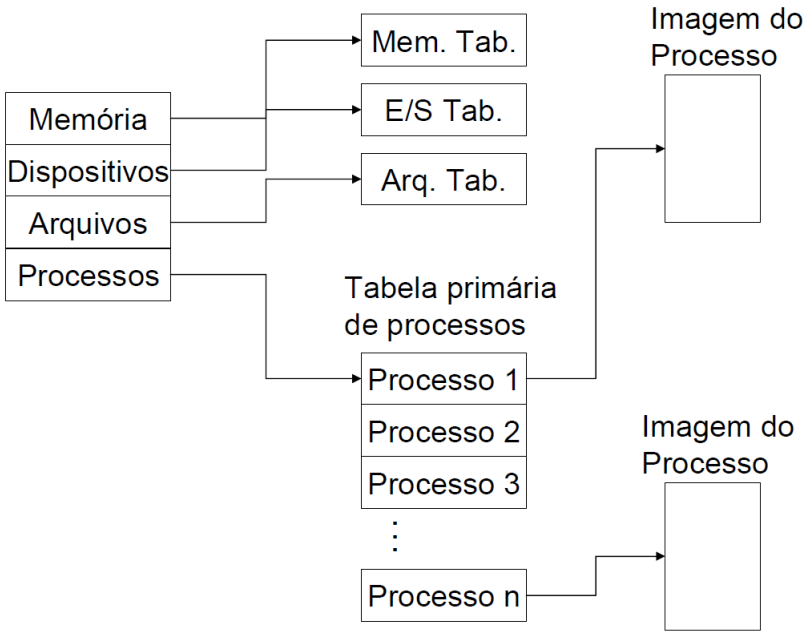
- Um processo é representado por uma imagem:
  - segmento de código (*o que ele vai fazer?*)
  - espaço de endereçamento (*onde, na memória, ele vai fazer algo?)*
  - contexto (*o que ele precisa para fazer algo?*)
- Uma parte da imagem é de responsabilidade do usuário, a outra é gerenciada em modo protegido (SO)

Componentes de um processo

- Processo inclui:
  - contador de programa (PC)
    - indica próxima instrução a executar
  - pilha de execução (stack)
    - com valores temporários (parâmetros de funções, endereços de retorno, ...)
  - área de dados
    - com os valores das variáveis globais

Implementação de processos

- Tabelas do sistema
  - SO precisa manter informações sobre o estado atual de cada processo e recurso
  - Usa estruturas de controle:
    - Tabelas de memória
    - Tabelas de dispositivos
    - Tabelas de arquivos
    - **Tabelas de processos**



Gerência de processos – PCB

- SO mantém uma **Tabela de Processos**
  - armazena informações que variam de um processo para outro
  - há uma entrada na tabela para cada processo
  - cada entrada é chamada de PCB (Bloco de Controle de Processos)
  - PCB
    - vetor ou lista encadeada de estruturas
    - um para cada processo do sistema

