INE5412 Sistemas Operacionais I

L. F. Friedrich

Capítulo 3

Memoria Virtual – Projeto/Implementação

Sistemas operacionais modernos
Terceira edição

ANDREW S. TANENBAUM

Capítulo 3
Gerenciamento de memória

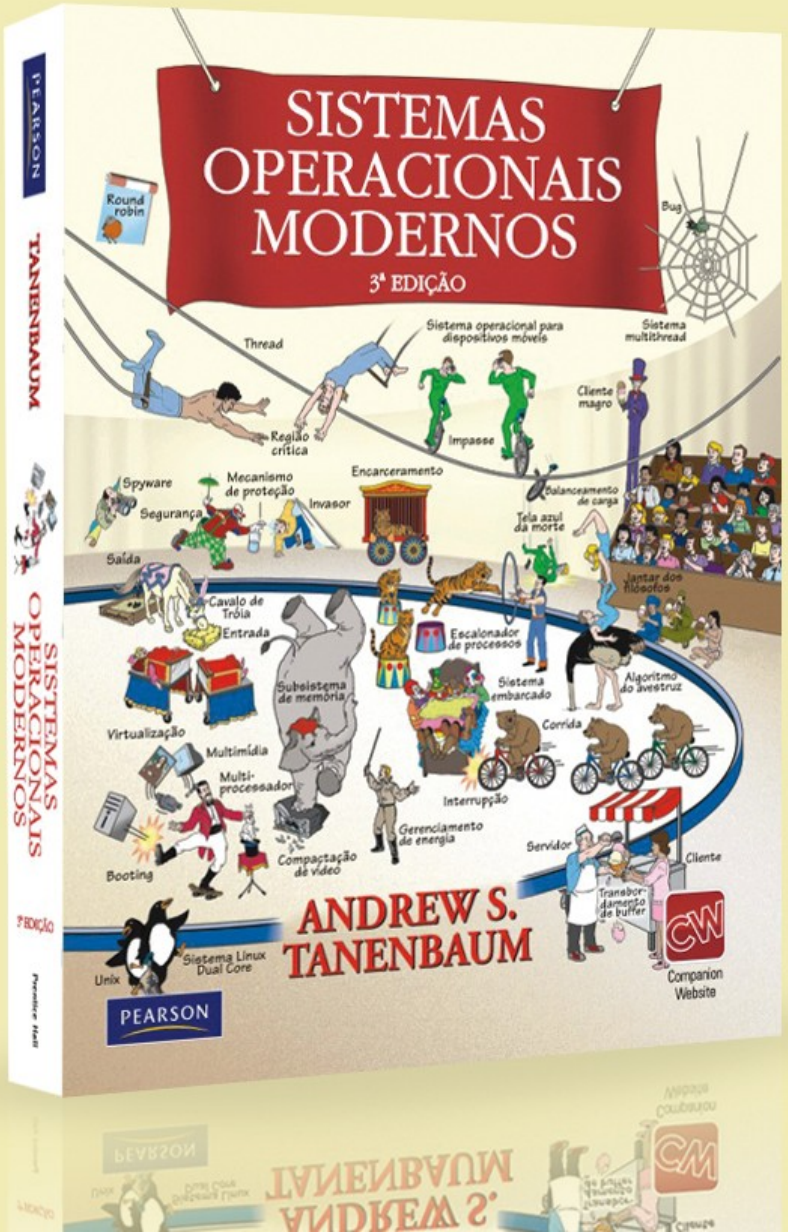
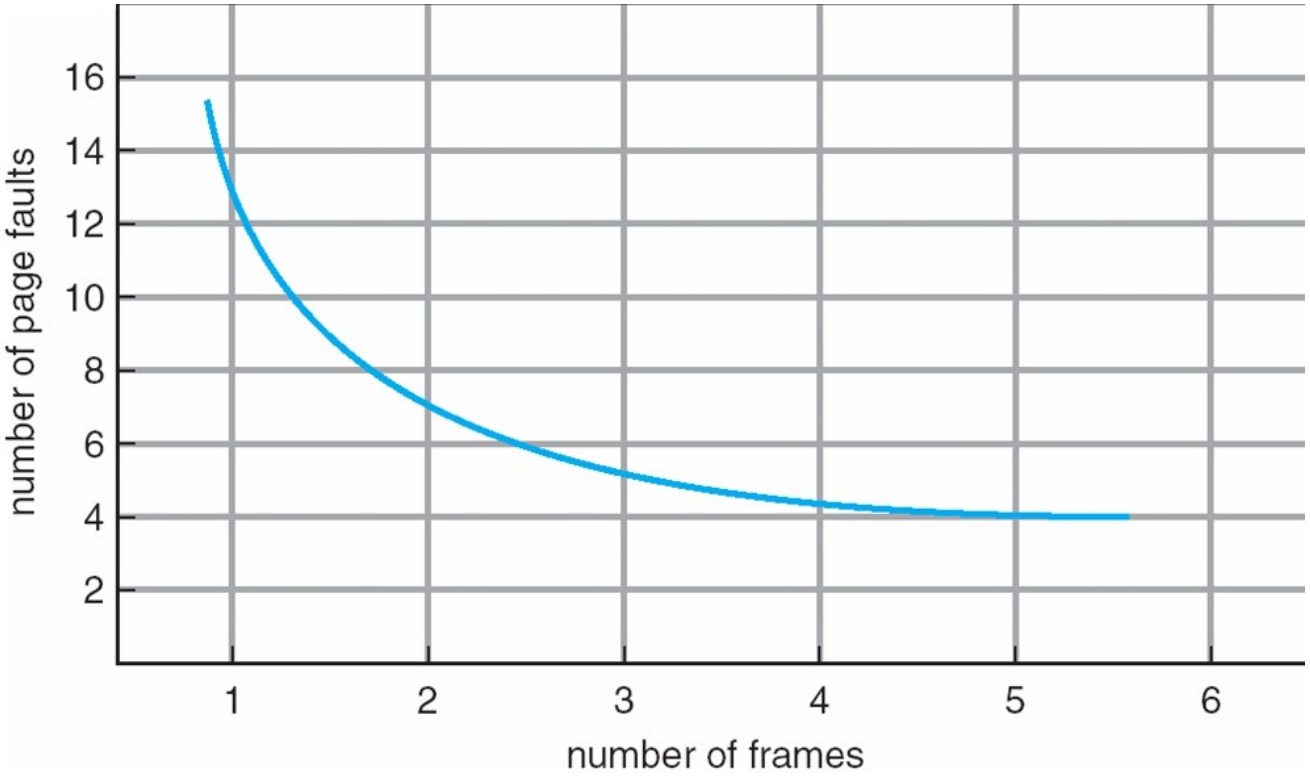
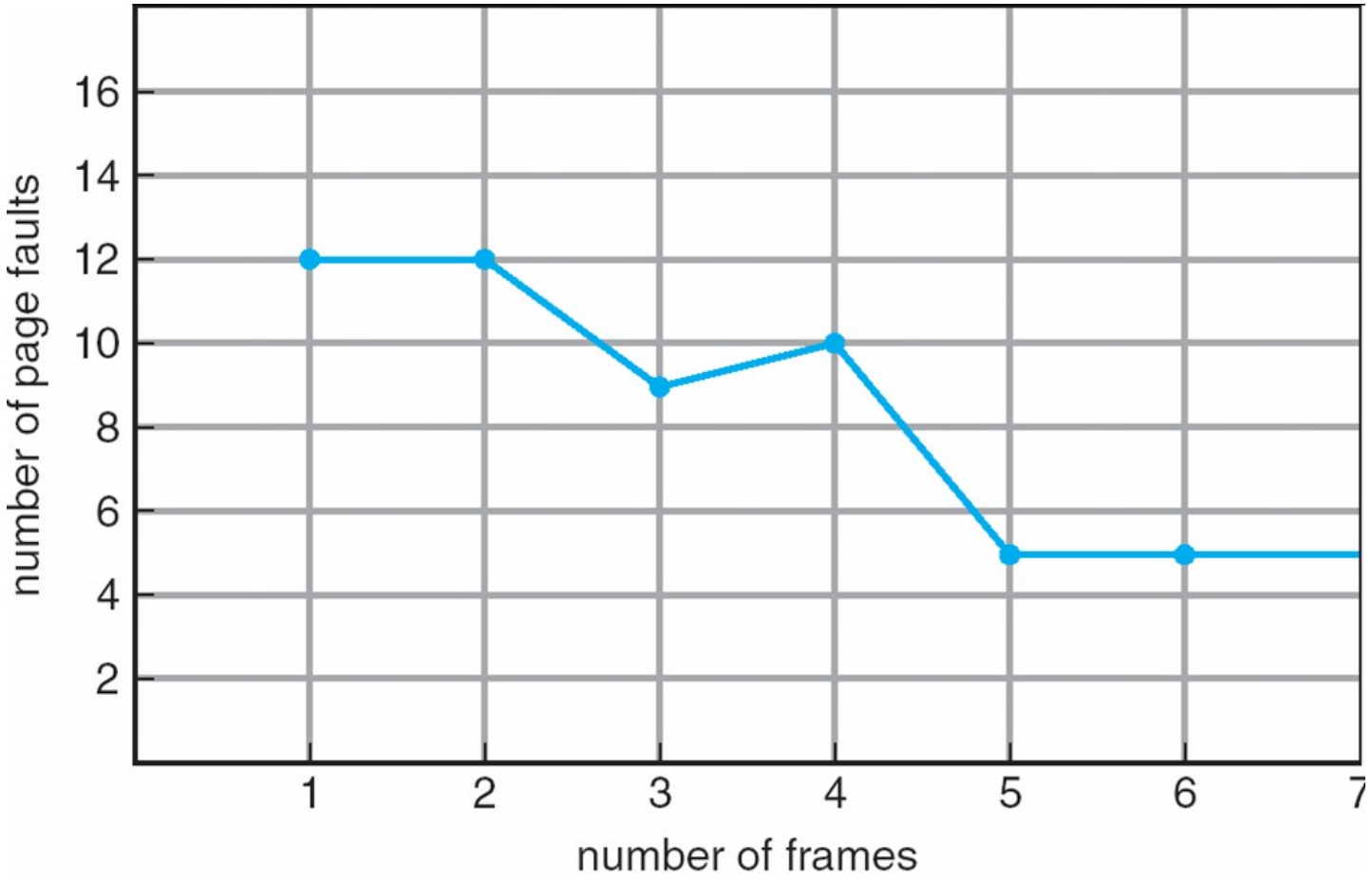


Gráfico Falta de Página X Número de Frames



Anomalia de Belady



Questões de Projeto

- Algumas questões de projeto que devem ser analisadas para um sistema de paginação funcionar, bem:
  - Política de Alocação
  - Controle de carga
  - Tamanho de página
  - Compartilhamento

Política de alocação local versus global

O número de molduras alocadas a cada processo varia com o tempo, e assim o conjunto de trabalho. Que páginas considerar? Local, Global?
Como determinar quantidade por processo? Dividir igualmente, proporcional?
Alocar um número mínimo de paginas para cada processo.

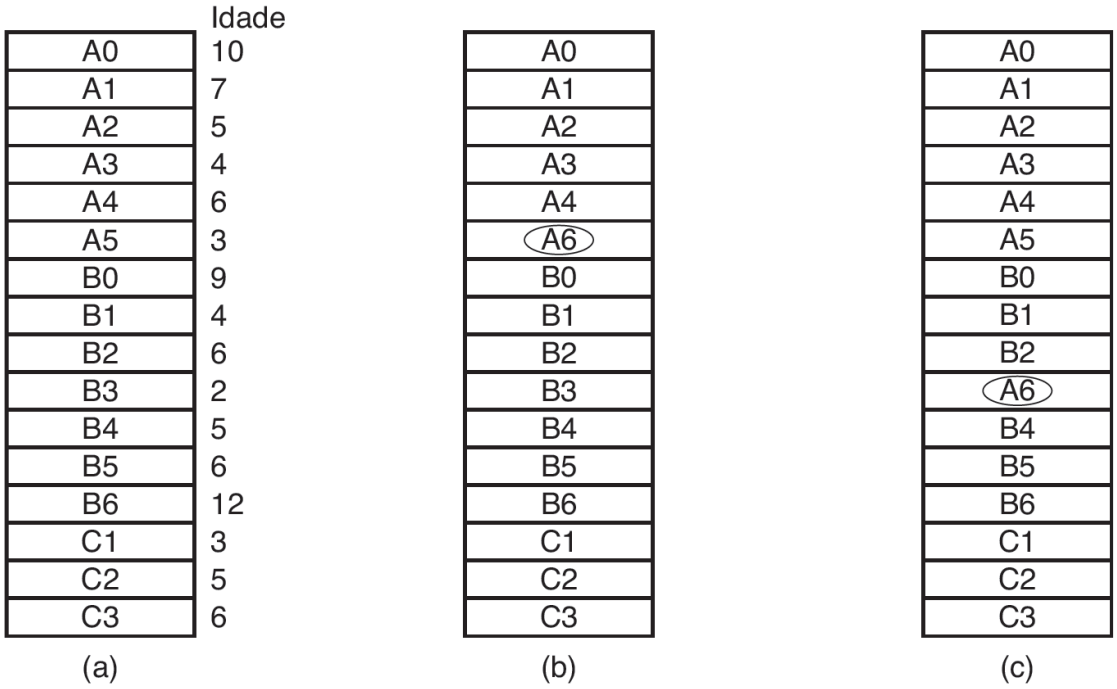


Figura 3.21 Substituição de página local *versus* global. (a) Configuração original. (b) Substituição de página local. (c) Substituição de página global.

