Sistemas Operacionais

Gerenciamento de Memória

Paginação e Segmentação

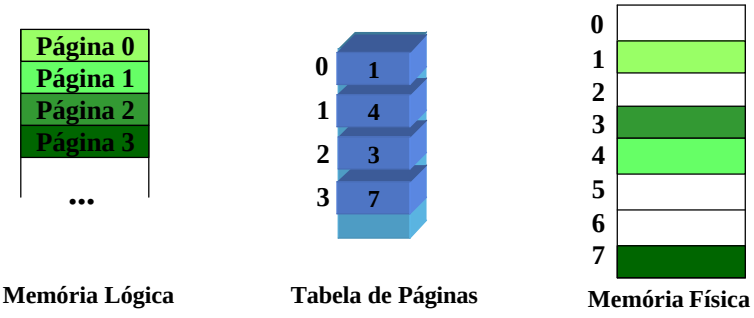
SO - Gerenciamento de Memória

- Paginação
  - *Hardware*
  - Tamanho das Páginas
  - Modelo
  - Estrutura da Tabela de Páginas
  - Paginação em Vários Níveis
  - Compartilhamento de Páginas
- Segmentação
  - *Hardware*
  - Implementação de Tabela de Segmentos
  - Proteção e Compartilhamento
  - Fragmentação
- Segmentação com Paginação

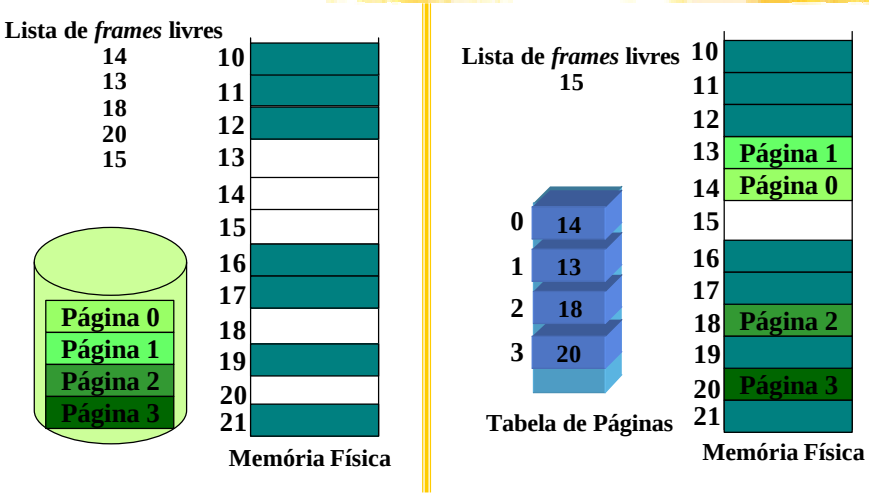
Paginação

- Outra solução (além da compactação) para o problema da fragmentação externa:
  - permitir que o espaço de endereçamento lógico seja não contíguo, facilitando a alocação dos buracos da memória
  - esse mecanismo evita o problema idêntico que ocorre com a memória secundária, onde a compactação tem alto custo
- A memória física é dividida em partes de tamanho fixo, chamadas de blocos (*frames* ou molduras)
- A memória lógica é dividida em partes do mesmo tamanho, chamadas de páginas
- A memória secundária também é dividida em partes do mesmo tamanho dos blocos da memória principal