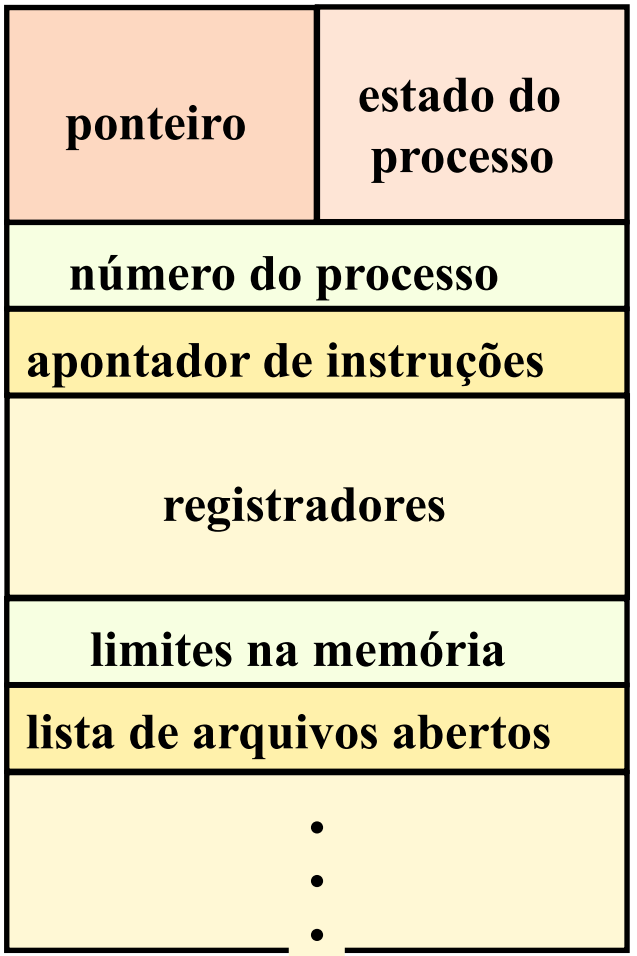
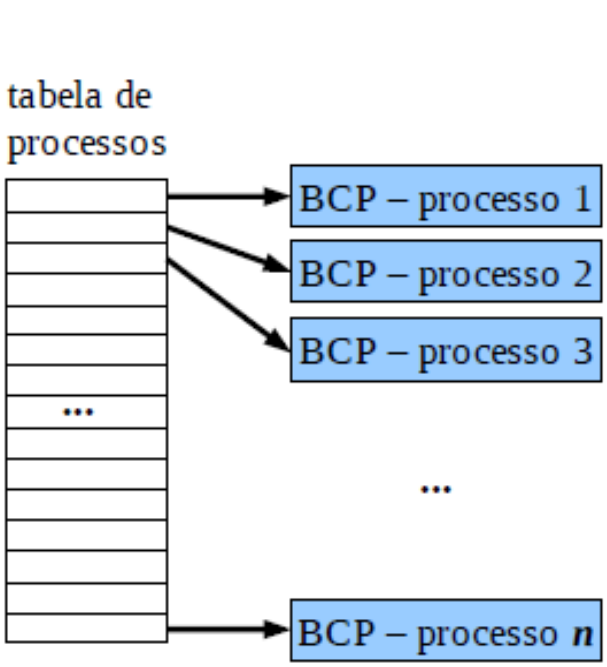