PFF – page fault frequency – frequência das faltas de página
O que acontece quando os conjuntos de trabalho de todos os processos combinados excedem a capacidade de memória?
Fazer o que? Swapping? Grau de multiprogramação?

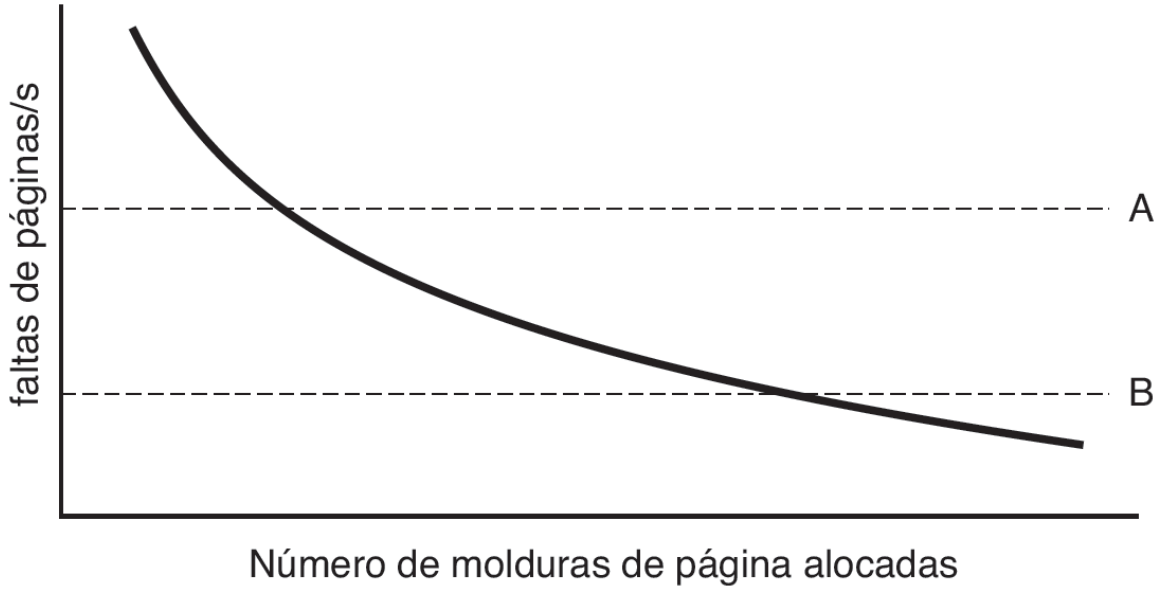
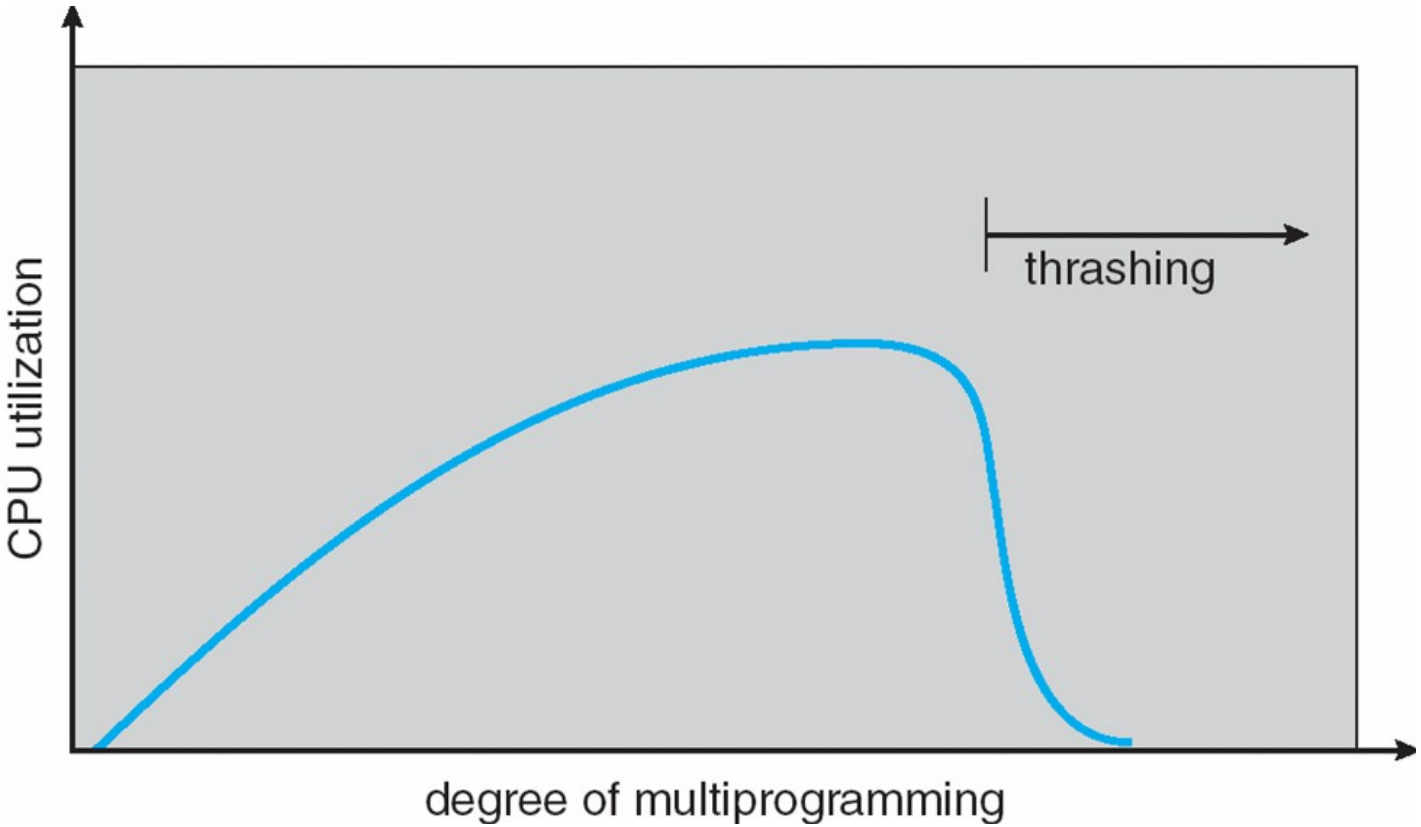


Figura 3.22 Frequência de faltas de página como função do número de molduras de página alocadas.

Thrashing

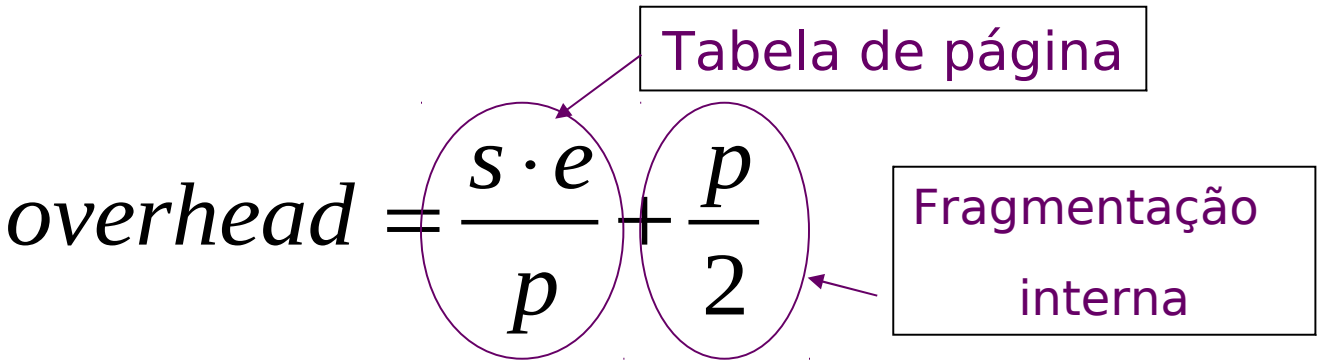


Tamanho de página

- Melhor tamanho depende do balanceamento de fatores conflitantes
- Argumentos favoráveis a páginas pequenas:
  - É provável que um segmento de código, dados ou pilha não ocupe um numero inteiro de páginas
  - Suponha um programa constituído de 8 fases sequenciais de 4KB cada (32KB), se o tamanho da página for <= 4K é preciso apenas 4K em qualquer instante
- Páginas grandes
  - fragmentação interna?
  - Carga? (disco 8KB, 12ms)
  - Tabela de páginas ?
- Pequenas páginas
  - fragmentação interna?
  - Carga? (disco 512bytes, 10ms)
  - Tabela de páginas ?

Tamanho da Página

- Overhead devido a tabela de página e fragmentação



- Onde
  - s = tamanho médio do processo
  - p = tamanho da página
  - e =bytes p/ entrada da tabela

- Ex : s=1MB e= 8bytes ==> 4KB

Tamanho ótimo

$$p = \sqrt{2se}$$

Espaços separados de instrução de dados

Quando EE é suficiente ok, em EE pequenos solução pode ser separar EE em espaço I e espaço D (figura).
O ligador deve saber quando isto é usado. Ambos podem ser paginados.

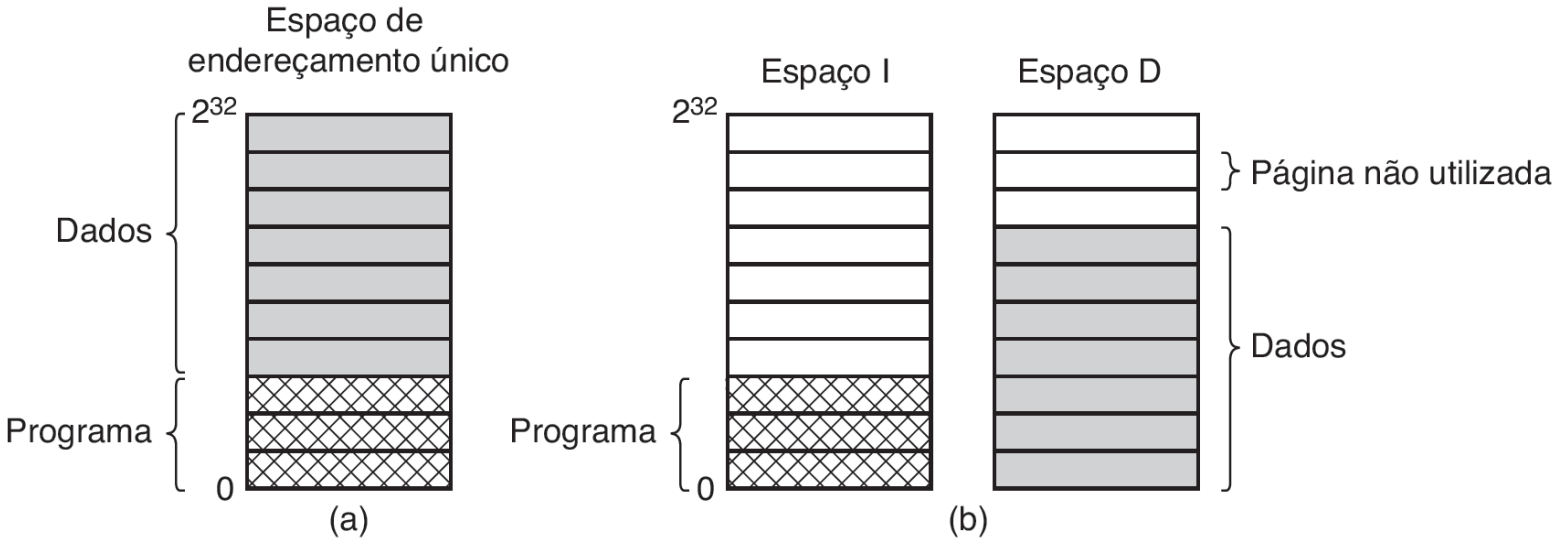


Figura 3.23 (a) Um espaço de endereçamento. (b) Espaços I e D independentes.

