Parte 3 – Processos e Threads

Gerência de Memória Virtual

Sumário

- Introdução
- Espaço de Endereçamento Virtual
- Mapeamento
- Memória Virtual por Paginação
  - Política de Busca de Páginas
  - Política de Alocação de Páginas
  - Política de Substituição de Páginas
  - Working Set
  - Algoritmo de Substituição de Páginas
  - Tamanho de Página
  - Translation Lookaside Buffer (TLB)

Sumário

- Memória Virtual por Paginação (cont.)
  - Proteção de Memória
  - Compartilhamento de Memória

- Memória Virtual por Segmentação
- Memória Virtual por Segmentação com Paginação
- Swapping em Memória Virtual
- Thrashing

Contextualização

- As diversas técnicas de gerenciamento de memória evoluíram no sentido de maximizar o número de processos residentes na memória principal e reduzir a fragmentação.

- O tamanho de um programa e de suas estruturas de dados estava limitado ao tamanho da memória disponível.

- Através das técnicas de swapping, vimos como ocorre a troca de processos entre a MP e a MS.

Contextualização

- Memória virtual é uma técnica que combina a memória principal e secundária dando ao usuário a ilusão de existir uma memória muito maior que a capacidade real da memória principal.

- Fundamenta-se em não vincular o endereçamento feito pelo programa dos endereços físicos da memória principal.

- Programas e suas estruturas de dados deixam de estar limitados ao tamanho da memória física disponível.

Contextualização

- Vantagem: permitir um número maior de processos compartilhando a memória principal, já que apenas partes de cada processo estarão residentes.

- Utilização mais eficiente do processador e possibilita minimizar o problema da fragmentação da memória principal.

Espaço de Endereçamento Virtual

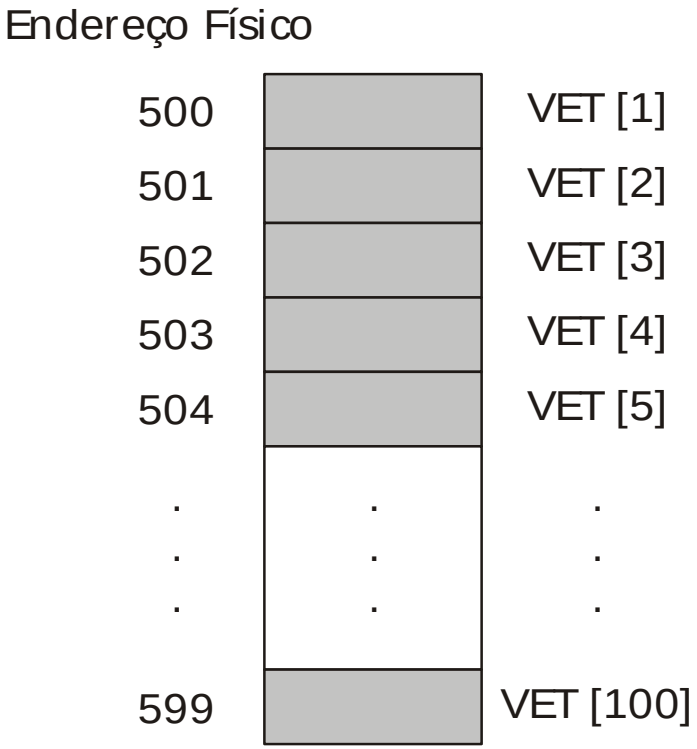
- O conceito de memória virtual se aproxima muito da ideia de um vetor.

- Quando um programa faz referência a um elemento do vetor, não há preocupação em saber a posição de memória daquele dado.

- O compilador se encarrega de gerar instruções que implementam esse mecanismo, tornando-o totalmente transparente ao programador.

Espaço de Endereçamento Virtual

- Vetor de 100 posições



Espaço de Endereçamento Virtual

- É utilizada uma abstração semelhante, porém em relação aos endereços dos programas e dados.

- Um programa no ambiente de memória virtual não faz referência a endereços físicos de memória, apenas virtuais.

- Execução de uma instrução → tradução de endereço virtual para físico → mapeamento.

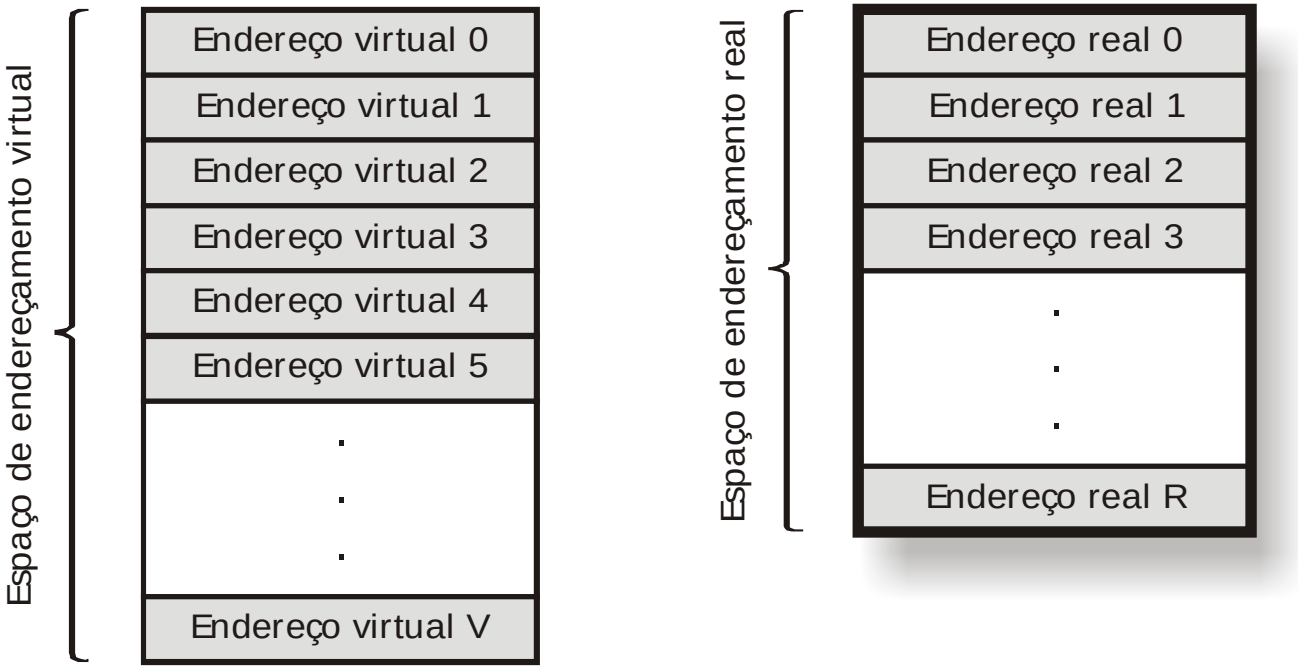
Espaço de Endereçamento Virtual

- Em ambientes que implementam memória virtual, o espaço de endereçamento do processo é conhecido como espaço de endereçamento virtual.

- O conjunto de endereços reais que o processador pode referenciar é chamado de espaço de endereçamento real.

Espaço de Endereçamento Virtual

- Espaço de endereçamento virtual e real



Espaço de Endereçamento Virtual

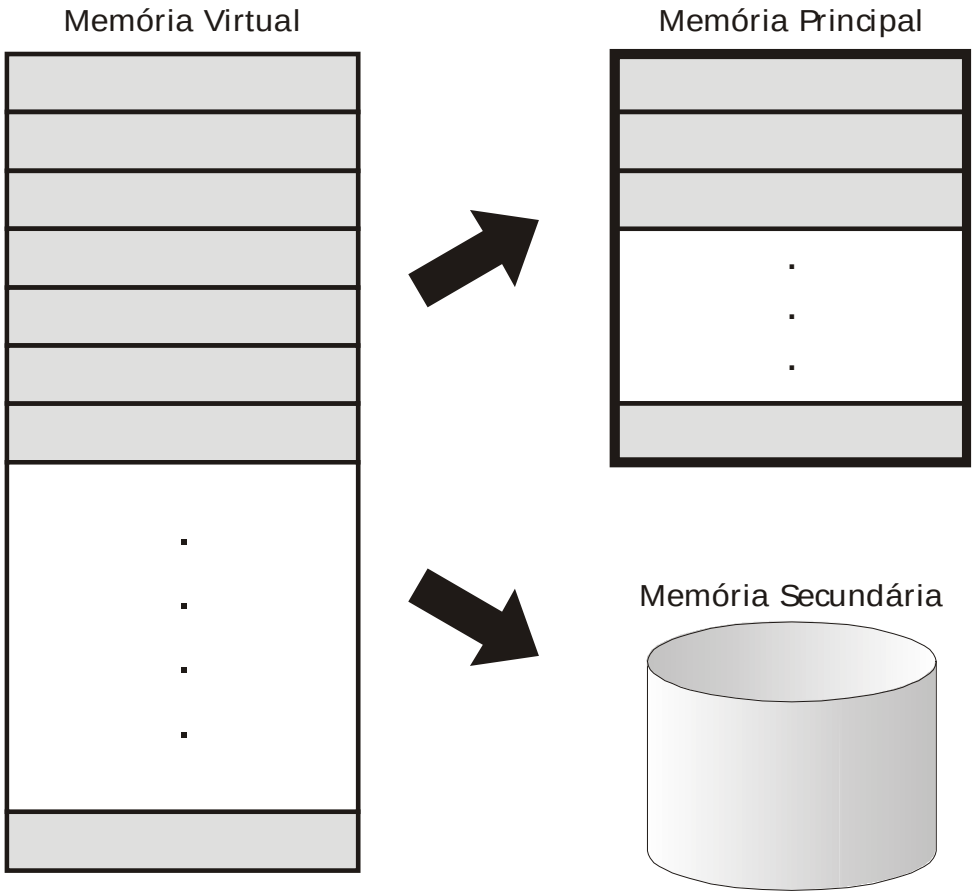
- O espaço de endereçamento virtual não tem nenhuma relação direta com os endereços no espaço real.

- Um programa pode fazer referência a endereços virtuais que estejam fora dos limites da memória principal.

- Para isso, o sistema utiliza a memória secundária como extensão da memória principal.