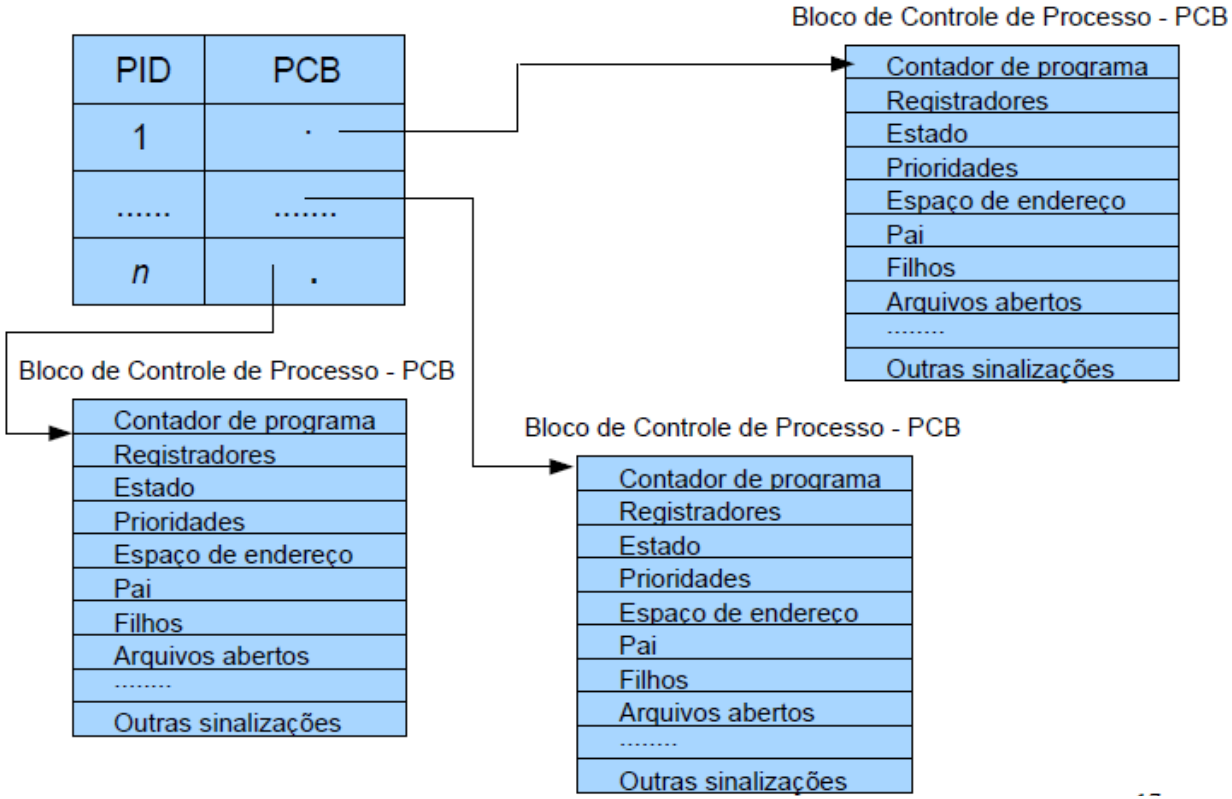
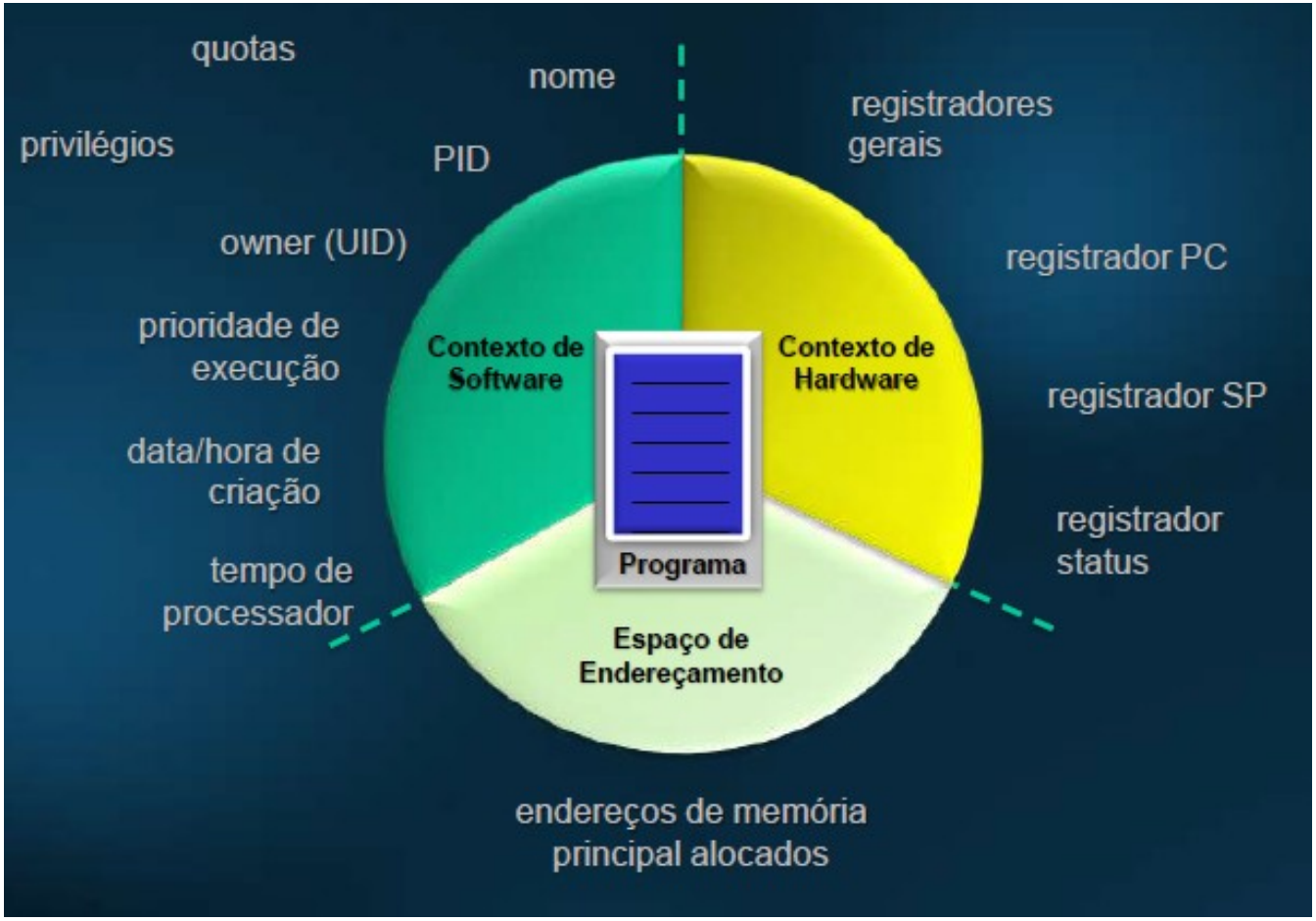