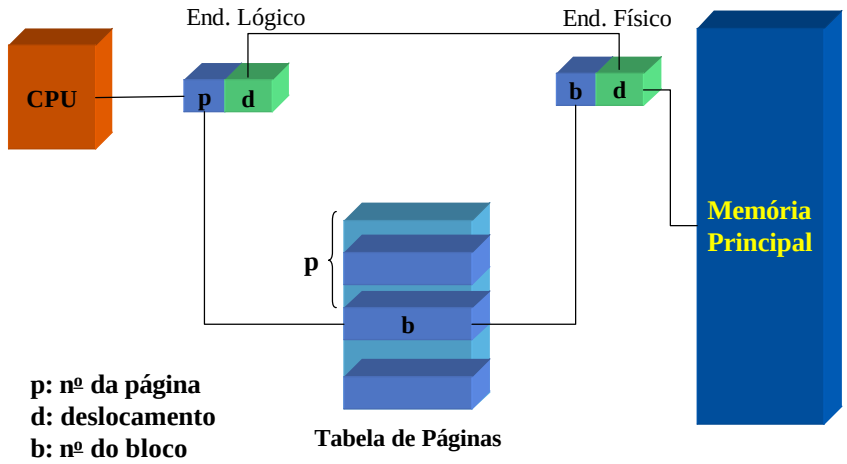
Paginação: modelo



Paginação: processo novo



Paginação: *hardware* de suporte



Paginação: tamanho das páginas

- O tamanho da página (e do bloco) é definido pelo *hardware*
- Ele é normalmente uma potência de 2, veja o por quê:

Se tamanho do espaço lógico: 2^a
e tamanho da página: 2^b

Paginação

- Forma de atribuição de endereço em tempo de execução ➔ usar paginação é semelhante a usar uma tabela de registradores base, um para cada bloco
- Fragmentação externa resolvida
- Fragmentação interna:
  - tam. pag. = 2048 bytes
  - processo = 72766 bytes

  } 35 pag + 1086 bytes

  fragmentação = 2048 - 1086 = 922 bytes
- Para processos de tamanho aleatório:
  - Fragmentação média = (tamanho de pagina) / 2

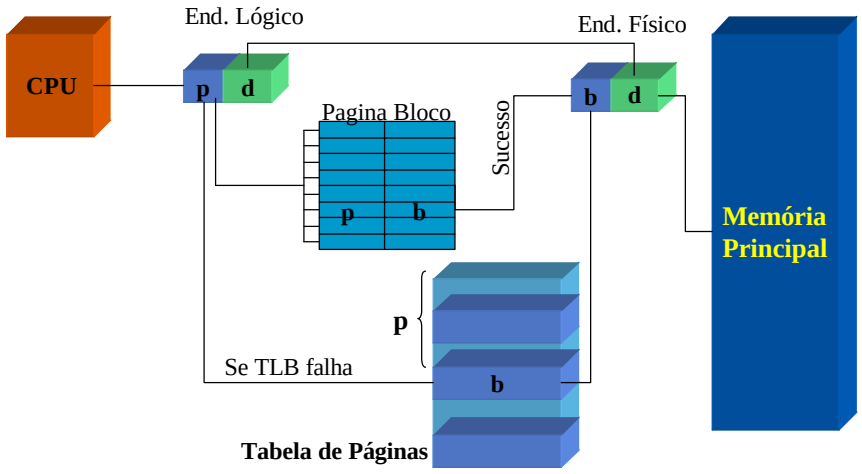
Paginação

- Assim,
  - páginas menores, menor fragmentação interna
- Entretanto,
  - páginas maiores, menores os gastos com as tabelas de páginas e mais eficientes as transferências (E/S) em discos
- Separação entre a memória física e a visão que o usuário tem da memória (espaço único e contíguo)
- SO mantém cópia (apontador) das tabelas de páginas de cada processo ➔ maior tempo gasto em uma mudança de contexto

Estrutura da Tabela de Páginas

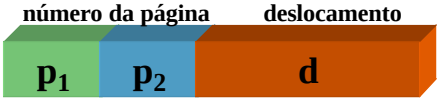
- Pode ser representada em *hardware* por:
  - conjunto de registradores dedicados - nº pequeno de entradas
  - registrador de tabela de páginas (RTB) - 2 acessos à memória
  - registradores associativos (TLB - *Translation Look-Aside Buffers*) - cache em *hardware* de pequena capacidade e busca rápida
    - Pesquisa: 20 ns e Memória: 100 ns
    - 80% acerto ➔ t.med. acesso = 0.8*120 + 0.2*220 = 140 ns
    - 98% acerto ➔ t.med. = 0.98*120 + 0.02*220 = 122 ns
    - Intel 80486: 32 reg. associativos e taxa de sucesso 98%