Espaço de Endereçamento Virtual

- Espaço de endereçamento virtual



Mapeamento

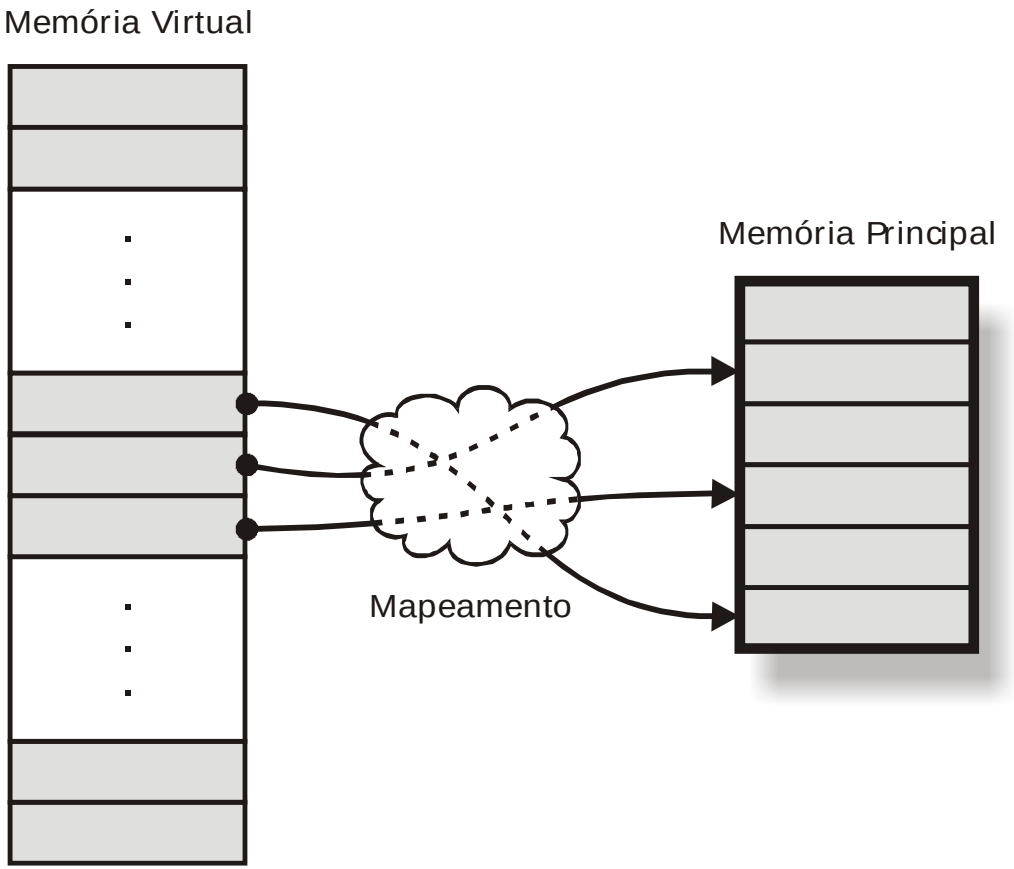
- A UCP apenas executa instruções e referencia dados residentes no espaço de endereçamento real.

- O Mapeamento permite traduzir um endereço localizado no espaço virtual para um associado no espaço real.

- Desta forma, um programa não mais precisa estar necessariamente em endereços contíguos na memória principal para ser executado.

Mapeamento

- Mapeamento

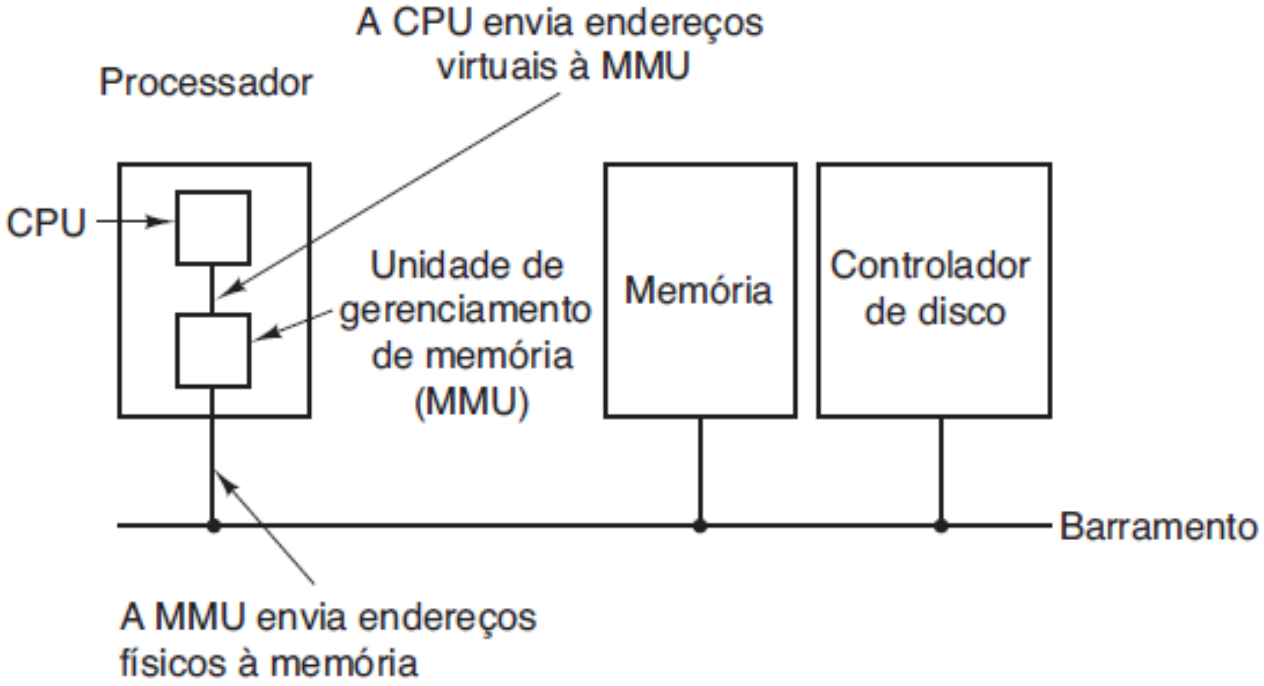


Mapeamento

- A tarefa de tradução dos endereços virtuais é realizada por hardware juntamente com o SO.
- O dispositivo de hardware responsável por esta tradução é conhecido como Unidade de Gerência de Memória.
- Memory Management Unit — MMU
- Cada processo tem o seu espaço de endereçamento virtual como se possuísse sua própria memória.

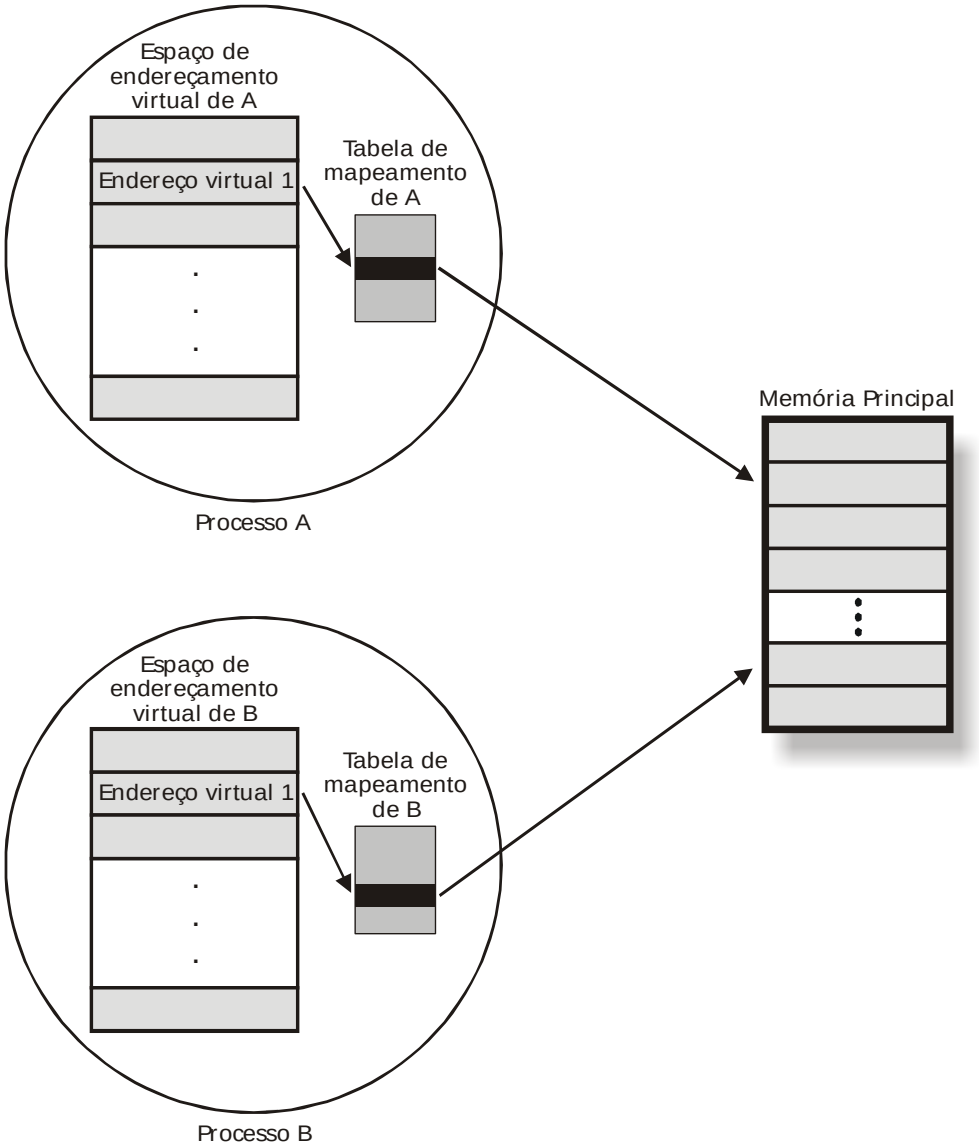
Mapeamento

A posição e função da MMU. Aqui a MMU é mostrada como parte do chip da CPU porque isso é comum hoje. No entanto, logicamente, poderia ser um chip separado, como era no passado.



Mapeamento

- Tabela de mapeamento



Mapeamento

- A tabela de mapeamento é uma estrutura de dados existente para cada processo.
- Quando em execução, o sistema utiliza sua tabela de mapeamento para realizar a tradução de seus endereços virtuais.
- Se outro processo for executado, o sistema deve referenciar a tabela do novo processo.
- A troca de tabelas de mapeamento é realizada por um registrador que indica a posição inicial da tabela corrente.

Mapeamento

- O mapeamento não é realizado para cada célula na memória principal.

- As tabelas mapeiam blocos de dados, cujo tamanho determina o número de entradas existentes nas tabelas de mapeamento.

- Quanto maior o bloco, menos entradas nas tabelas de mapeamento e, consequentemente, tabelas de mapeamento que ocupam um menor espaço de memória.

Mapeamento

- Espaço virtual x tamanho do bloco

Espaço de endereçamento virtual	Tamanho do bloco	Número de blocos	Número de entradas na tabela de mapeamento
2^32	512	2^23	2^23
2^32	4k (2^12)	2^20	2^20
2^64	4k	2^52	2^52
2^64	2^32	2^32	2^32

Bloco de tamanho fixo (paginação)
Bloco de tamanho variável (segmentação)

Memória Virtual por Paginação

- Técnica de gerência de memória em que o espaço de endereçamento virtual e o real são divididos em blocos de mesmo tamanho chamados páginas.

- São denominadas páginas virtuais no espaço virtual.