Tabela de Processos e Bloco de Controle de Processos





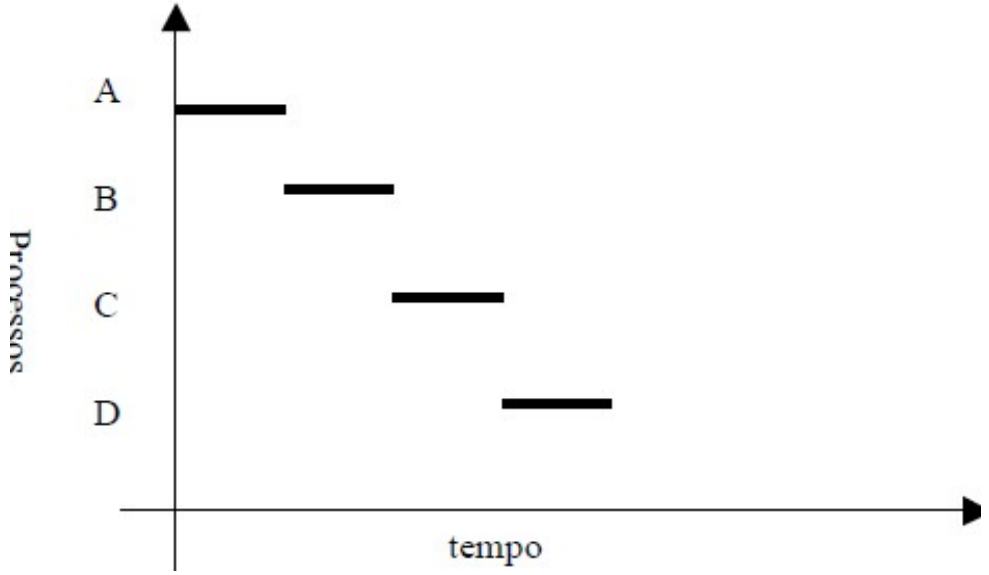
Estrutura do processo: PCB



Exemplo de PCB

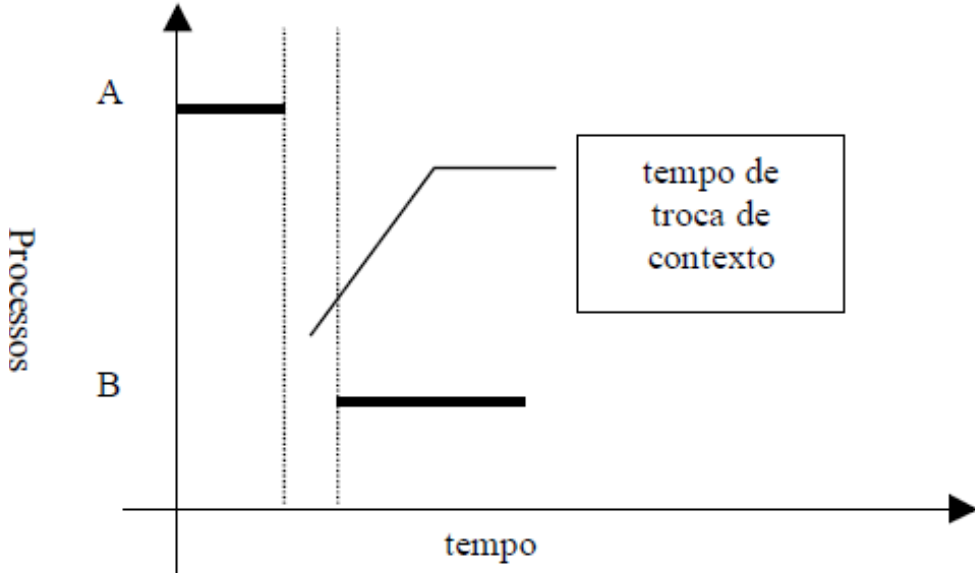
```
struct desc_proc {
    char estado_atual;              /* Estado do processo */
    int prioridade;                 /* Prioridade do processo */
    unsigned inicio_memoria;        /* Endereço inicial da memória */
    unsigned tamanho_memoria;       /* memória usada (bytes) */
    struct arquivo arq_abertos[20]  /* Arquivos abertos */
    unsigned tempo_de_CPU;          /* Tempo de CPU */
    unsigned proc_pc;               /* Valor do PC (registrador) */
    unsigned proc_sp;               /* Valor do SP (registrador) */
    unsigned proc_acc;              /* Valor do ACC (registrador) */
    unsigned proc_rx;               /* Valor do RX (registrador) */
    struct desc_proc *proximo       /* Aponta para o próximo */
}
struct desc_proc tab_desc[MAX_PROCESS];
```