Estrutura da Tabela de Páginas

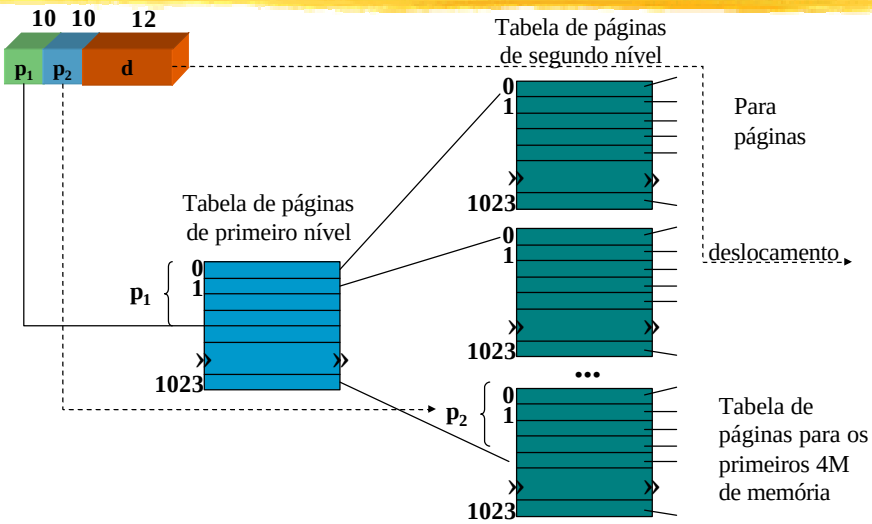


Paginação: vários níveis

- Espaços de endereçamento lógicos muito grandes (2^{32} a 2^{64} bytes) ➔ tabelas enormes...
- Exemplo:
  - Espaço endereçamento lógico: 32 bits
  - Tamanho de páginas: 4k (2^{12})
    - $2^{32}/2^{12}$ = 1 M (2^{20}) entradas
    - 1 M entradas de 4 bytes = Tabelas de 4 M
- Solução: dividir a tabela de páginas em várias partes
  - Ex.:



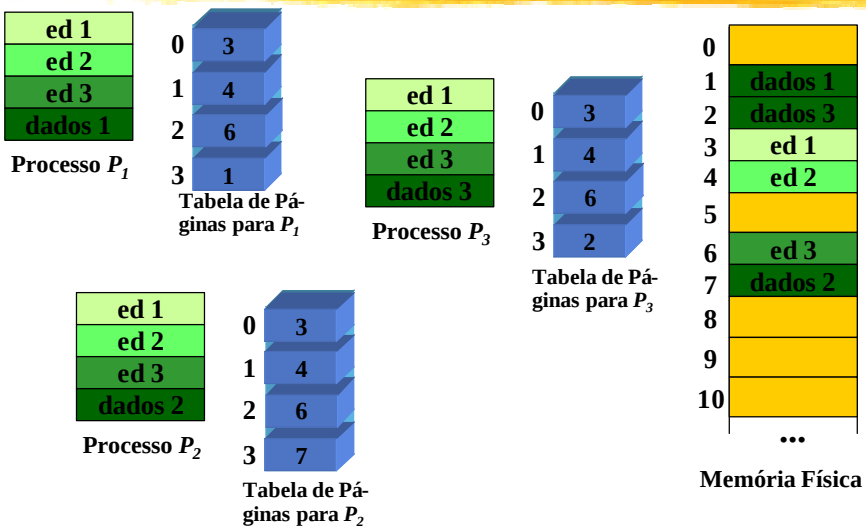
Paginação: vários níveis



Paginação: páginas compartilhadas

- Outra vantagem do mecanismo de paginação é a possibilidade de compartilhamento de código.
- Exemplo:
  - 40 usuários usando um editor de textos ➔ 150 k de código + 50 k de dados
    - Total: 8000 k
- Compiladores, Sistemas de Janelas, Sistemas de Banco de Dados ➔ códigos que nunca modificam a si próprios durante sua execução
  - Usando compartilhamento de páginas: 2150 k no total