Páginas compartilhadas

Que páginas são compartilhadas?
Espaços I e D?
Supor processos A e B executem o mesmo editor (compartilhando pag) é necessário estruturas de dados para manter o controle das páginas compartilhadas.

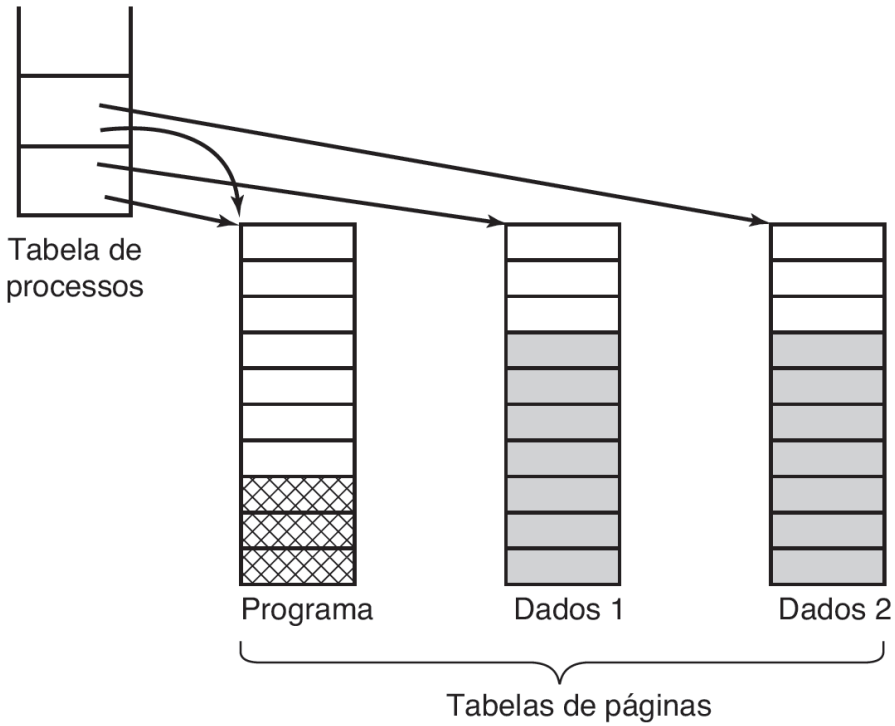
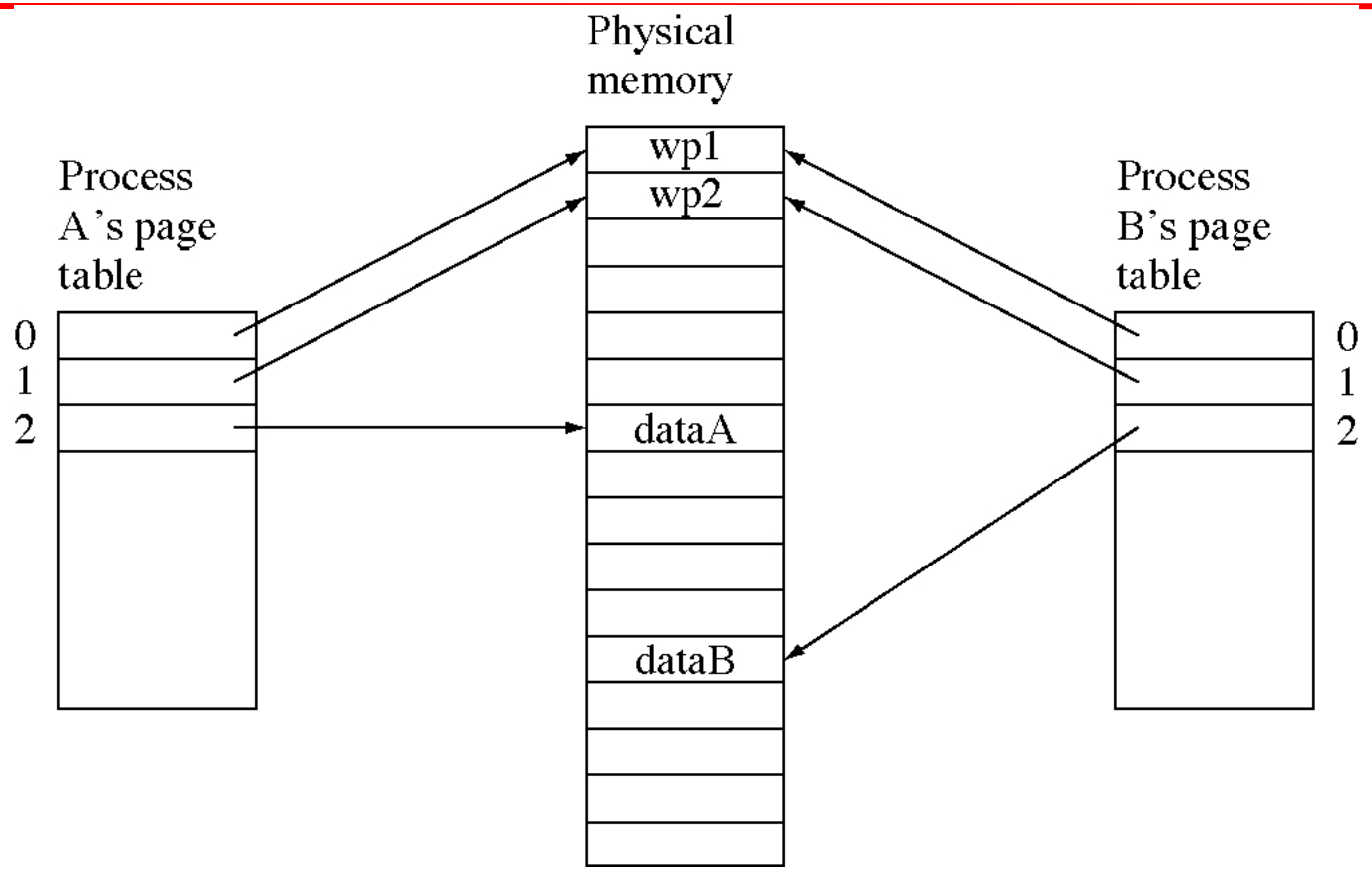


Figura 3.24 Dois processos que compartilham o mesmo programa compartilhando sua tabela de páginas.

Dois processos compartilhando código



Bibliotecas compartilhadas

DLLs, shared object (so) - bibliotecas dinâmicas
Problema – supor que a primeira coisa a ser feita pela primeira função é saltar para o endereço 16 na biblioteca. O compartilhamento faz com que a realocação dinâmica não funcione – quando chamada em P2 deve ir para 12K+16 - flag compilador gera só endereços relativos não absolutos – código independente da posição

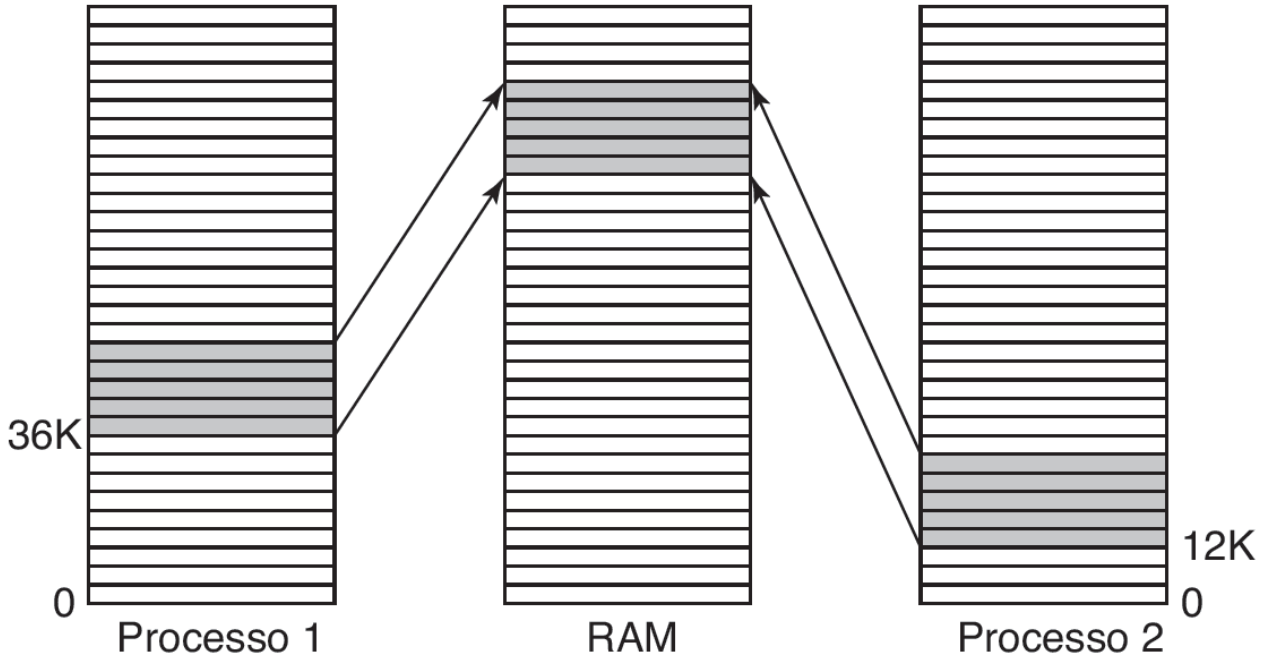
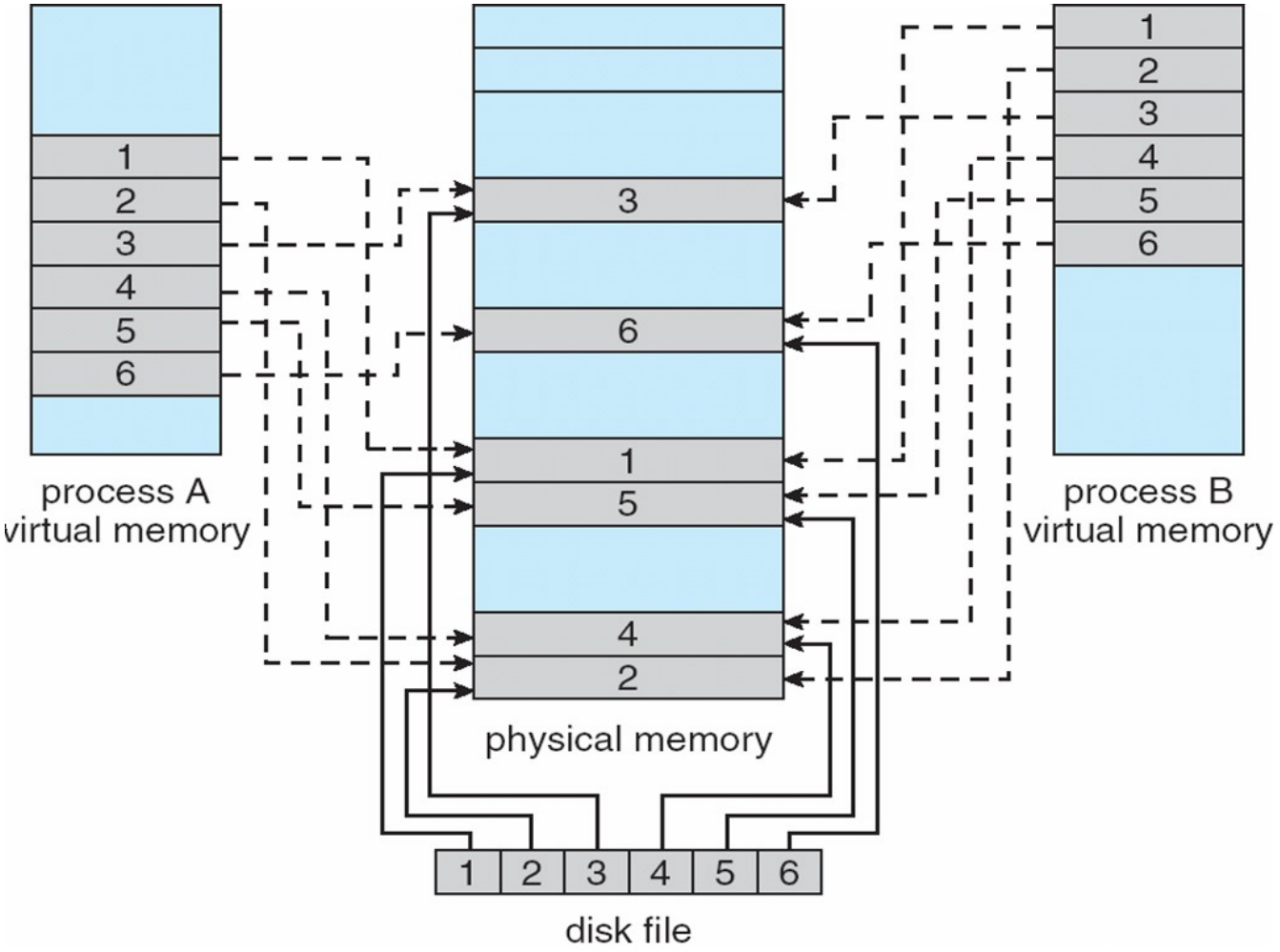


Figura 3.25 Uma biblioteca compartilhada sendo usada por dois processos.