Execução de Processos



Processos em execução timesharing

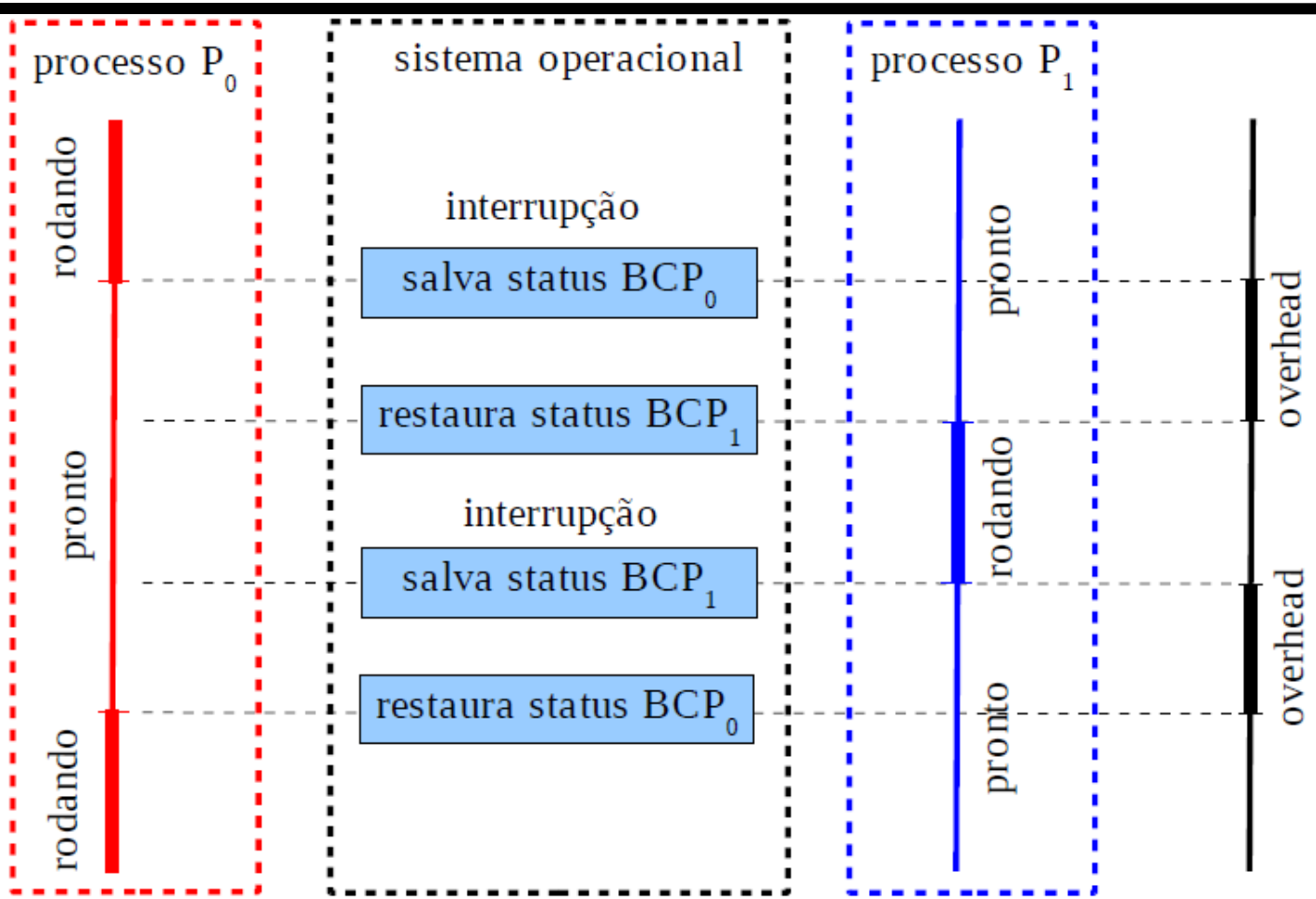
Tempo adicional da **troca de contexto**