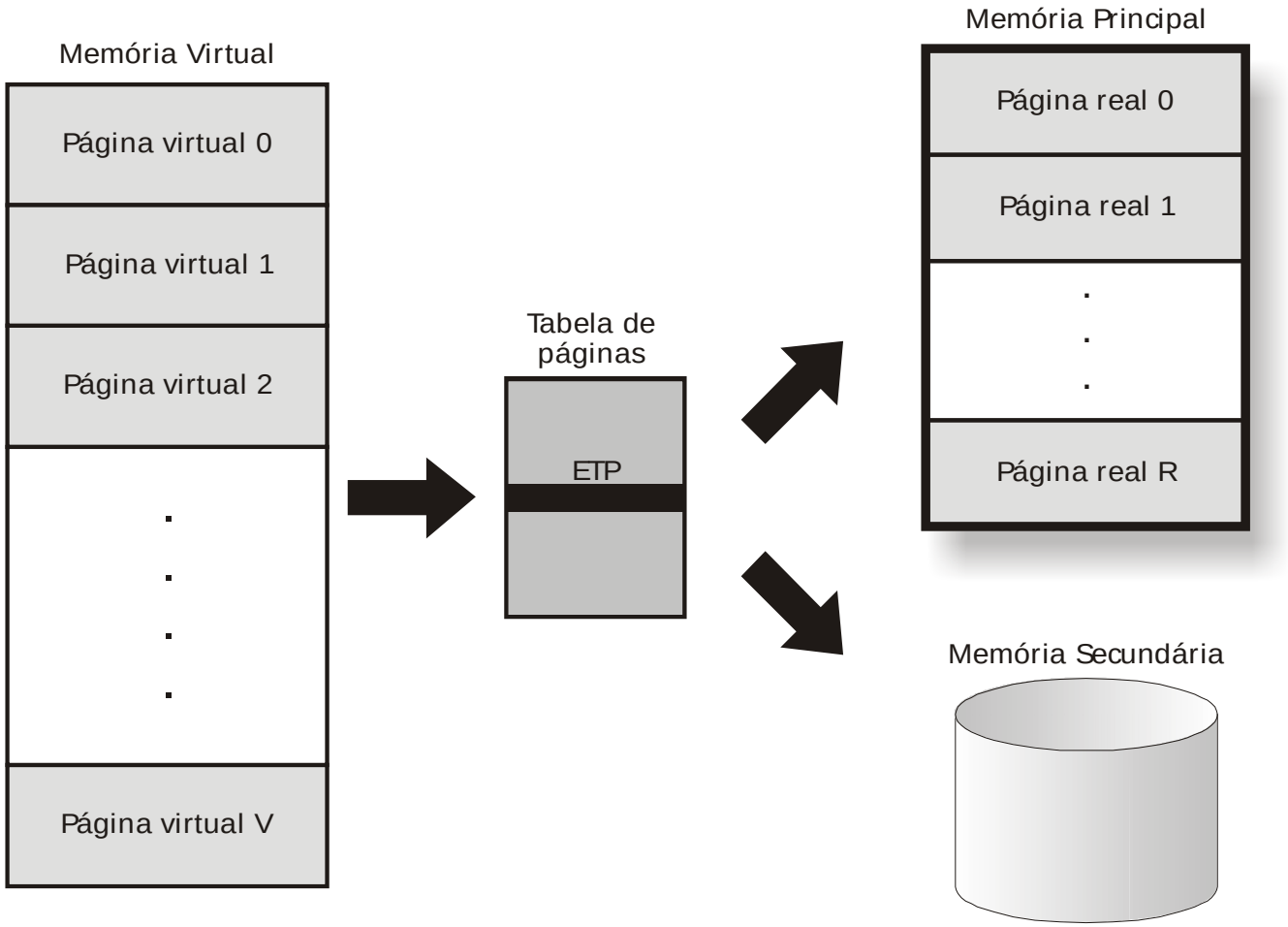
- São denominadas páginas reais ou frames no espaço real (memória).

Memória Virtual por Paginação

- Mapeamento de endereço é realizado pela tabela de páginas.
- Cada processo possui sua própria tabela de páginas.
- Cada página virtual do processo possui uma entrada na tabela (entrada na tabela de páginas — ETP), com informações de mapeamento que permitem ao sistema localizar a página real correspondente.

Memória Virtual por Paginação

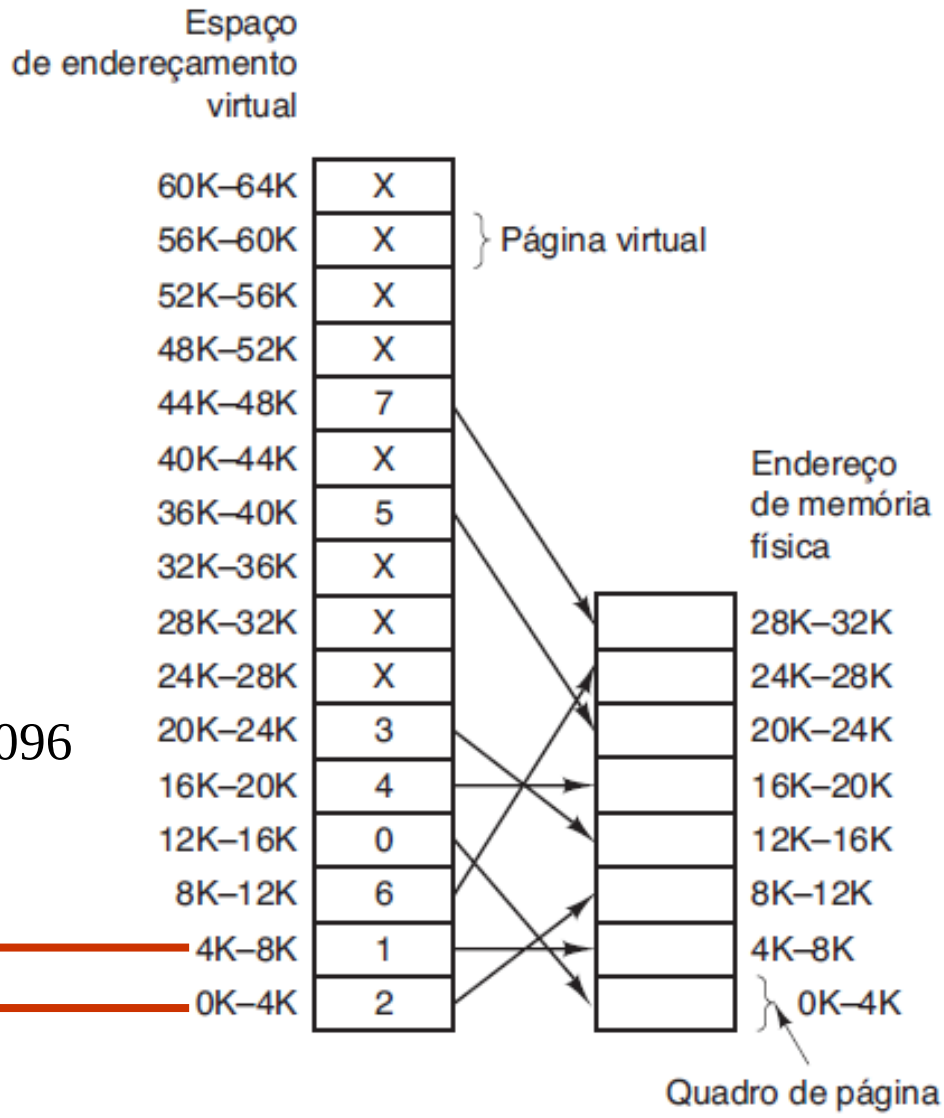
- Tabela de páginas



Memória Virtual por Paginação

- Exemplo:

Endereços de 16 bits
Tamanho de página: 4Kb



Cada página contém exatamente 4096 endereços

Endereços de 4096 a 8191

Endereços de 0 a 4095

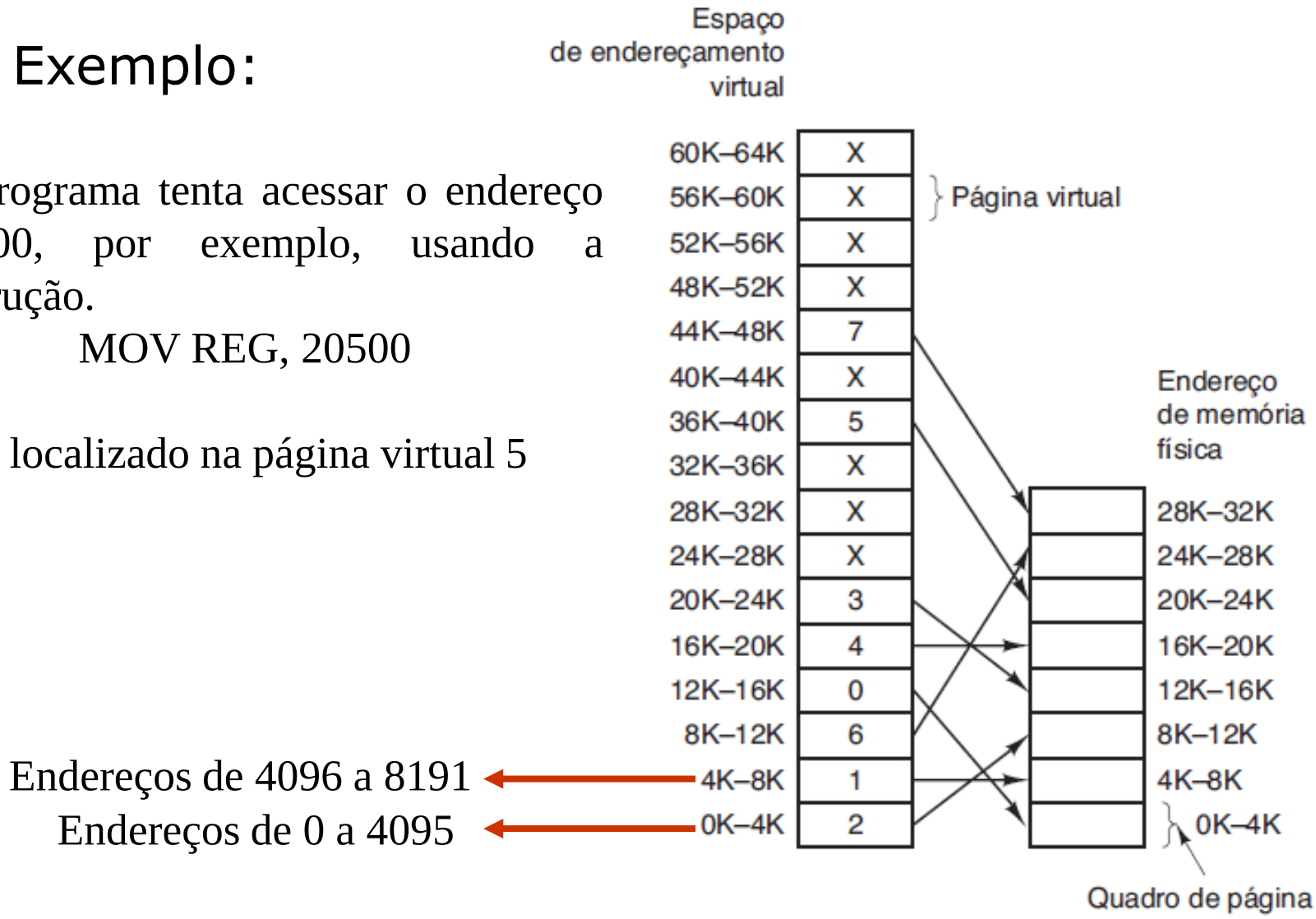
Memória Virtual por Paginação

- Exemplo:

O programa tenta acessar o endereço 20500, por exemplo, usando a instrução.