Paginação: páginas compartilhadas

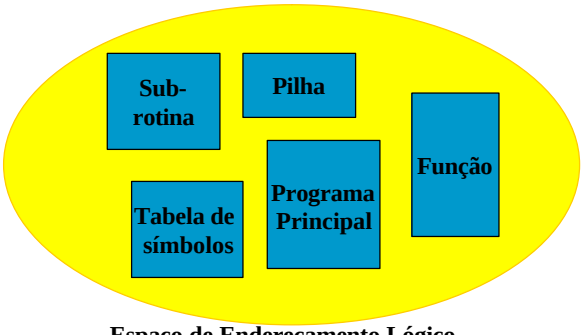


Paginação: páginas compartilhadas

- Suponha que os processos P_1 e P_2 estejam compartilhando páginas de um editor de texto
  - Problemas:
  - Se o escalonador remover P_1 da memória, P_2 terá que trazer de voltas todas as páginas que compartilhava de P_1
  - Se P_1 terminar, é preciso determinar se suas páginas ainda estão em uso, de forma que seu espaço em disco não seja liberado
- Para evitar ter que buscar em todas as tabelas de páginas se uma página é compartilhada ou não (o que é caro!) ➔ montam-se estruturas de dados especiais para tratar das páginas compartilhadas

Segmentação

- Com a paginação, ocorre a separação entre a visão da memória pelo usuário e a memória física ➔ Qual é a visão que o usuário tem da memória?



Espaço de Endereçamento Lógico

sub-rotinas
procedimentos
funções
módulos

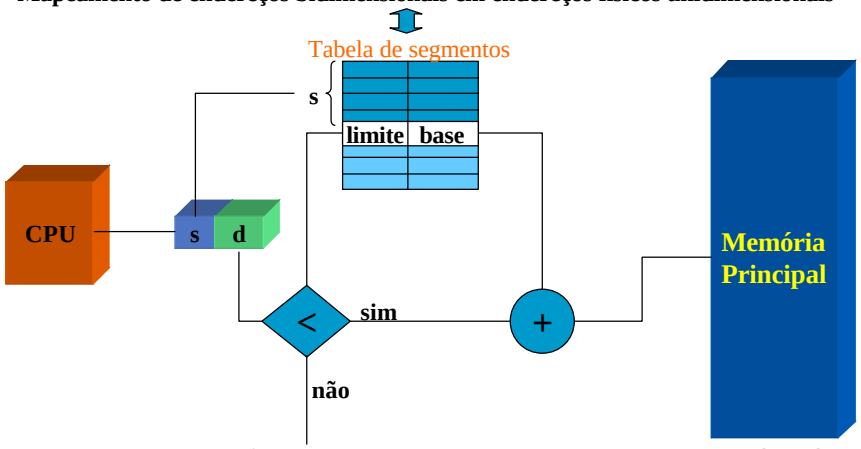
Tabelas
Vetores
Pilhas
Listas
Variáveis

Segmentação

- Cada segmento pode ter um tamanho definido pelo seu propósito
  - Elementos dentro do segmento são identificados pelo deslocamento (*offset*) a partir do início (5ª entrada da tabela, 3ª instrução da função)
- Segmentação é um esquema de gerência de memória que dá suporte a esta visão do usuário:
  - o espaço de end. lógico é uma coleção de segmentos
  - cada segmento tem um nome e um tamanho
  - cada endereço é especificado por dois valores: nome (número) do segmento e a posição (deslocamento) no segmento (*offset*) (<número, posição>)
  - Compilador constrói automaticamente os segmentos
  - Carregador aloca (determina) o número de cada segmento

Segmentação: *hardware*

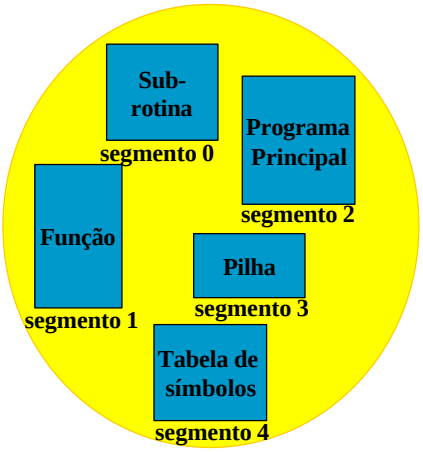
Mapeamento de endereços bidimensionais em endereços físicos unidimensionais