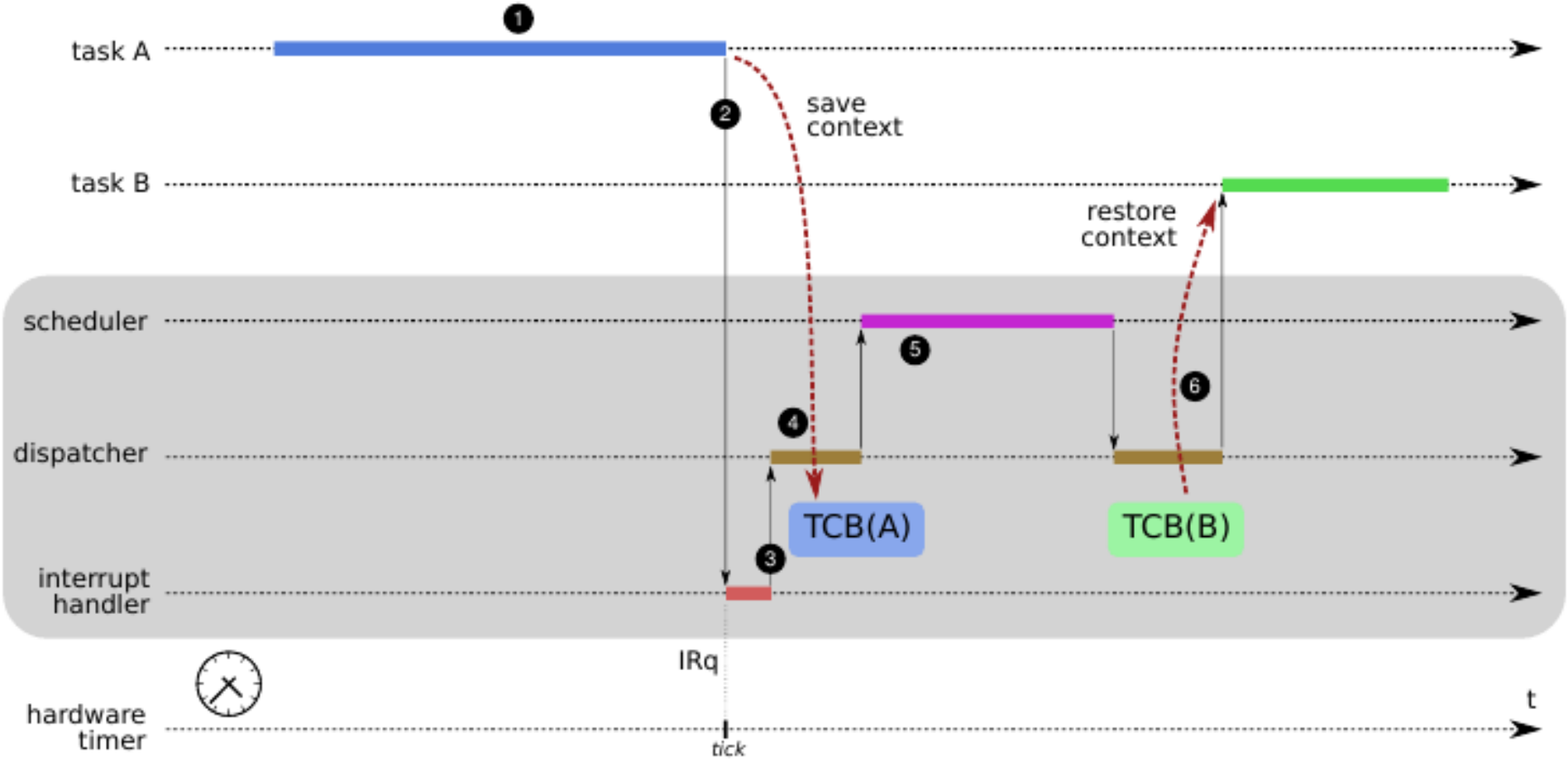
Troca de Contexto (1)