Memoria Mapeada



Questões de Implementação

- Algumas questões práticas de implementação que os projetistas devem conhecer bem:
  - Envolvimento do SO com a paginação
  - Retenção de páginas
  - Memória secundária

MV: envolvimento do SO

- Em um sistema de MV (paginação) o SO deve responder a outros eventos
  - Criação de processo
  - Terminação de processo
  - Despacho de processo

Criação de Processo

- 1. Calcular tamanho do programa (N páginas)
- 2. Alocar N páginas do espaço de swap
- 3. Alocar uma tabela de páginas (na memória do SO) para N entradas
- 4. Inicializar a área de swap
- 5. Inicializar a tabela de páginas: todas as página são marcadas como não presentes
- 6. Registrar a posição na área de swap e da tabela de páginas no descritor do processo

Despacho de Processo

- 1. Invalidar a TLB (uma vez que acontece uma mudança de espaço de endereçamento)
- 2. Carregar o registrador base da tabela de paginas com o endereço da tabela de paginas deste processo.

Terminação de Processo

- 1. liberar memória usada pela tabela de páginas
- 2. liberar espaço de disco na área de swap
- 3. liberar os frames que o processo estava usando

Paginação por demanda: Performance

Taxa de Faltas $0 \leq p \leq 1.0$

se $p = 0$ sem faltas

se $p = 1$, toda referência é uma falta

Tempo de Acesso Efetivo (TAE)

TAE = (1 – *p*) x acesso memória
+ *p* (overhead da falta
+ [swap-out página]
+ swap-in página
+ overhead reínicio)

Exemplo: 200ns tempo de acesso, 1 falta a cada 1000referências,
Média tempo da falta 8milis → TAE = 8,2 micros

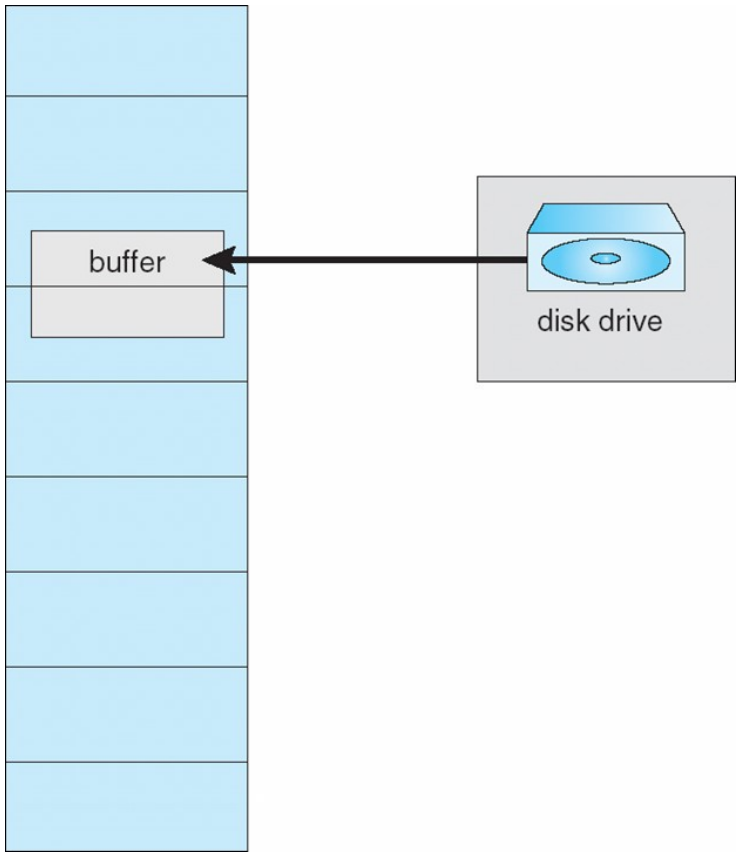
Tratamento da falta de página

- O hardware cria uma cilada para o núcleo, salvando o contador do programa na pilha.
- Uma rotina em código é iniciada para salvar o conteúdo dos registradores de uso geral e outras informações voláteis.
- O sistema operacional descobre a ocorrência de uma falta de página e tenta descobrir qual página virtual é necessária.
- Uma vez conhecido o endereço virtual que causou a falta da página, o sistema verifica se esse endereço é válido e se a proteção é consistente com o acesso.

- Se existe alguma moldura livre aloca. Senão seleciona uma moldura para realizar a **troca de página**.
- Se a moldura da página selecionada estiver suja, a página é escalonada para ser transferida para o disco e será realizado um chaveamento de contexto.
- Quando a moldura da página estiver limpa, o sistema operacional buscará o endereço em disco onde está a página virtual solicitada e escalonará uma operação para trazê-la.
- Quando a interrupção de disco indicar que a página chegou na memória, as tabelas de páginas serão atualizadas para refletir sua posição, e será indicado que a moldura de página está normal.

- A instrução que estava faltando é recuperada para o estado em que se encontrava quando começou, e o contador de programa é reiniciado a fim de apontar para aquela instrução.
- O processo em falta é escalonado, o sistema operacional retorna para a rotina, em linguagem de máquina, que o chamou.
- Esta rotina recarrega os registradores e outras informações de estado e retorna ao espaço de usuário para continuar a execução como se nada tivesse ocorrido.

Retenção de Páginas na Memória



Memória secundária

Alocação de espaço em disco para área de swap:
1. reserva área do tamanho do processo – end. Mantido na TP
2. não reservar nada, alocar sob demanda – manter o end de cada pag

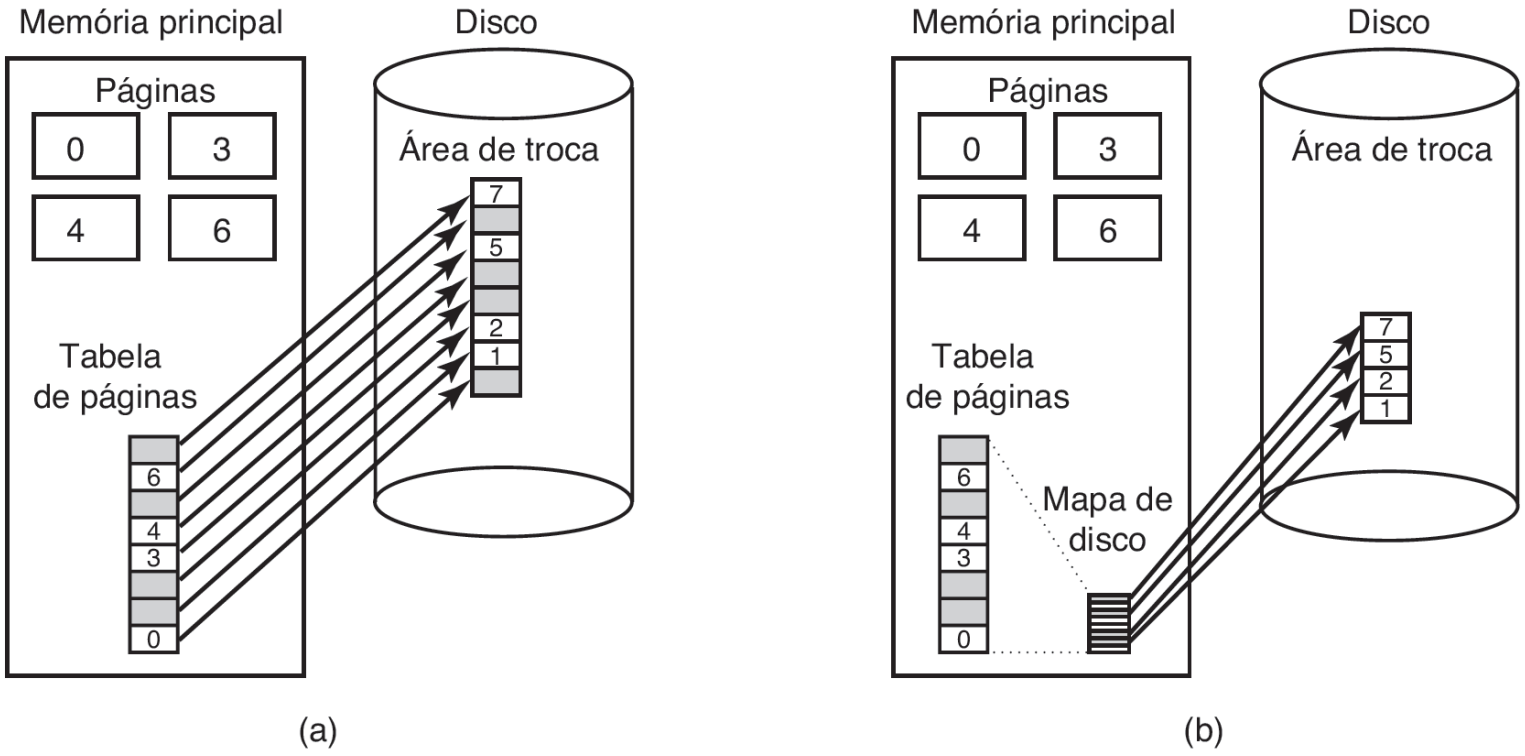


Figura 3.27 (a) Paginação para uma área de troca estática. (b) Recuperando páginas dinamicamente.

Separação da política e mecanismo

Sistema de gerenciamento de memória é dividido em três partes:

- Um manipulador de MMU de baixo nível.
- Um manipulador de falta de página que faz parte do núcleo.
- Um paginador externo executado no espaço do usuário.

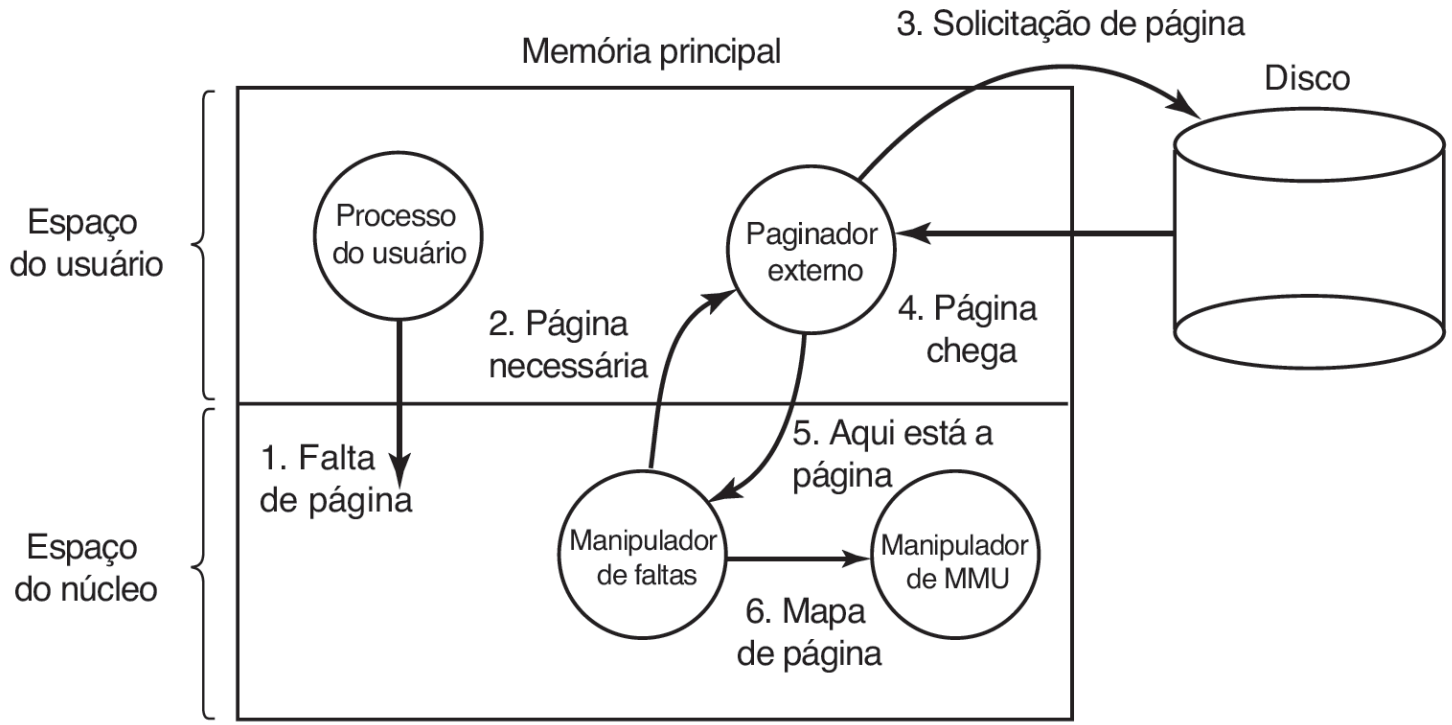


Figura 3.28 Tratamento de falta de página com um paginador externo.