MOV REG, 20500

Está localizado na página virtual 5



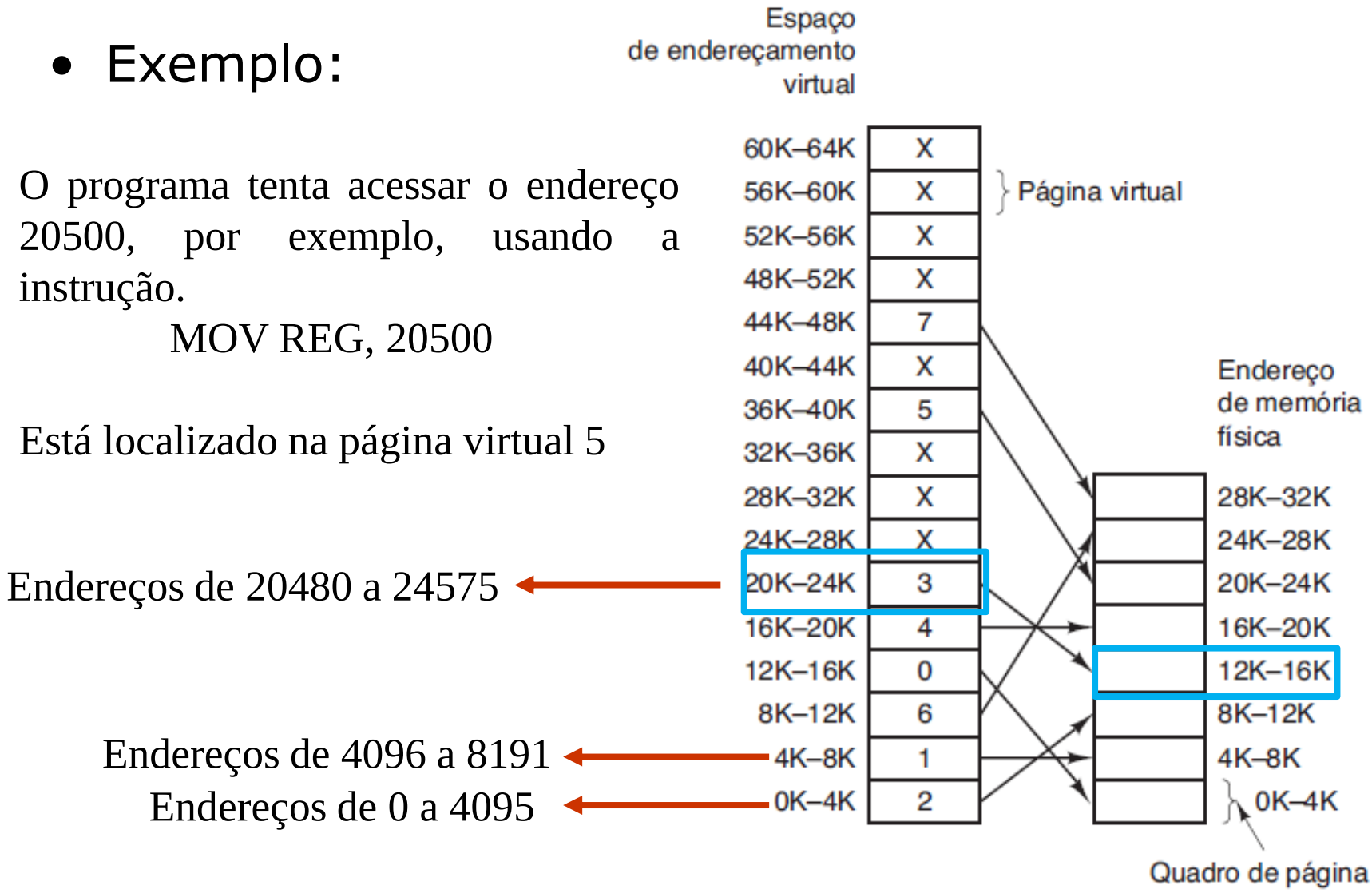
Memória Virtual por Paginação

- Exemplo:

O programa tenta acessar o endereço 20500, por exemplo, usando a instrução.

MOV REG, 20500

Está localizado na página virtual 5



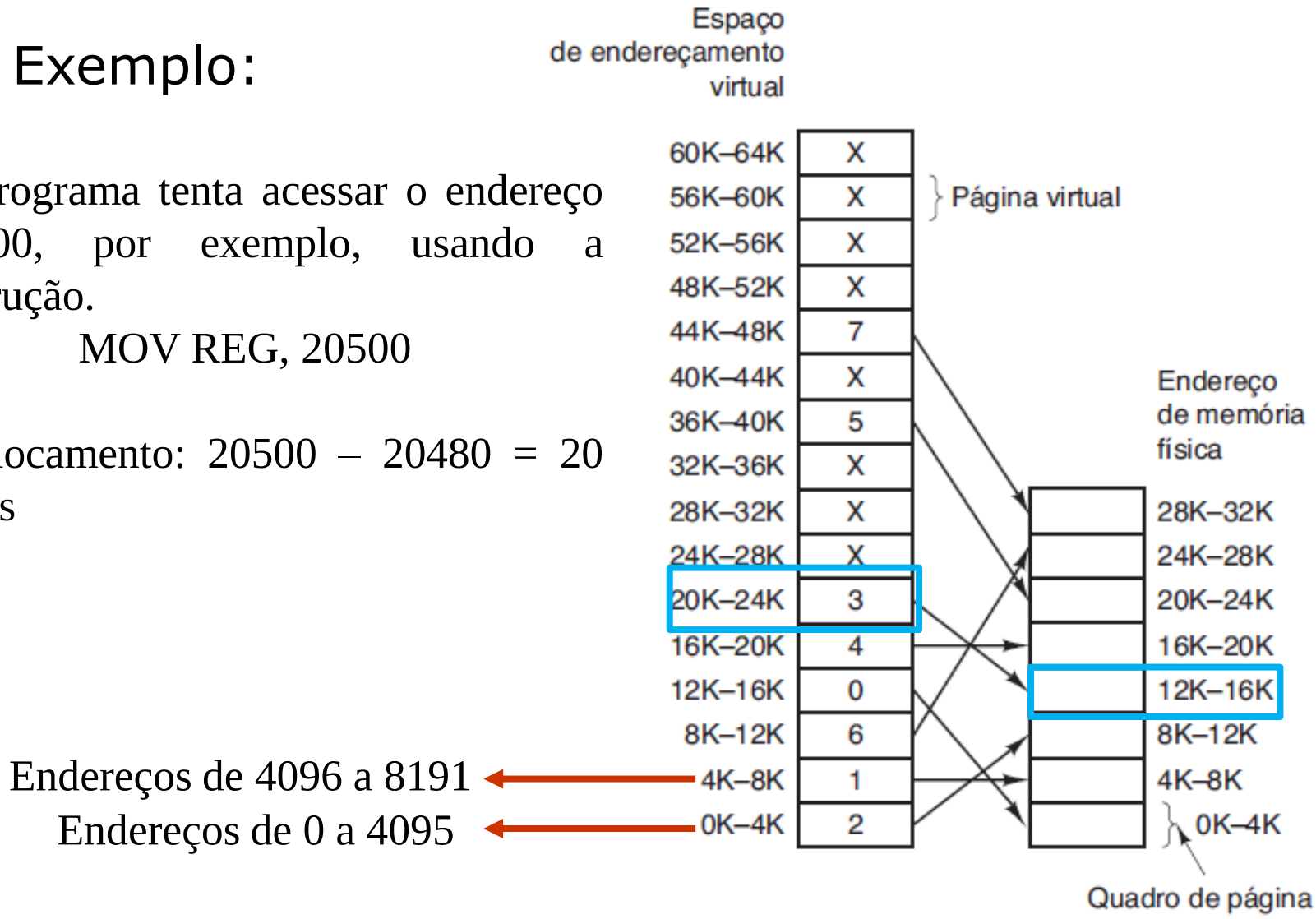
Memória Virtual por Paginação

- Exemplo:

O programa tenta acessar o endereço 20500, por exemplo, usando a instrução.

MOV REG, 20500

Deslocamento: 20500 – 20480 = 20 bytes



Memória Virtual por Paginação

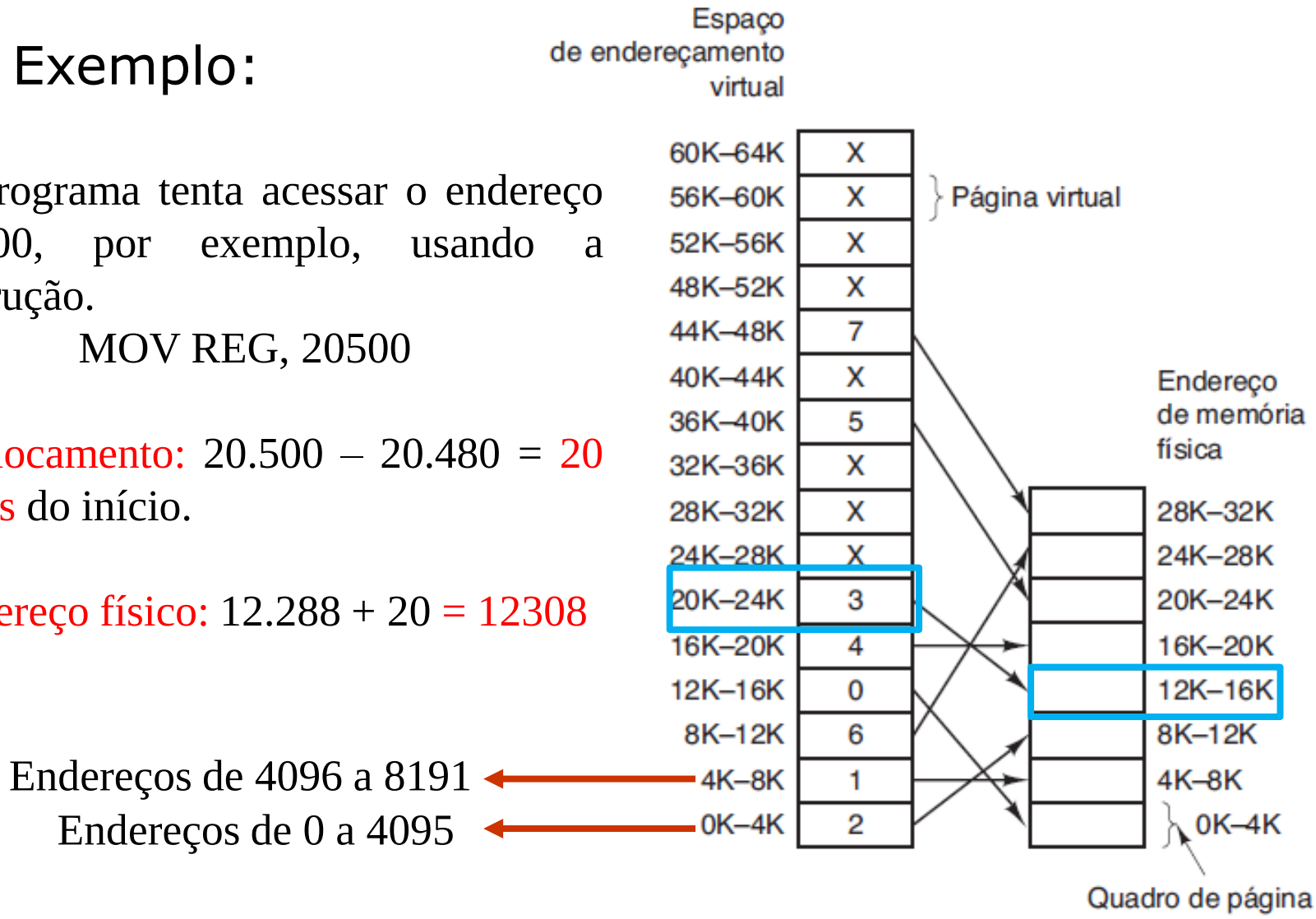
- Exemplo:

O programa tenta acessar o endereço 20500, por exemplo, usando a instrução.

MOV REG, 20500

Deslocamento: 20.500 – 20.480 = 20 bytes do início.