- Lembrando ...
  - contexto de um processo
    - informações necessárias p/ que o processo possa ser restaurado a partir do ponto em que foi interrompida a sua execução
  - a troca de um processo de hw por outro no processador é chamada de troca de contexto

Troca de Contexto (2)



Dispatcher x scheduler



créditos: prof. Maziero

Troca de Contexto: etapas

- Salvar o estado do processador
- Mudar o estado do processo
- Mudar o processo para a fila apropriada
- Selecionar o novo processo
- Atualizar o PCB do novo processo
- Modificar os mapeamentos de memória
- Restaurar o estado do processador

Troca de Contexto: causas

- interrupção do relógio: fatia de tempo de posse do processador expirou
- interrupção de I/O: resposta de um dispositivo de I/O
- falta de memória (page fault): endereço de memória procurado está na memória virtual (disco)
- interrupção por erro: associada a erro na execução de uma instrução
- chamada de sistema: solicitação de um serviço do SO (ex.: I/O)

Overhead da Troca de Contexto

- tempo da troca de contexto = desperdício (*overhead*)
  - nenhum processo está rodando
  - depende do suporte de hardware
  - pode consistir em um gargalo no sistema
  - atraso varia, conforme o hw (tamanho da memória, nº de registradores, velocidade da CPU, ...)
- possível solução: **threads**
  - visa diminuir o tempo gasto na criação/eliminação de um PCB p/ cada subprocesso
  - Threads - compartilham o mesmo espaço de endereçamento

Ciclos de um processo

- A execução de um processo é composta por ciclos:
  - na CPU (***CPU-burst***)
  - na E/S (***I/O-burst***)
  - primeiro ciclo é sempre de processador
  - Troca de ciclos por:
    - CPU → E/S: chamada de sistema
    - E/S → CPU: ocorrência de evento (interrupção)
    - CPU → E/S ou E/S → CPU: interrupção

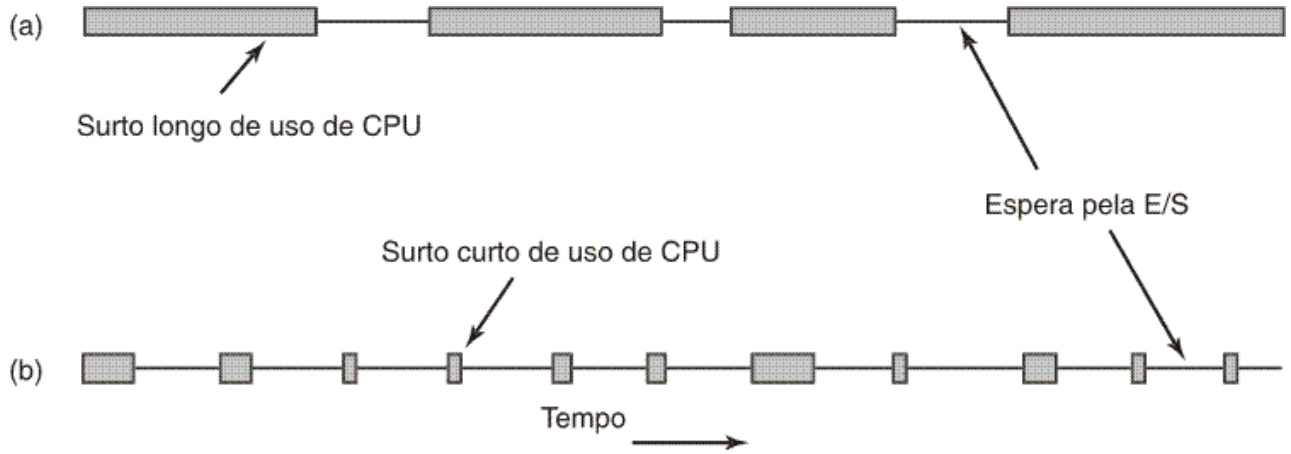
Classificação dos Processos: taxa de uso da CPU ou I/O