Estrutura do Programa

Program structure

`Int[128,128] data;`

Each row is stored in one page

Program 1

```
for (j = 0; j <128; j++)
    for (i = 0; i < 128; i++)
        data[i,j] = 0;
```

128 x 128 = 16,384 page faults

Program 2

```
for (i = 0; i < 128; i++)
    for (j = 0; j < 128; j++)
        data[i,j] = 0;
```

128 page faults

Segmentação

MV discutida até agora é unidimensional (0..n), situações onde 2 ou mais EE podem ser melhor.

Um compilador tem muitas tabelas que são construídas conforme a compilação ocorre, possivelmente incluindo:

- O código-fonte sendo salvo para impressão (em sistema em lotes).
- A tabela de símbolos – os nomes e atributos das variáveis.
- A tabela com todas as constantes usadas, inteiras e em ponto flutuante.
- A árvore sintática, a análise sintática do programa.
- A pilha usada pelas chamadas de rotina dentro do compilador.

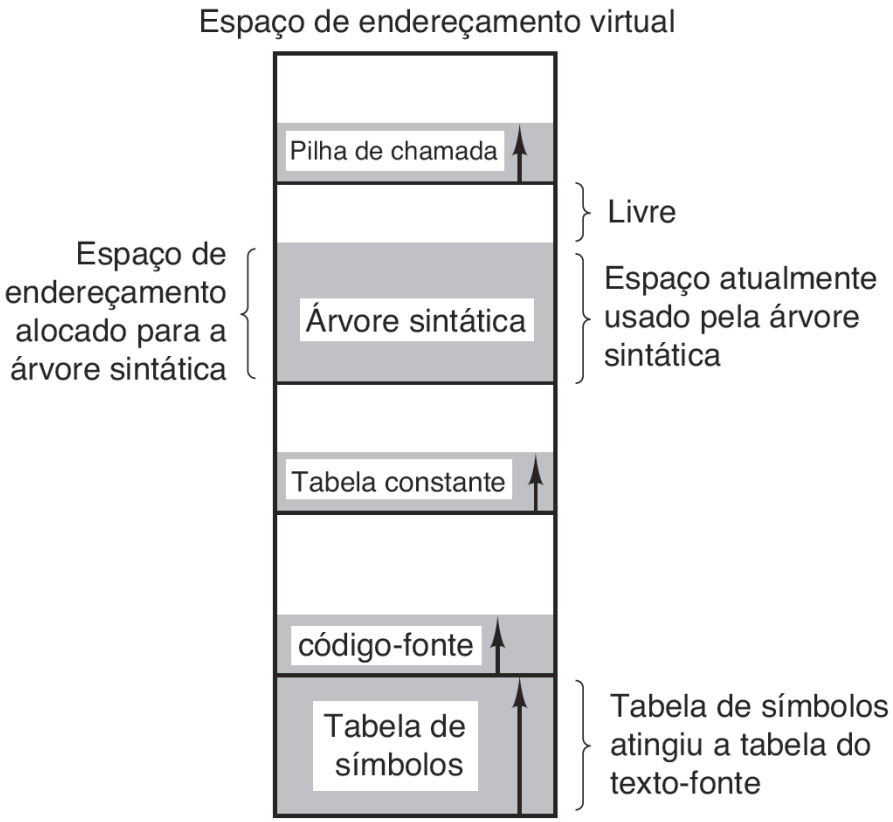


Figura 3.29 Em um espaço de endereçamento unidimensional com tabelas crescentes, uma tabela poderá atingir outra.

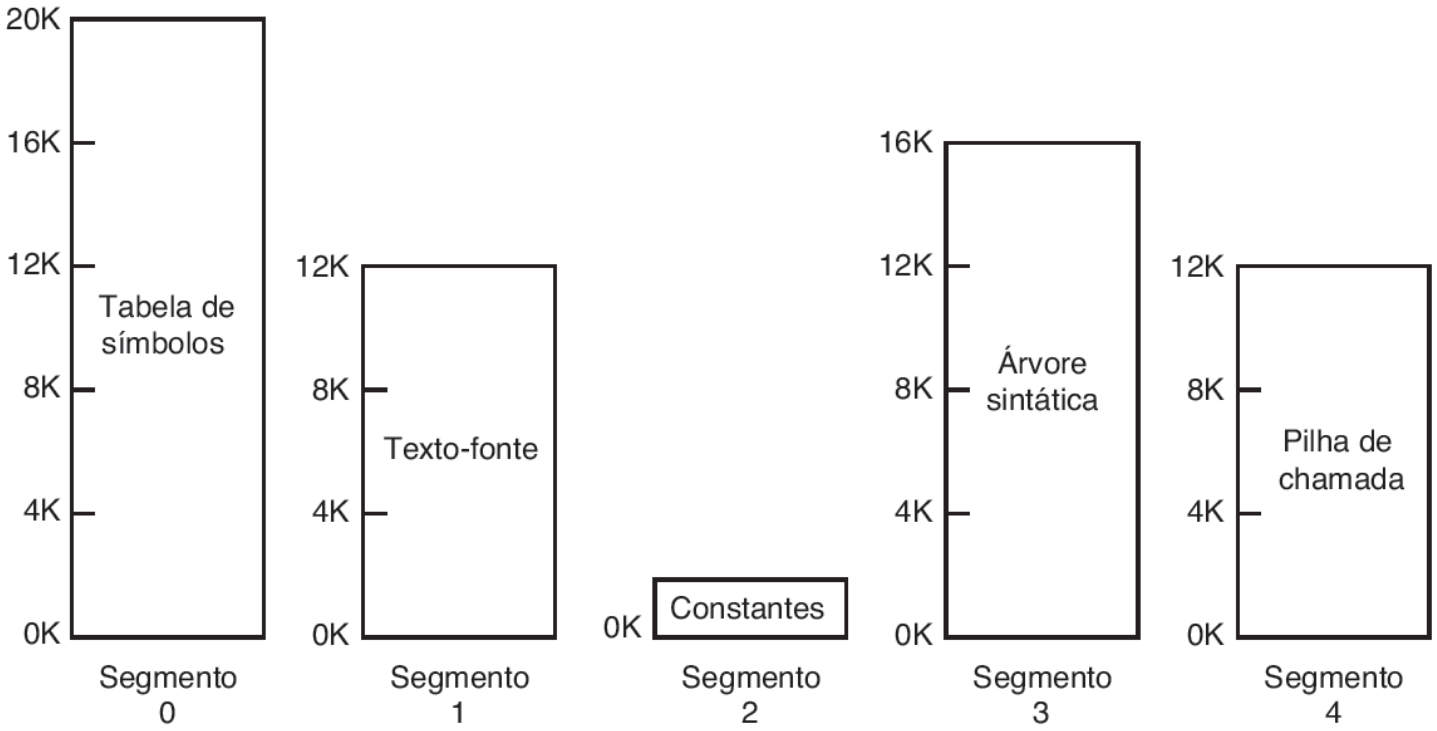
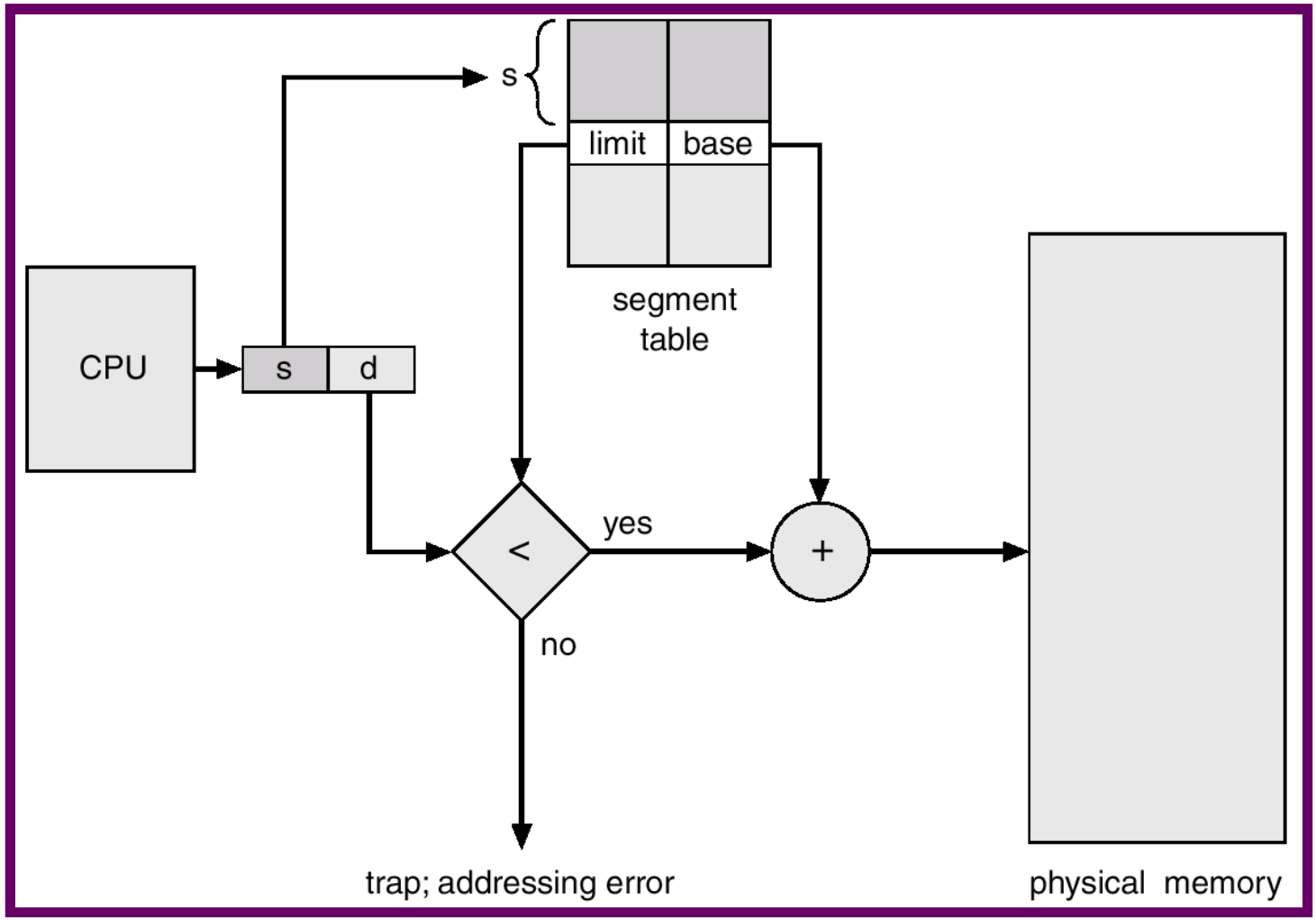


Figura 3.30 Uma memória segmentada permite que cada tabela cresça ou encolha de modo independente das outras tabelas.