Endereço físico: 12.288 + 20 = 12308



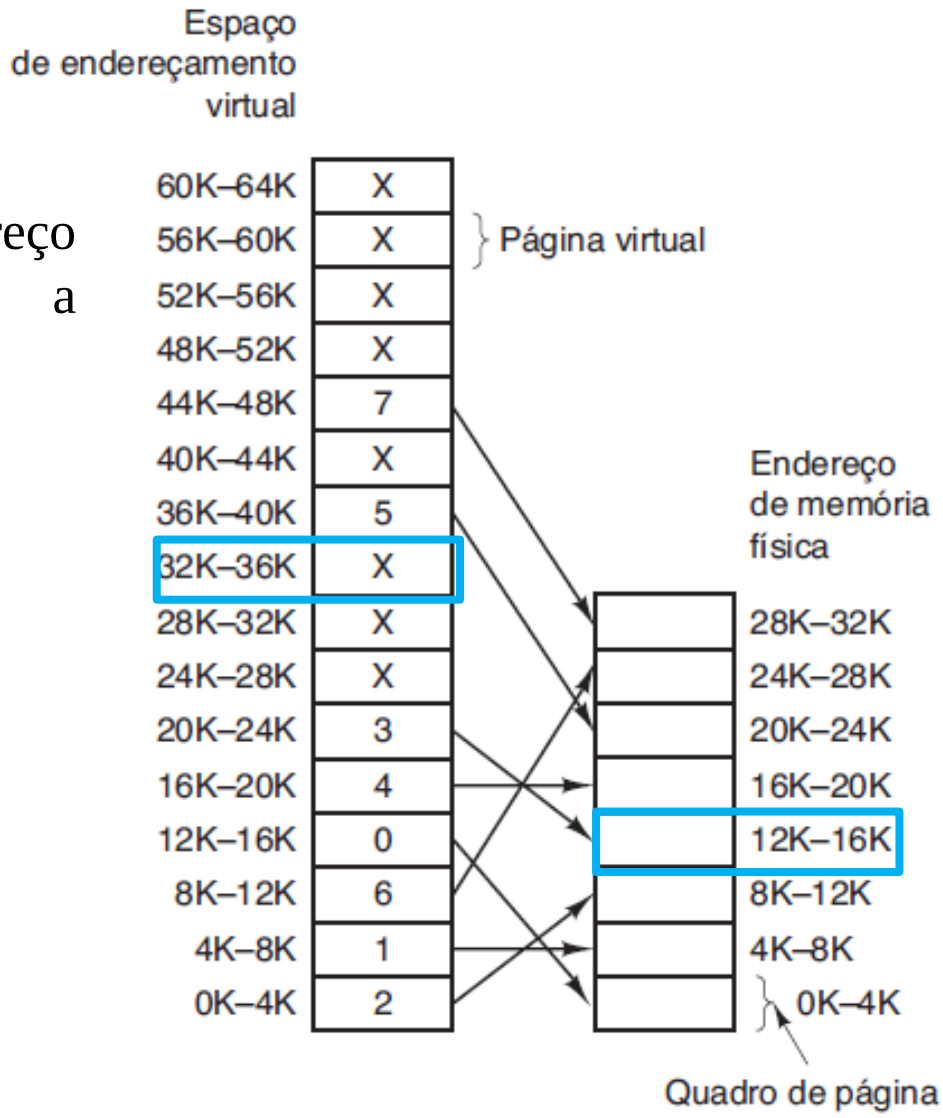
Memória Virtual por Paginação

- Exemplo:

O programa tenta acessar o endereço 32780, por exemplo, usando a instrução.

MOV REG, 32780

Byte 12 da página 8 (32.768 + 12)



Memória Virtual por Paginação

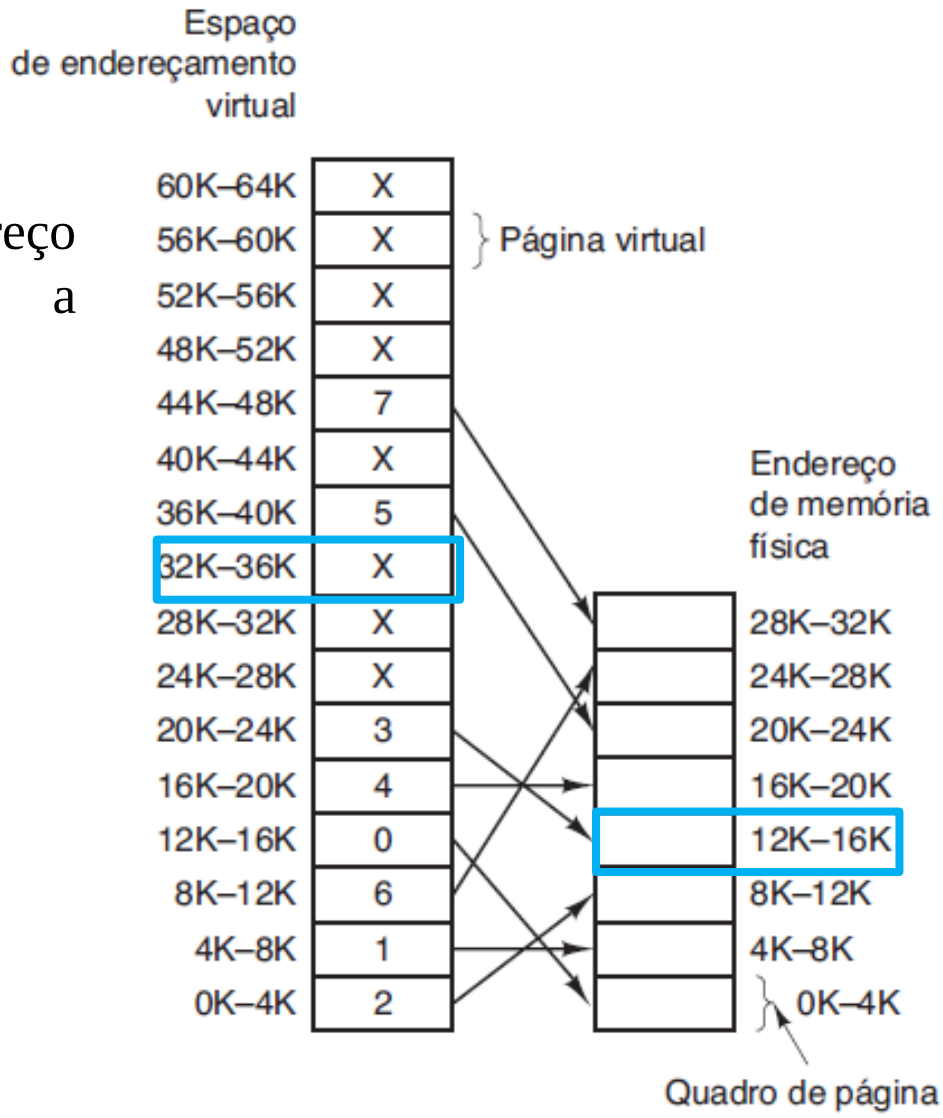
- Exemplo:

O programa tenta acessar o endereço 32780, por exemplo, usando a instrução.

MOV REG, 32780

Byte 12 da página 8 (32.768 + 12)

Não há mapeamento
"Page fault"



Memória Virtual por Paginação

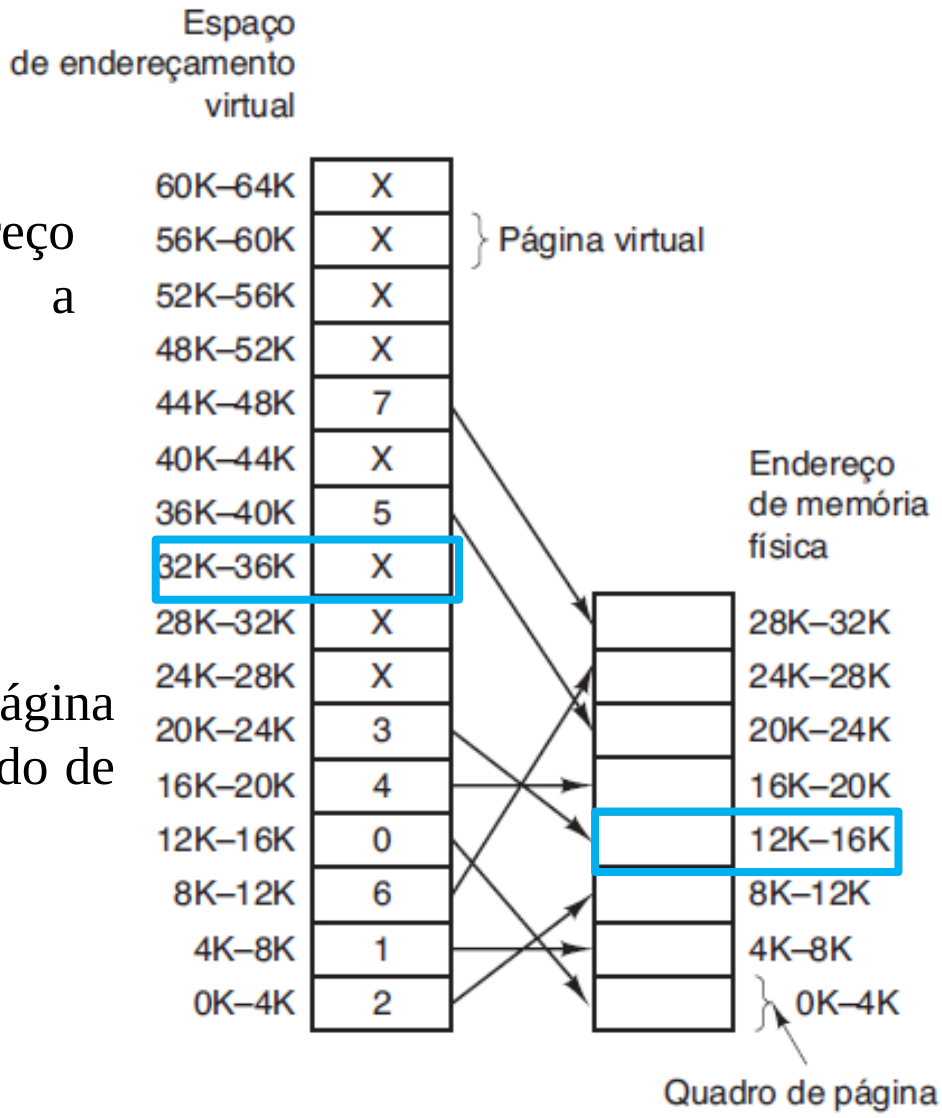
- Exemplo:

O programa tenta acessar o endereço 32780, por exemplo, usando a instrução.

MOV REG, 32780

Byte 12 da página 8 (32.768 + 12)

O SO escolhe um quadro de página pouco usado e escreve seu conteúdo de volta para o disco.



Memória Virtual por Paginação

- Exemplo:

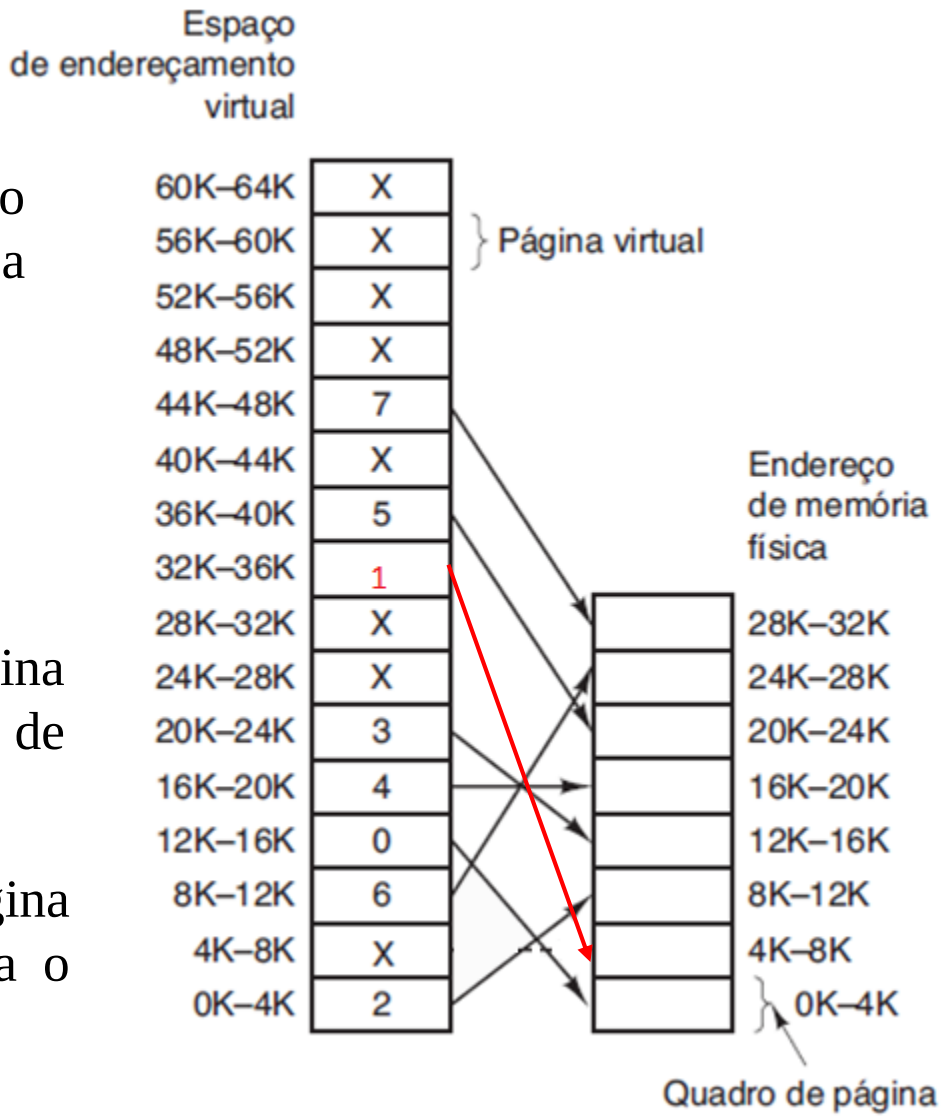
O programa tenta acessar o endereço 32780, por exemplo, usando a instrução.