(a) processo CPU-bound

- Ciclo de CPU >>> ciclo de I/O
- processamento predominante, pouco I/O
- Ex.: processo p/ multiplicação de matrizes, processo renderizador de imagens

(b) processo I/O-bound

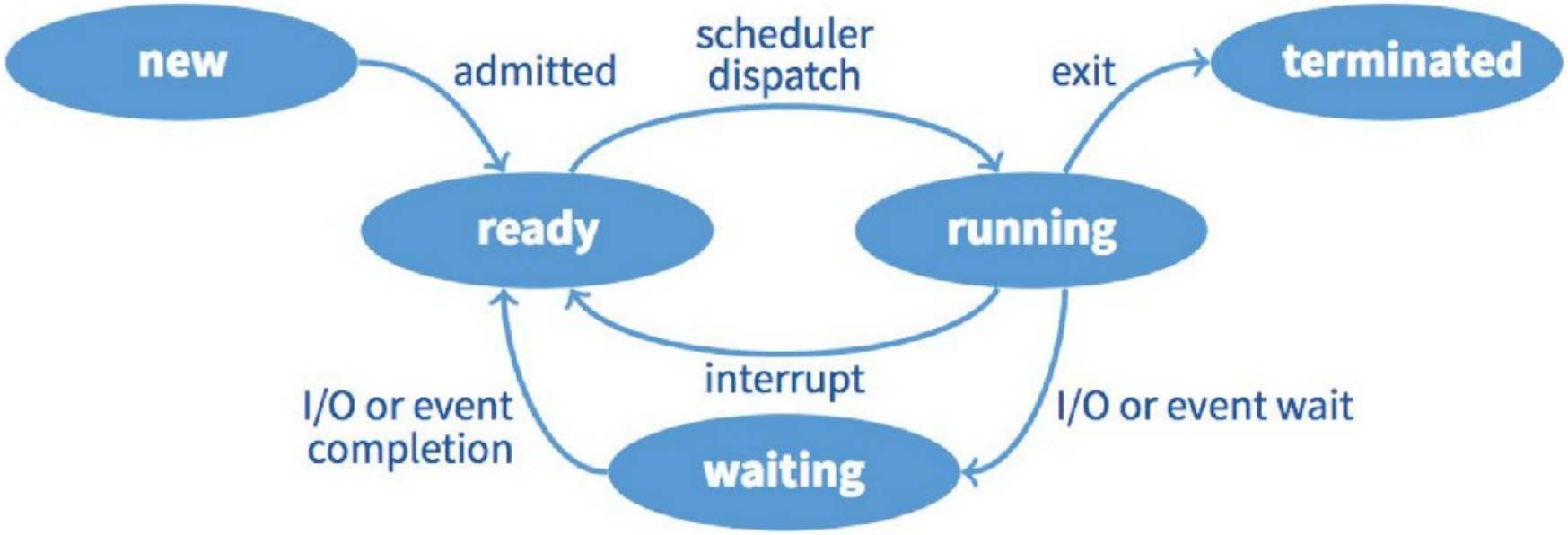
- Ciclo de I/O >>> ciclo de CPU
- I/O predominante (ou espera por I/O), pouco processamento
- Ex.: processo p/ cópia de arquivos, processos interativos (processadores de texto)



Estados de um processo

- **novo (new)**
  - processo em criação (sendo admitido no sistema)
- **executando (running)**
  - instruções estão sendo executadas
- **bloqueado (waiting)**
  - processo está esperando a ocorrência de um evento
- **pronto (ready)**
  - processo está esperando p/ ganhar o processador
- **terminado (terminated)**
  - processo terminou sua execução

Diagrama de estados de um processo



Exemplos de transições de estado

Ready → running: Algoritmo de escalonamento

Running → ready: Interrupção por tempo
Interrupção pelo escalonador
Decisão espontânea (yield)

Running → waiting: Solicitação de serviço (ex.: E/S)

Waiting → ready: Interrupção (término de operação)

Running → exit: Término normal