Hardware de Segmentação



Implementação da segmentação pura

Consideração	Paginação	Segmentação
O programador precisa saber que essa técnica está sendo usada?	Não	Sim
Há quantos espaços de endereçamento linear?	1	Muitos
O espaço de endereçamento total pode superar o tamanho da memória física?	Sim	Sim
Rotinas e dados podem ser distinguidos e protegidos separadamente?	Não	Sim
As tabelas cujo tamanho flutua podem ser facilmente acomodadas?	Não	Sim
O compartilhamento de rotinas entre os usuários é facilitado?	Não	Sim
Por que essa técnica foi inventada?	Para obter um grande espaço de endereçamento linear sem a necessidade de comprar mais memória física	Para permitir que programas e dados sejam divididos em espaços de endereçamento logicamente independentes e para auxiliar o compartilhamento e a proteção

Tabela 3.3 Comparação entre paginação e segmentação.

Segmentação com paginação: MULTICS

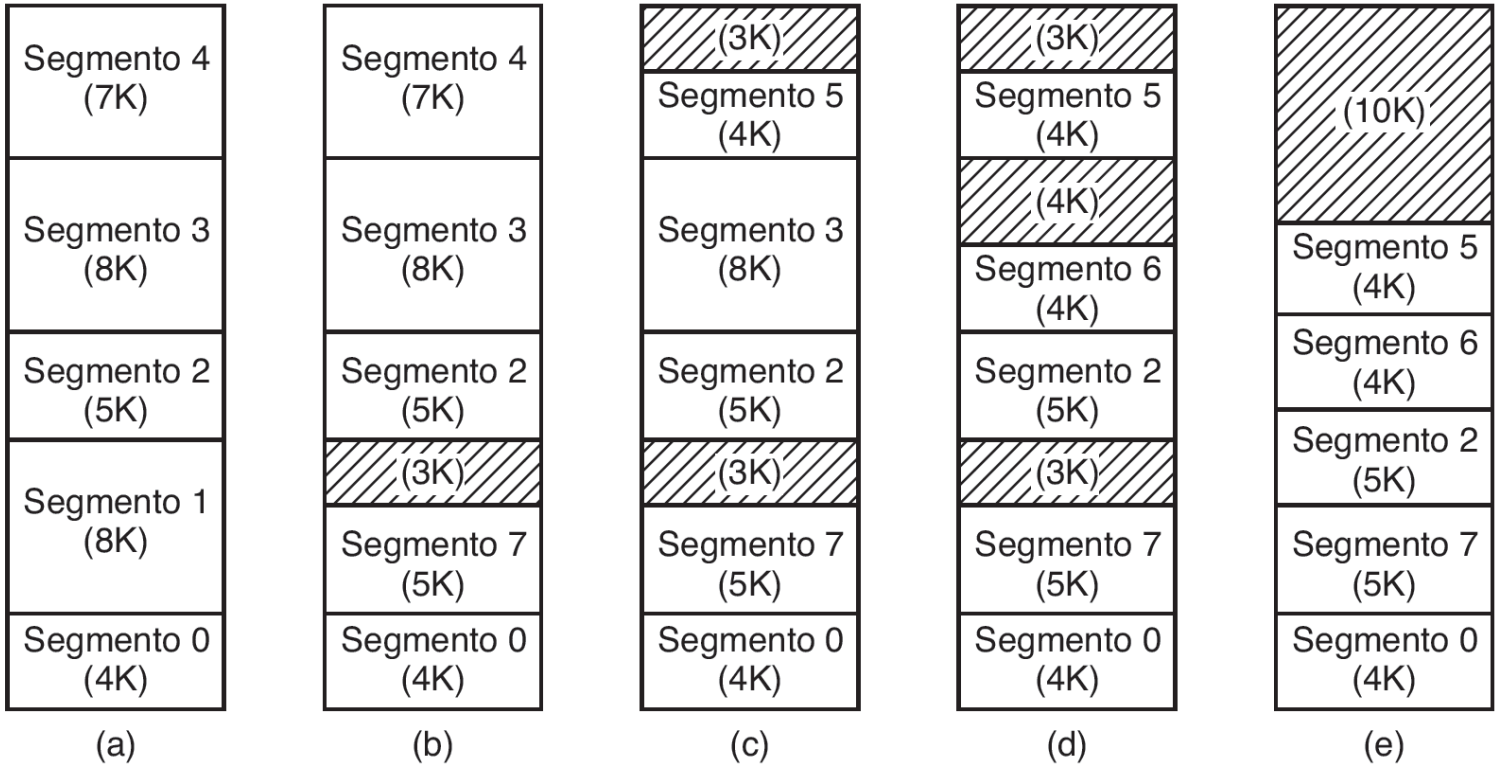


Figura 3.31 (a)–(d) Desenvolvimento de checkerboarding. (e) Remoção do checkerboarding por compactação.

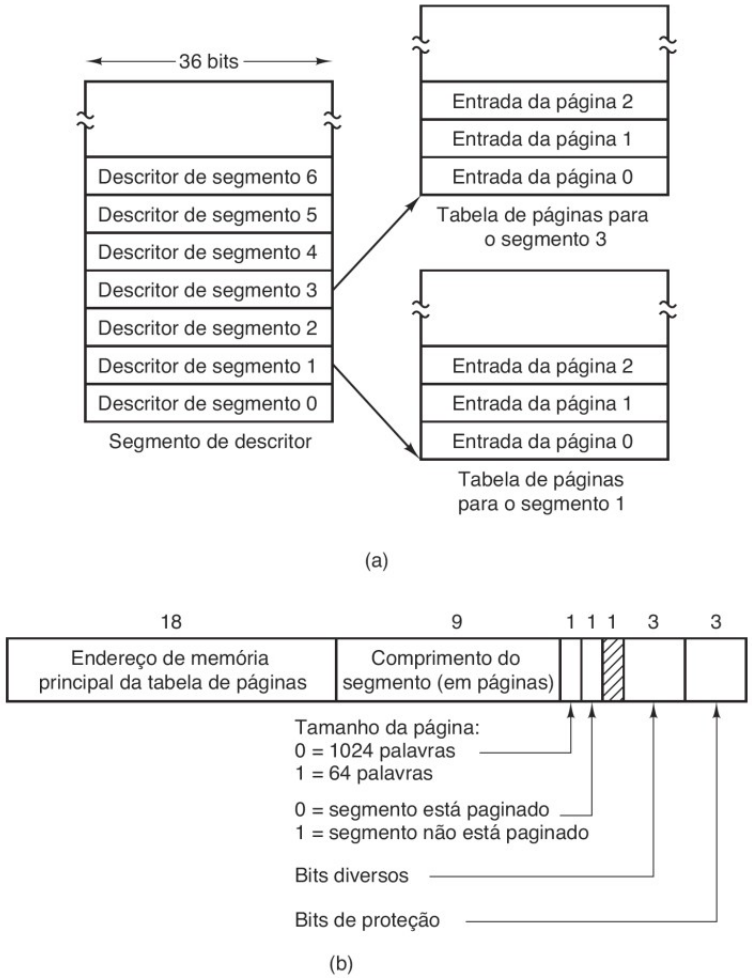


Figura 3.32 Memória virtual MULTICS. (a) O descritor de segmento aponta para tabelas de páginas. (b) Um descritor de segmento Os números são o comprimento do campo.

Quando ocorre uma referência à memória, o seguinte algoritmo é executado:

- O número de segmento é usado para encontrar o descritor desse segmento.
- Realiza-se uma verificação para ver se a tabela de páginas do segmento está na memória.
  - Se não estiver, ocorre uma falta do segmento.
  - Se houver uma violação na proteção, uma falta ocorre.

- A entrada da tabela de páginas para a página virtual requerida é examinada.
  - Se a página não estiver na memória, ocorre uma falta de página.
  - Se estiver na memória, o endereço da memória principal do início da página é extraído da entrada da tabela de páginas.
- O deslocamento é adicionado à página de origem para gerar o endereço da memória principal onde a palavra está localizada.
- A leitura ou registro podem ser realizados.

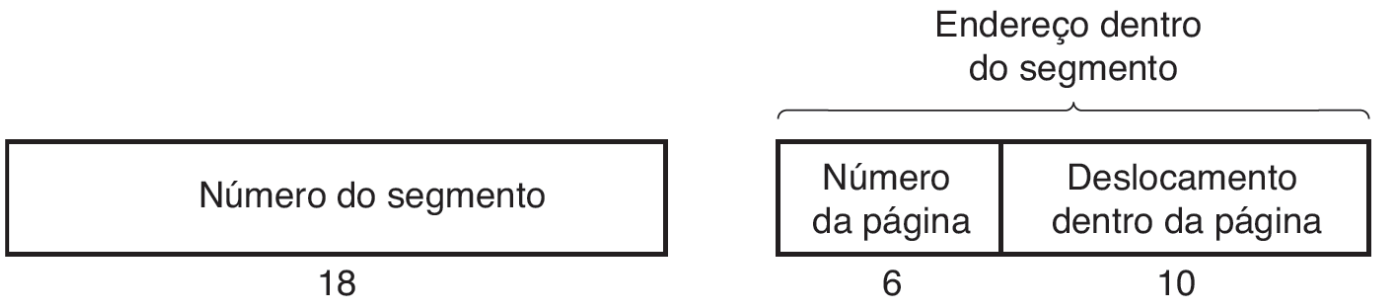


Figura 3.33 Um endereço virtual MULTICS de 34 bits.

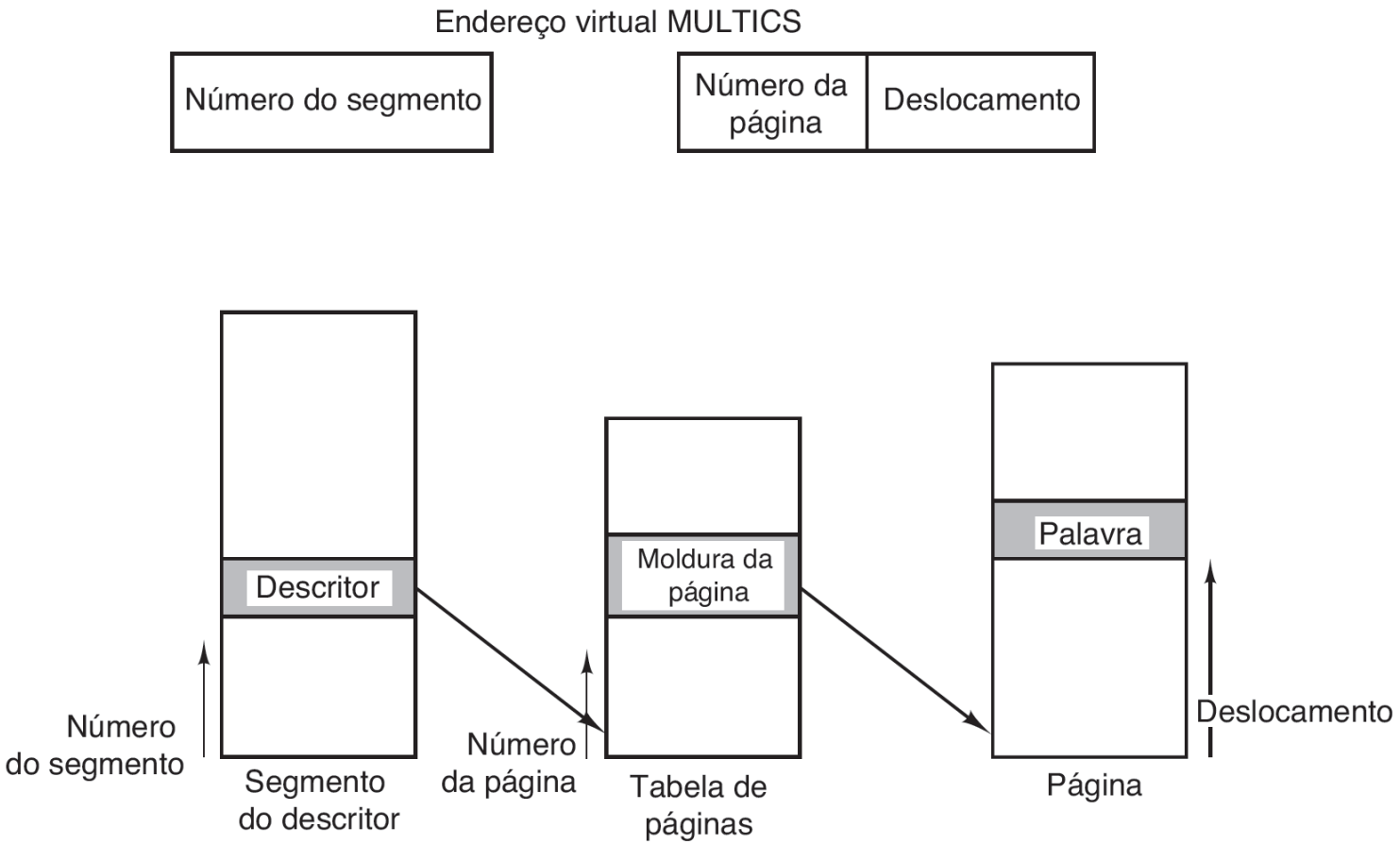


Figura 3.34 Conversão de um endereço de duas partes do MULTICS em um endereço de memória principal.