MOV REG, 32780

Byte 12 da página 8 (32.768 + 12)

O SO escolhe um quadro de página pouco usado e escreve seu conteúdo de volta para o disco.

Ele então carrega do disco a página recém-referenciada no frame, muda o mapa e reinicia a instrução.



Memória Virtual por Paginação

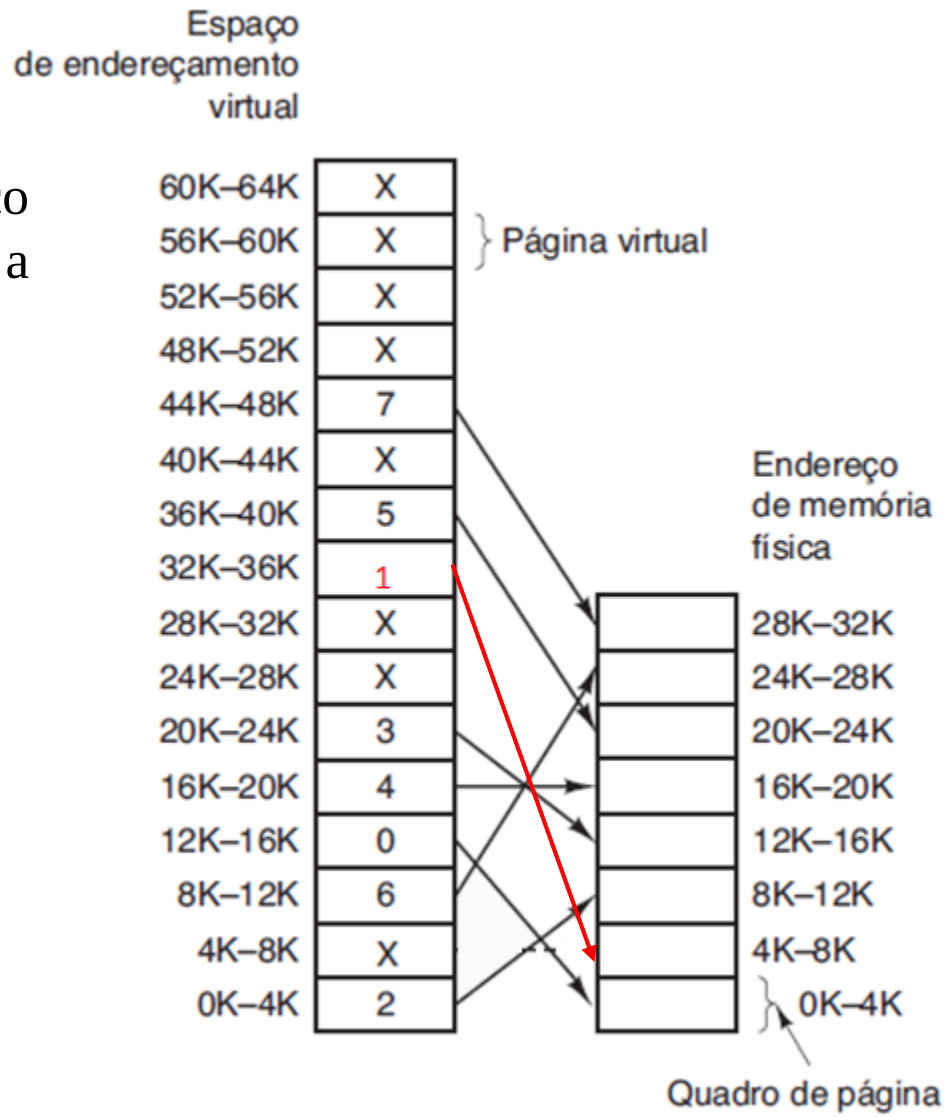
- Exemplo:

O programa tenta acessar o endereço 32780, por exemplo, usando a instrução.

MOV REG, 32780

Byte 12 da página 8 (32.768 + 12)

Endereço físico: 4.096 + 12 = 4108



Memória Virtual por Paginação

- Quando um programa é executado, as páginas virtuais são transferidas da memória secundária para a memória principal e colocadas nos frames.

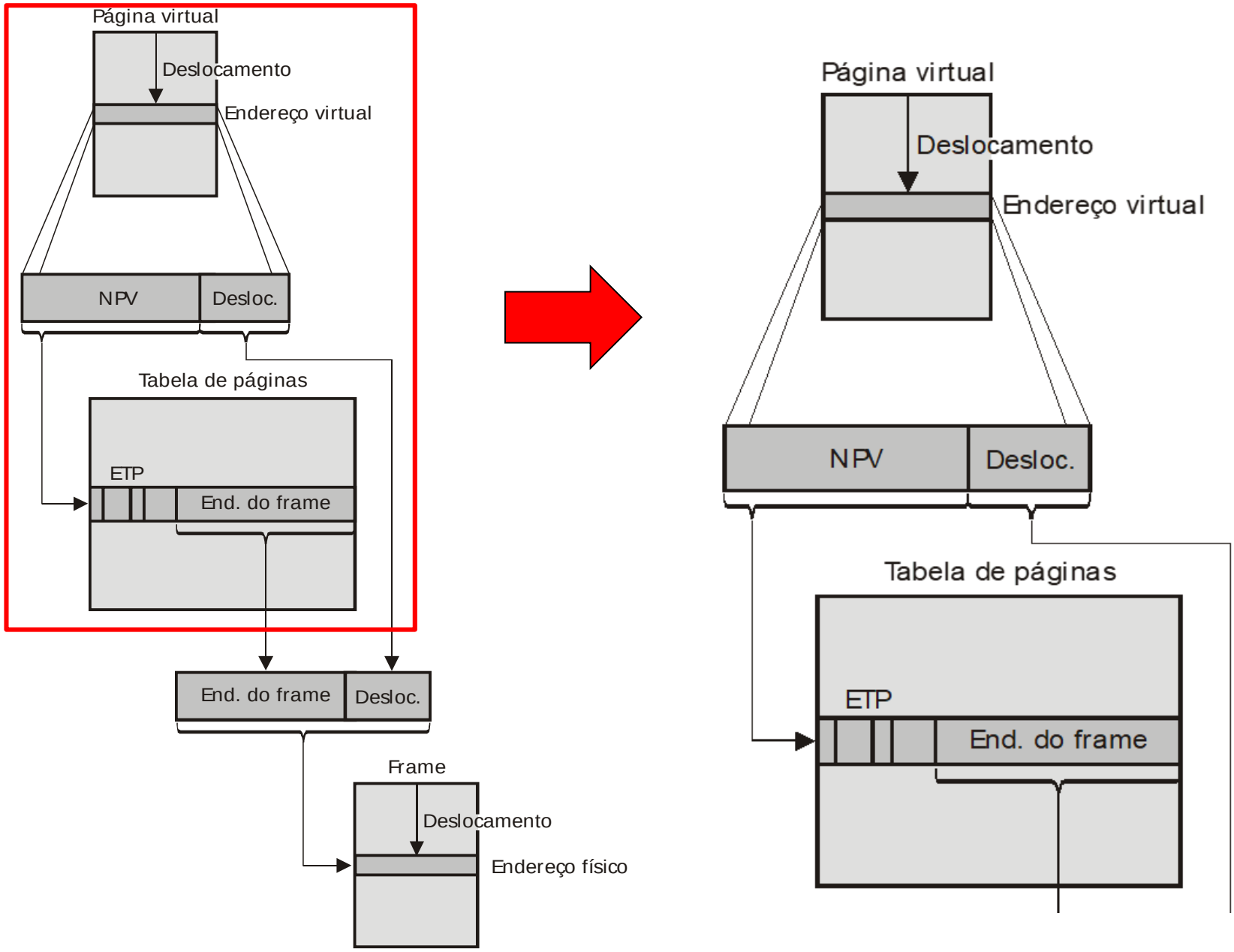
- Nessa técnica, o endereço virtual é formado pelo número da página virtual (NPV) e por um deslocamento.

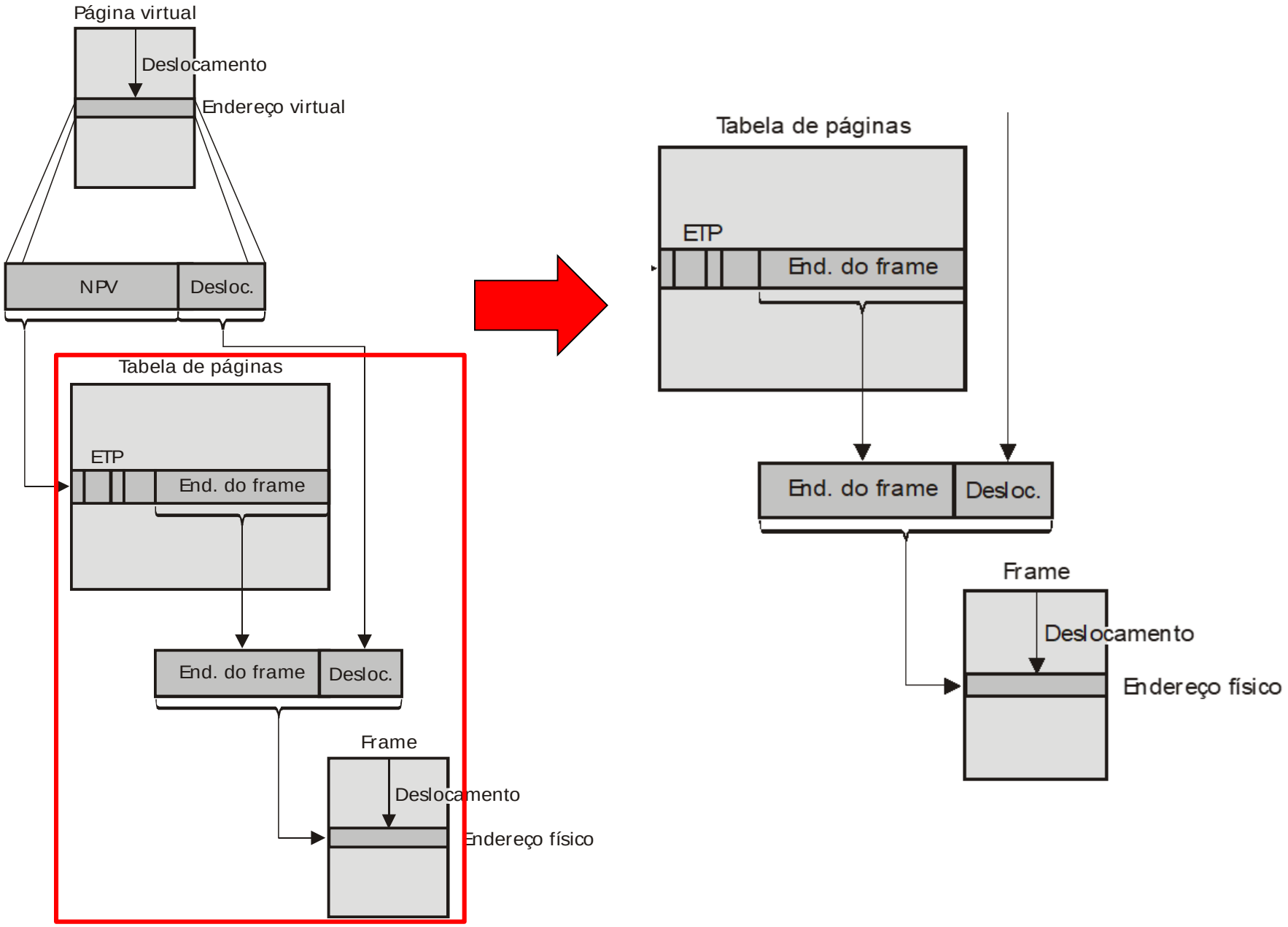
- O NPV identifica unicamente a página virtual que contém o endereço, funcionando como um índice na tabela de páginas.