Interrupção de *software*; erro de endereçamento

Memória Física

Segmentação: *hardware*



Espaço de Endereçamento Lógico

Tabela de segmentos

	Limite	Base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

Memória Física

Implementação de Tabelas de Segmentos

- Assim como a tabela de páginas, a tabela de segmentos pode ser armazenada tanto em registradores rápidos quanto na memória
- Se um programa consiste em muitos segmentos, deve-se usar a memória. Um registrador RTS (reg. de tabela de segmento) aponta para a tabela
- Como o tamanho da tabela varia, usas-se um registrador RTTS (reg. de tamanho da tabela de segmentos) para indicar o comprimento da mesma
- Pode-se também usar um conjunto de registradores associativos para armazenar as entradas mais recentemente usadas

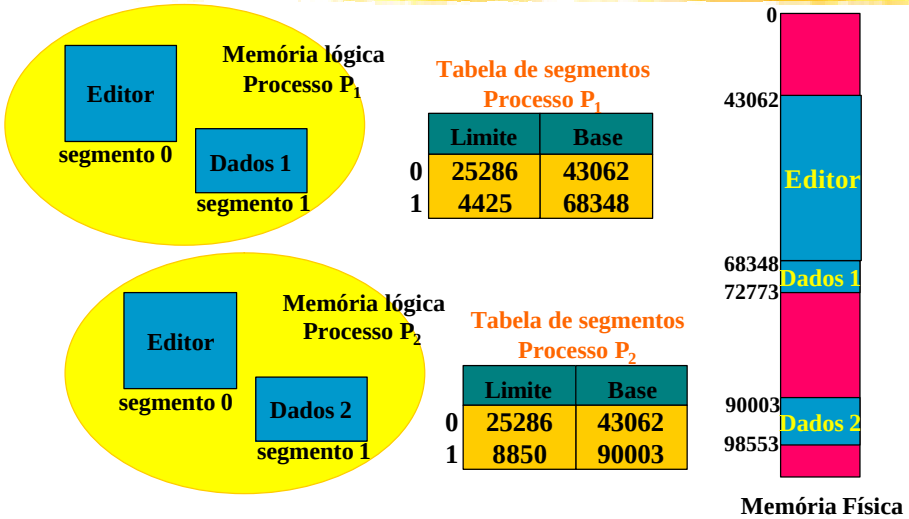
Proteção

- Direitos de acesso diferentes a cada segmento
- Como os segmentos representam porções semanticamente distintas de um programa, todos os acessos a este segmento serão usados da mesma maneira
- Segmento de dados ➔ leitura/escrita/execução
- Segmento de instruções ➔ somente leitura/execução
- O *hardware* de tradução de endereços verifica os bits de proteção associados com cada entrada na tabela de segmentos
- Possibilidade de detecção pelo *hardware* de violação de limites de arranjos, matrizes, vetores, ...

Compartilhamento

- Cada processo tem sua tabela de segmentos: pode-se compartilhar segmentos entre dois processos diferentes, fazendo-se entradas de suas tabelas apontarem para a mesma posição de memória física
- Pode-se compartilhar programas inteiros ou apenas funções, procedimentos, etc
- Problemas quando o segmento compartilhado faz referências internas a si mesmo (ex.: comando de desvio) ➔ se esse segmento é compartilhado, todos os processos que o compartilham devem usar o mesmo número de segmento

Compartilhamento



Fragmentação

- Segmentos têm tamanho variável (ao contrário das páginas)
- A alocação de dinâmica de memória é normalmente resolvida com um algoritmo de escolha do primeiro (*first fit*) ou melhor (*best fit*) ou ..., bloco de memória
- Segmentação pode causar fragmentação externa (como no esquema de partição da memória em partes de tamanho variável) ➔ pode-se usar compactação
- Se o tamanho médio de segmentos for pequeno ➔ fragmentação também é pequena (é mais fácil alocar pequenos segmentos do que processos)

Segmentação com Paginação