Segmentação com paginação: o Pentium

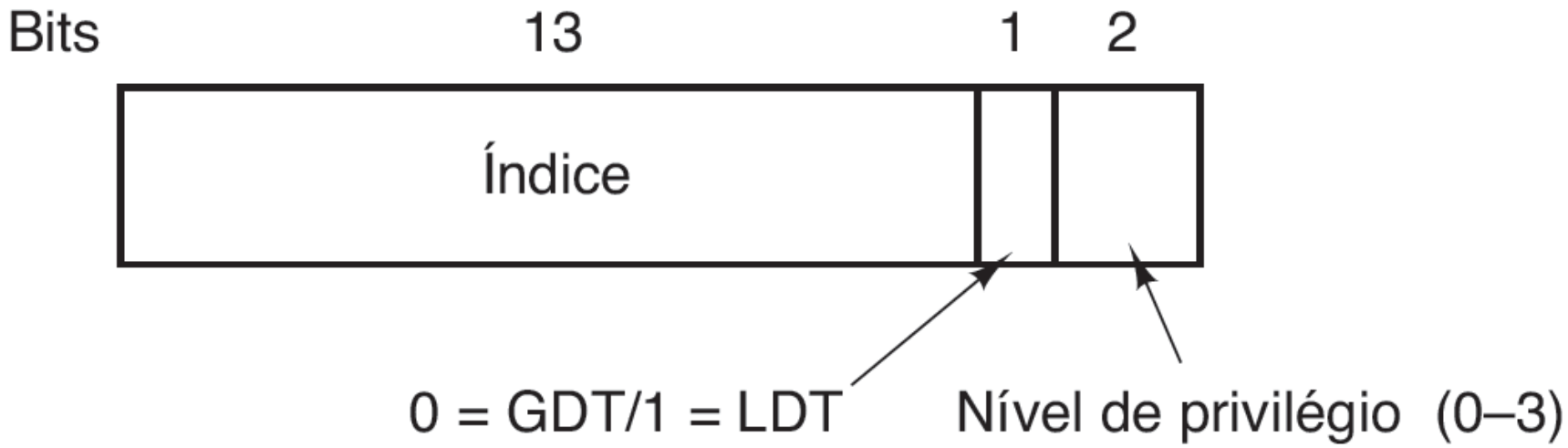


Figura 3.36 Seletor do Pentium.

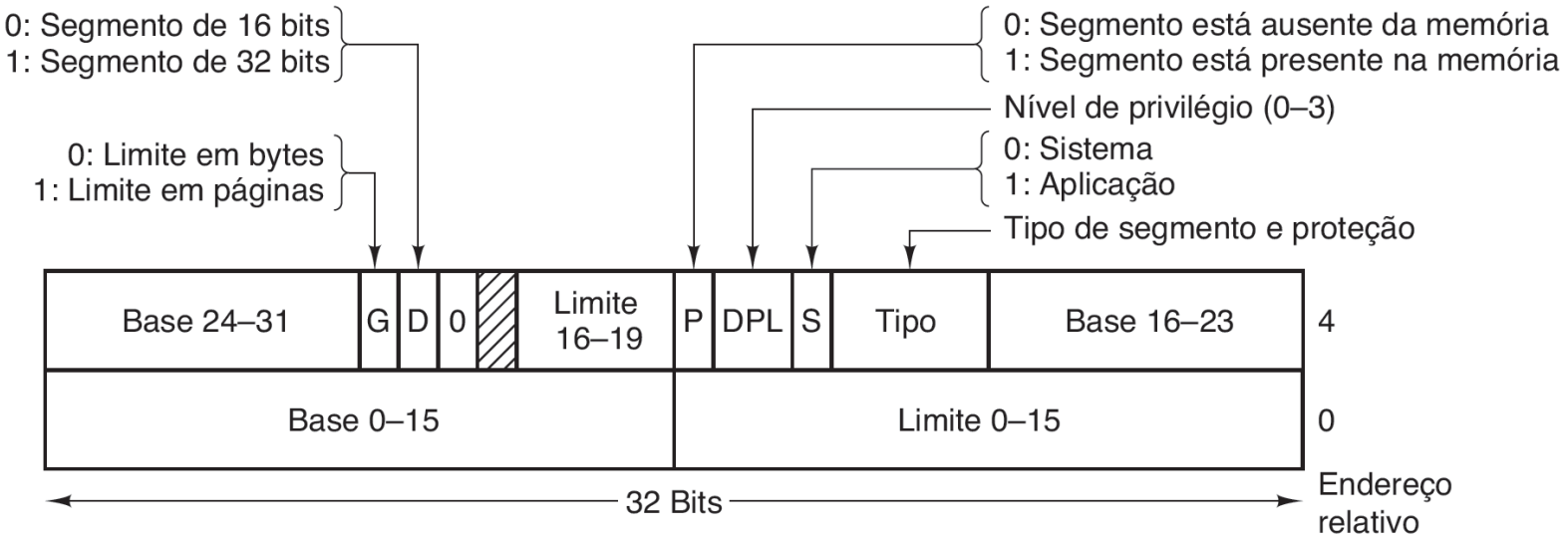


Figura 3.37 Descritor de segmento de código do Pentium. Os segmentos de dados são ligeiramente diferentes.

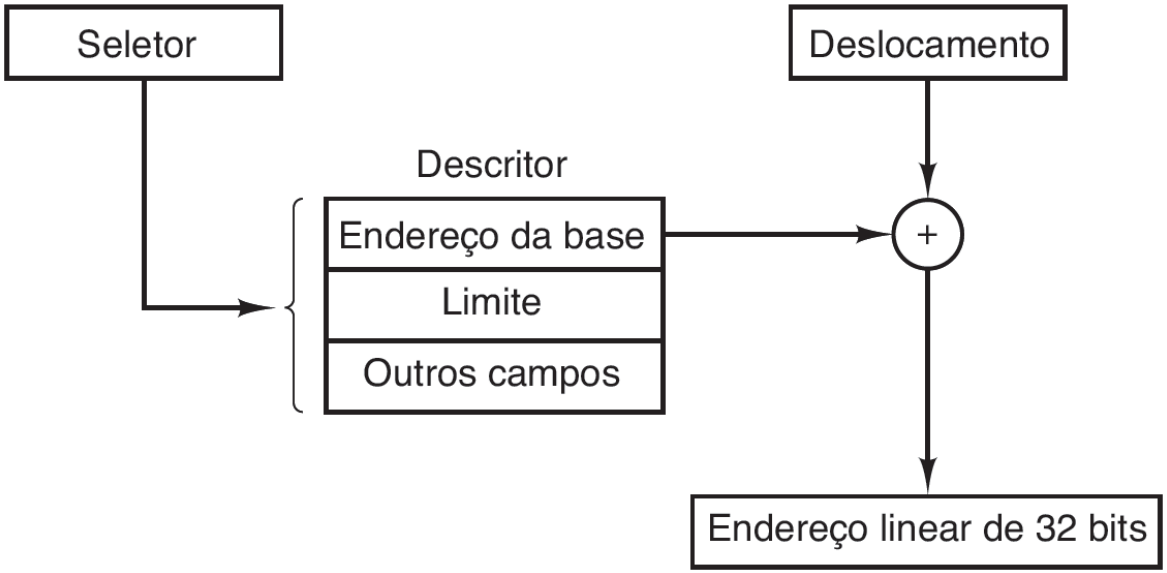
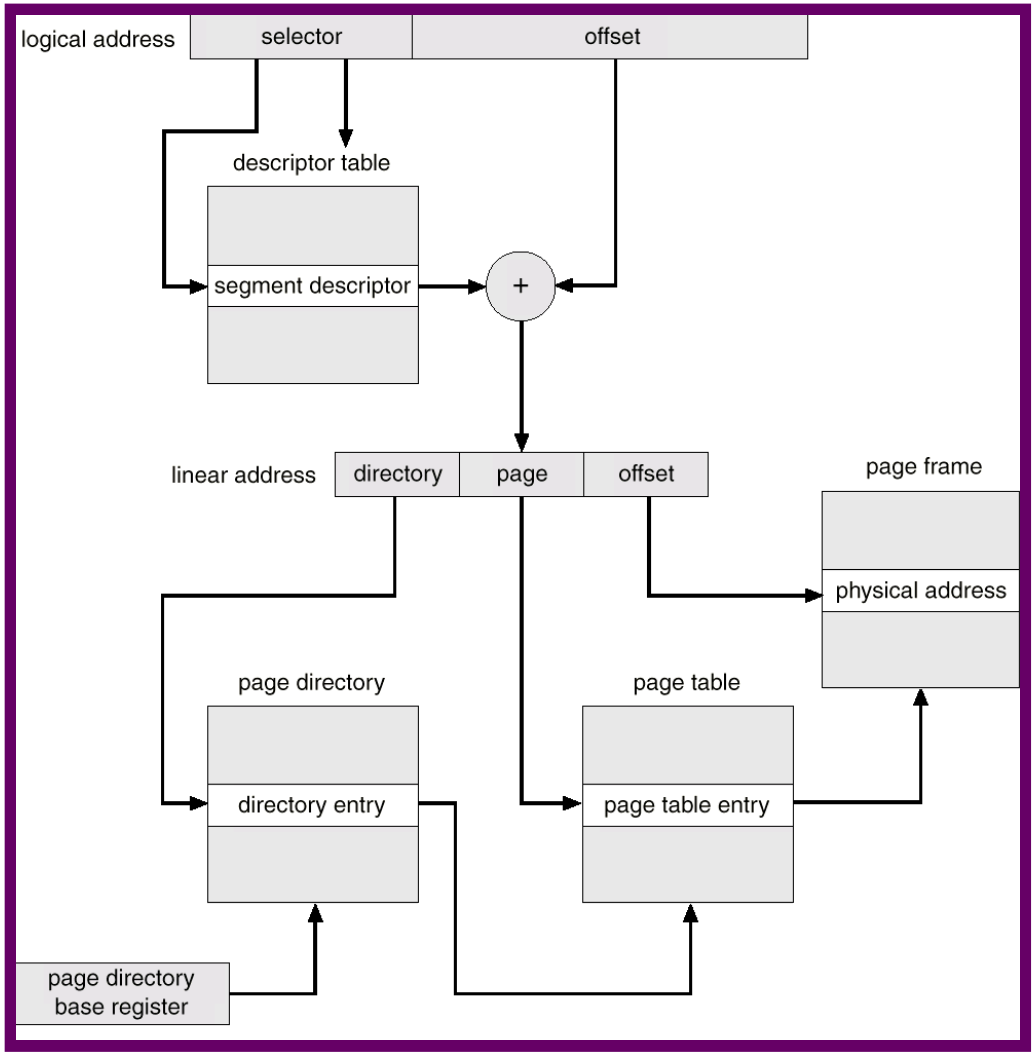


Figura 3.38 Conversão de um par (de seletores, deslocamentos) em um endereço linear.