Memória Virtual por Paginação

- O deslocamento indica a posição do endereço virtual em relação ao início da página na qual se encontra.

- O endereço físico é obtido, então, combinando-se o endereço do frame, localizado na tabela de páginas, com o deslocamento, contido no endereço virtual.


Página virtual
Deslocamento
Endereço virtual
NPV
Desloc.
Tabela de páginas
ETP
End. do frame
End. do frame
Desloc.
Frame
Deslocamento
Endereço físico
Página virtual
Deslocamento
Endereço virtual
NPV
Desloc.
Tabela de páginas
ETP
End. do frame


Página virtual
Deslocamento
Endereço virtual
NPV
Desloc.
Tabela de páginas
ETP
End. do frame
End. do frame
Desloc.
Frame
Deslocamento
Endereço físico
Tabela de páginas
ETP
End. do frame
End. do frame
Desloc.
Frame
Deslocamento
Endereço físico

Memória Virtual por Paginação

- Agora vamos olhar dentro da MMU para ver como ela funciona e por que escolhemos usar um tamanho de página que é uma potência de 2.

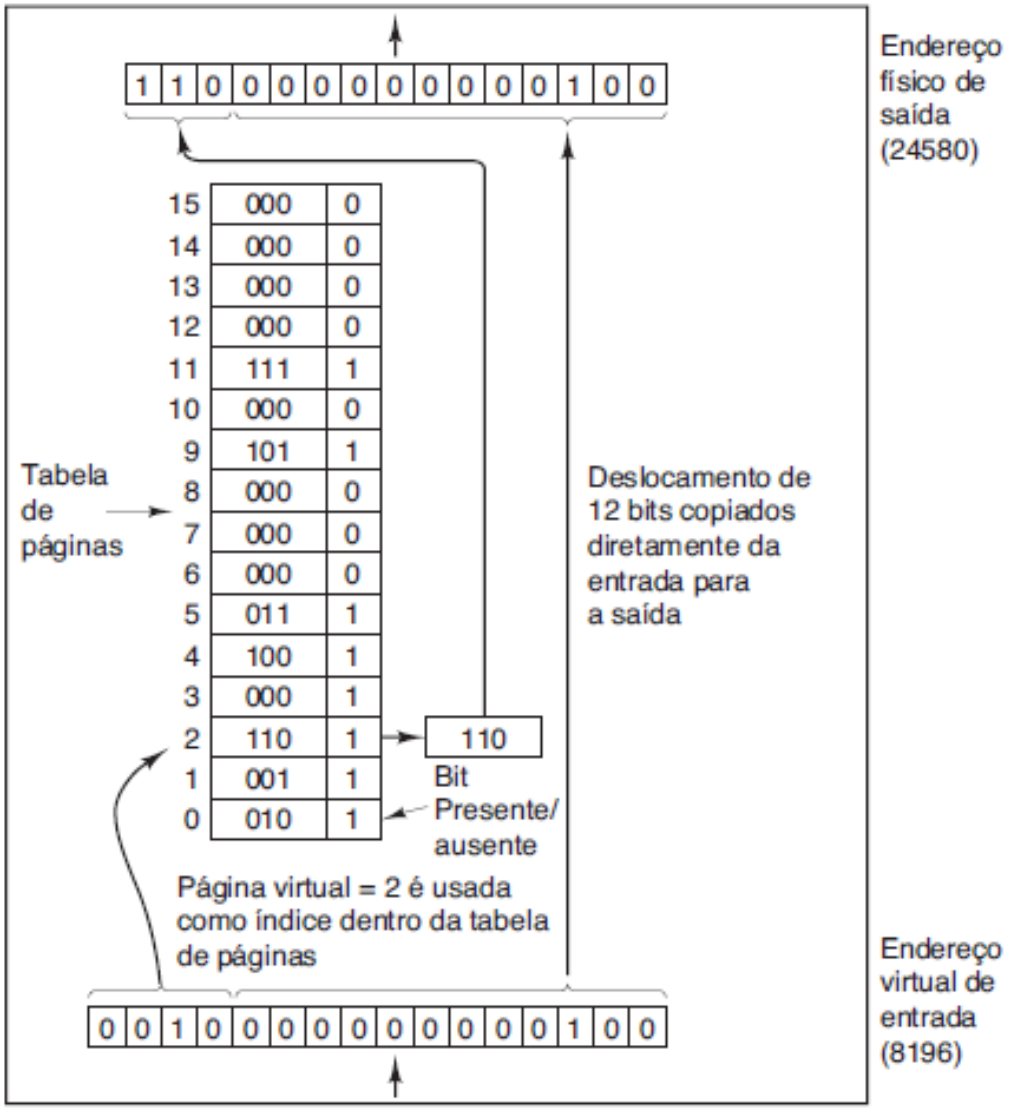
- 4K = 4096 = 2^12
  - 0k a 4095B → 0000000000000000 a 0000111111111111
  - 4k a 8191B → 0001000000000000 a 0001111111111111
  - 8k a 12287B → 0010000000000000 a 0010111111111111
  - 12k a 16383B → 0011000000000000 a 0011111111111111
  - …

- Os bits mais significativos (vermelho) são o número da página.

Memória Virtual por Paginação

- Exemplo:

Endereço virtual 8196 sendo mapeado pela MMU pela tabela de página.



Memória Virtual por Paginação

- A ETP possui outras informações, como o bit de validade (*valid bit*), que indica se uma página está ou não na memória principal.
  - Bit com valor "0" → página virtual não está na memória;
  - Bit com valor "1" → página virtual está na memória.
- Caso a página não esteja na memória → page fault

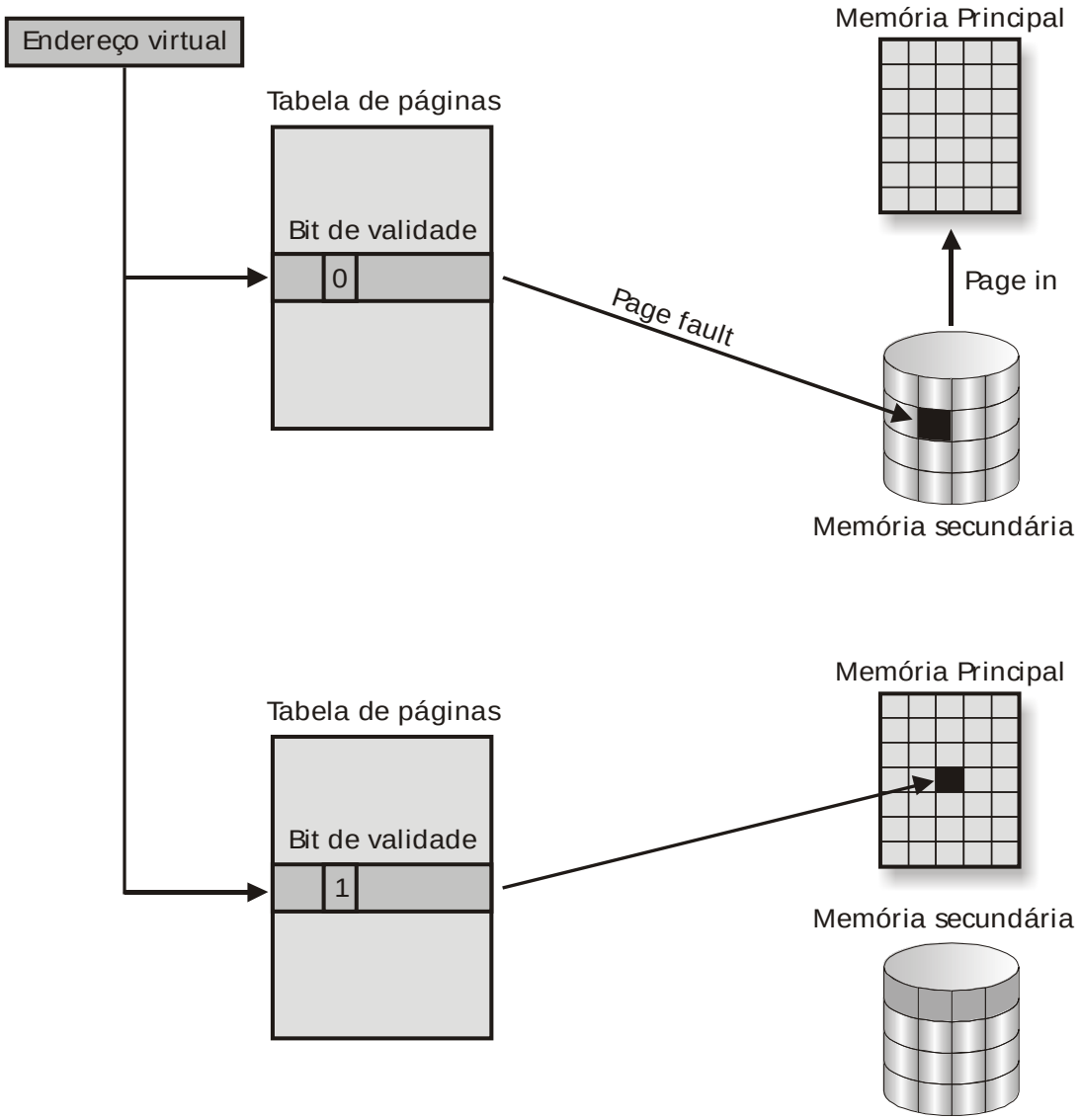
Memória Virtual por Paginação

- Caso a página não esteja na memória → page fault

- O sistema transfere a página da memória secundária para a memória principal, realizando uma operação de E/S conhecida como page in, ou paginação.

- O número de page faults gerado por um processo depende de como o programa foi desenvolvido, além da política de gerência de memória.

Memória Virtual por Paginação



Política de Busca de Páginas

- Determina quando uma página deve ser carregada para a memória.

- Basicamente, existem duas estratégias para este propósito:
  - Paginação por demanda: as páginas dos processos são transferidas da memória secundária para a principal apenas quando são referenciadas.