Intel 386


logical address
selector
offset
descriptor table
segment descriptor
+
linear address
directory
page
offset
page frame
physical address
page directory
directory entry
page table
page table entry
page directory
base register

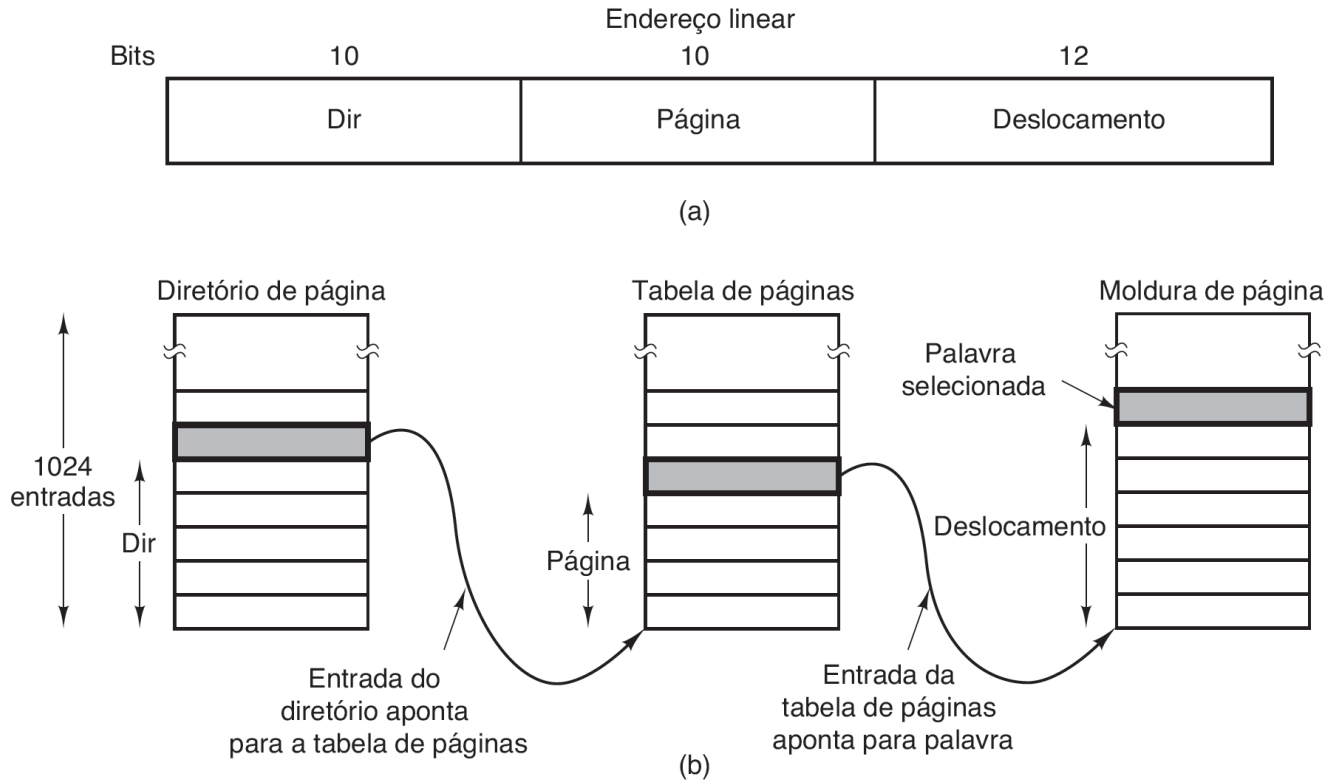


Figura 3.39 Mapeamento de um endereço linear em um endereço físico.

Gerenciamento de memória no Linux

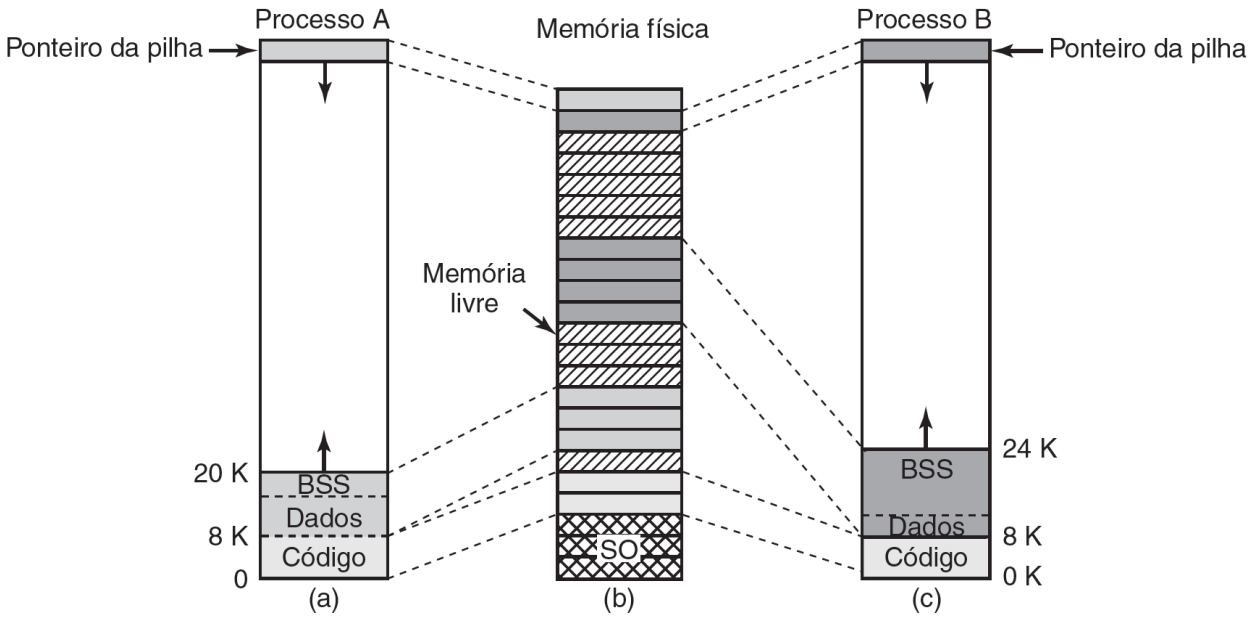


Figura 10.8 (a) Espaço de endereçamento virtual do processo *A*. (b) Memória física. (c) Espaço de endereçamento virtual do processo *B*.

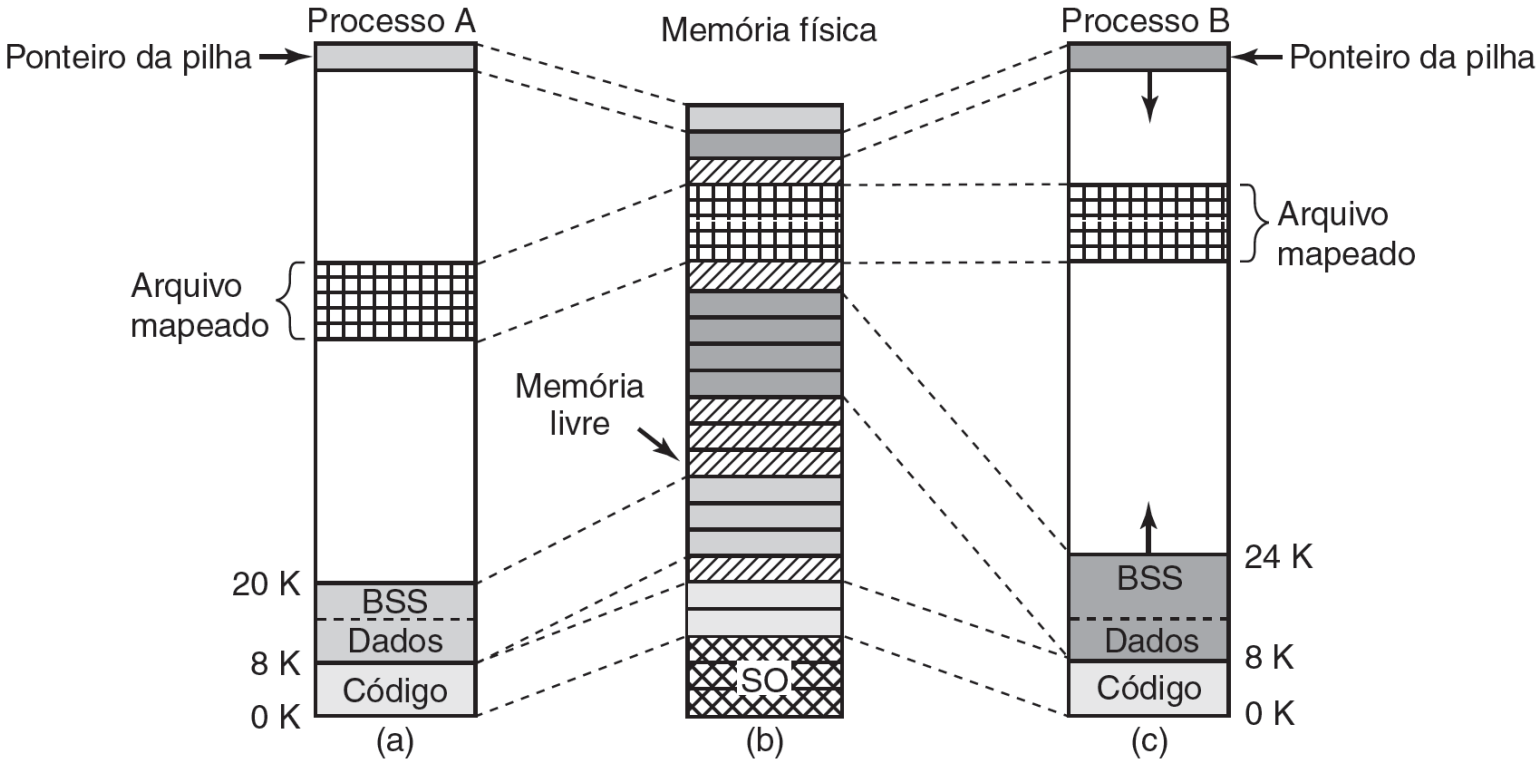


Figura 10.9 Dois processos podem compartilhar um arquivo mapeado.

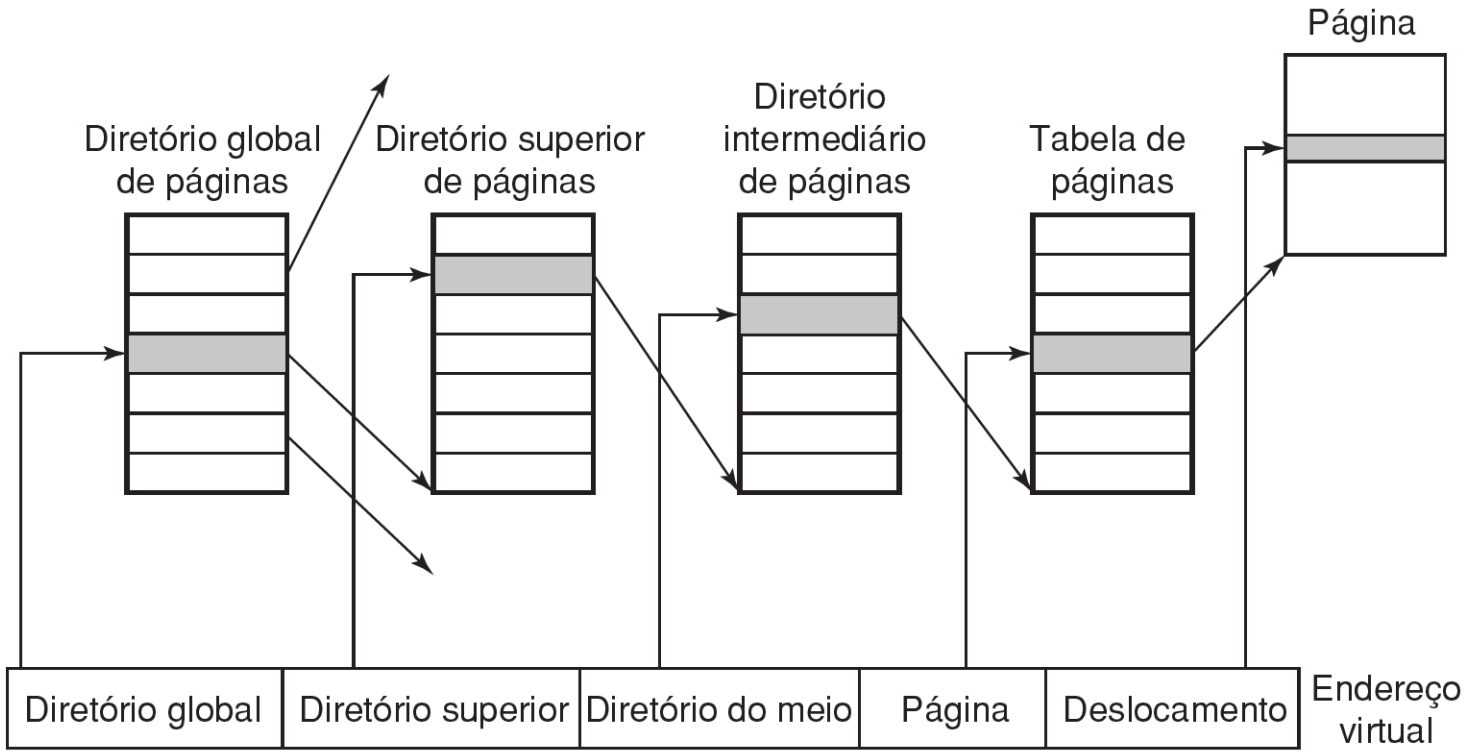


Figura 10.11 O Linux utiliza tabelas de páginas de quatro níveis.