  - Paginação antecipada: o sistema carrega para a memória principal, além da página referenciada, outras páginas que podem ou não ser necessárias ao processo ao longo do seu processamento.

Política de Alocação de Páginas

- A política de alocação de páginas determina quantos frames cada processo pode manter na memória principal.

- Existem, basicamente, duas alternativas:
  - Política de alocação fixa: cada processo tem um número máximo de frames que pode ser utilizado durante a execução do programa.
  - Caso o número de páginas reais seja insuficiente, uma página do processo deve ser descartada para que uma nova seja carregada.
  - O limite de páginas deve ser definido na criação do processo.

Política de Alocação de Páginas

- Existem, basicamente, duas alternativas:
  - Política de alocação fixa – Problemas:
  - Se o número máximo de páginas alocadas for muito pequeno → elevado número de page faults.
  - Caso o número de páginas seja muito grande → cada processo ocupará um espaço maior que o necessário → reduz o número de processos residentes.

Política de Alocação de Páginas

- A política de alocação de páginas determina quantos frames cada processo pode manter na memória principal.

- Existem, basicamente, duas alternativas:
  - Política de alocação variável: número máximo de páginas alocadas ao processo pode variar durante sua execução em função de sua taxa de paginação e da ocupação da memória principal.

  - Processos com elevadas taxa de paginação → ampliar limite de máximo de frames → reduzir page faults.

  - Mais flexível, porém maior monitoramento → Overhead.

Políticas de Substituição de Páginas

- Quando um processo atinge o seu limite de alocação de frames e necessita alocar novas páginas, o SO deve selecionar, dentre as páginas alocadas, qual será liberada.

- Qualquer estratégia de substituição de páginas deve considerar se uma página foi ou não modificada antes de liberá-la.

- Caso contrário, os dados armazenados na página podem ser perdidos.

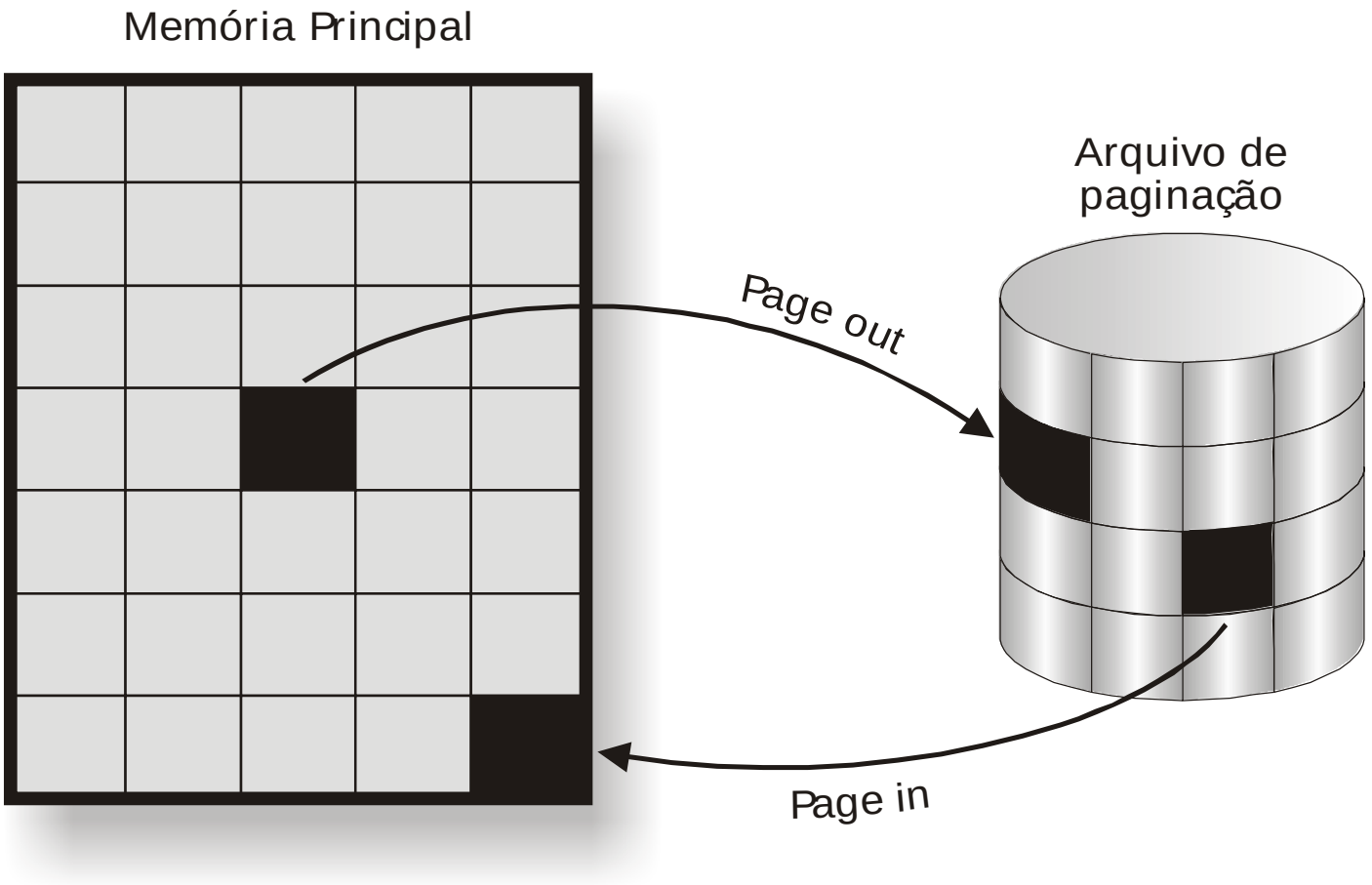
Políticas de Substituição de Páginas

- O sistema operacional consegue identificar as páginas modificadas através de um bit que existe em cada entrada da tabela de páginas, chamado bit de modificação.

- Na política de substituição local, apenas as páginas do processo que gerou o page fault são candidatas a realocação.

- Na política de substituição global, todas as páginas alocadas na memória principal são candidatas à substituição, independente do processo que gerou o page fault.

Políticas de Substituição de Páginas



Working Set

- O conceito de working set surgiu com o objetivo de reduzir o problema do thrashing e está relacionado ao princípio da localidade.

- Existem dois tipos de localidade que são observados durante a execução da maioria dos programas:

  - localidade espacial - é a tendência de que após uma referência a uma posição de memória sejam realizadas novas referências a endereços próximos.

Working Set

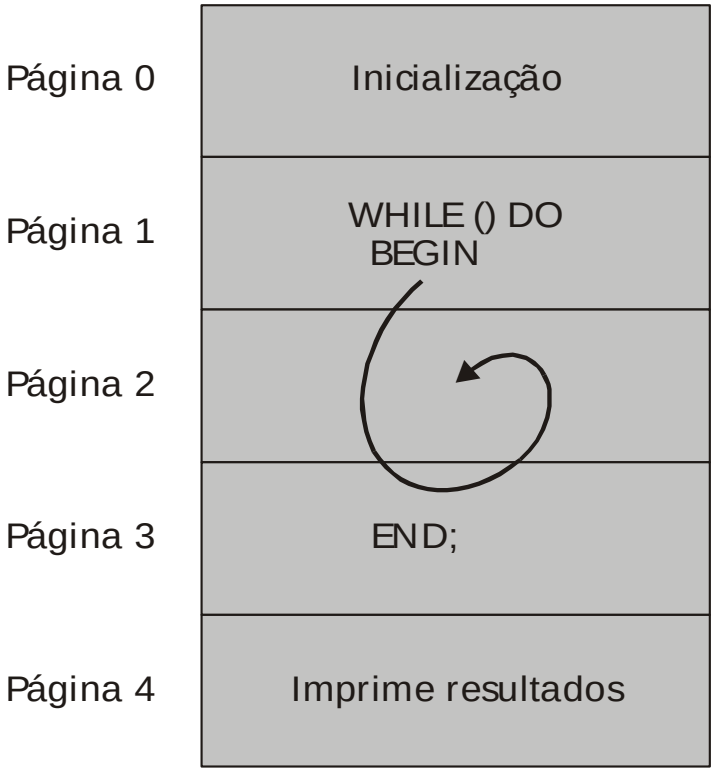
- O conceito de working set surgiu com o objetivo de reduzir o problema do thrashing e está relacionado ao princípio da localidade.

- Existem dois tipos de localidade que são observados durante a execução da maioria dos programas:

  - localidade temporal - é a tendência de que após a referência a uma posição de memória esta mesma posição seja novamente referenciada em um curto intervalo de tempo.

Working Set

- Conceito de localidade
  - Conjunto de páginas referenciadas durante um determinado tempo.



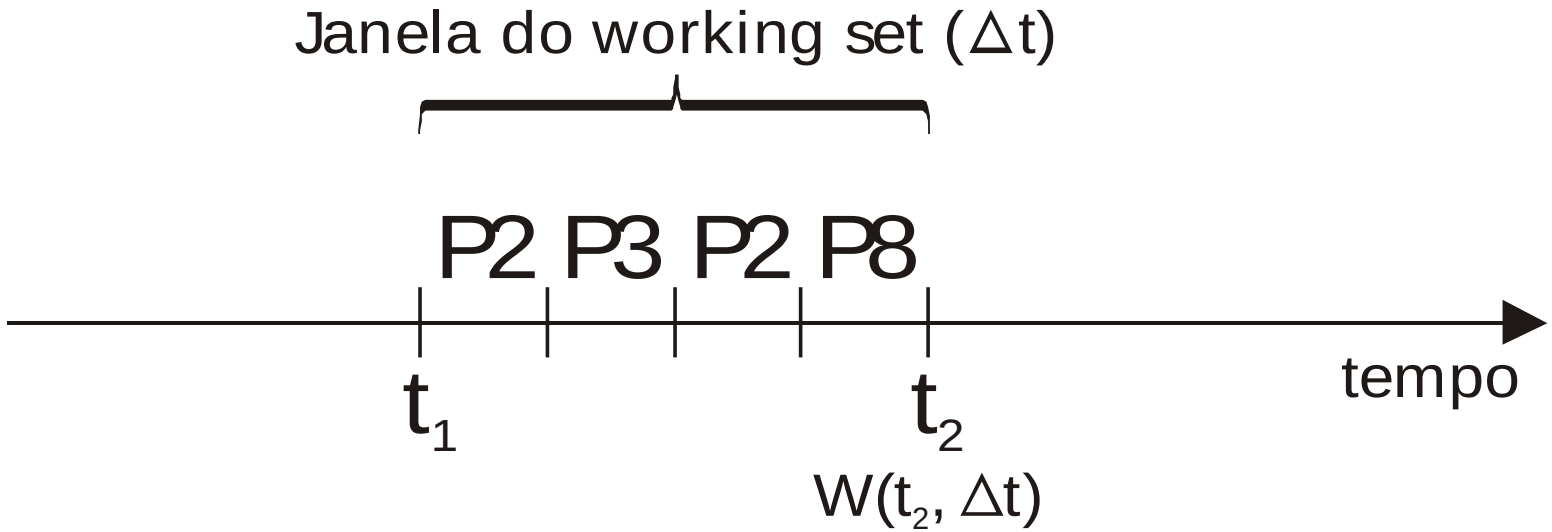
Working Set

- A partir da observação do princípio da localidade, Peter Denning formulou o modelo de working set.

- O conceito de working set é definido como o conjunto das páginas referenciadas por um processo durante determinado intervalo de tempo.

Working Set

- Modelo de working set



Working Set

- No instante t2 o working set do processo, W(t2,Δt), consiste nas páginas referenciadas no intervalo Δt(t2 – t1), isto é, as páginas P2, P3 e P8.

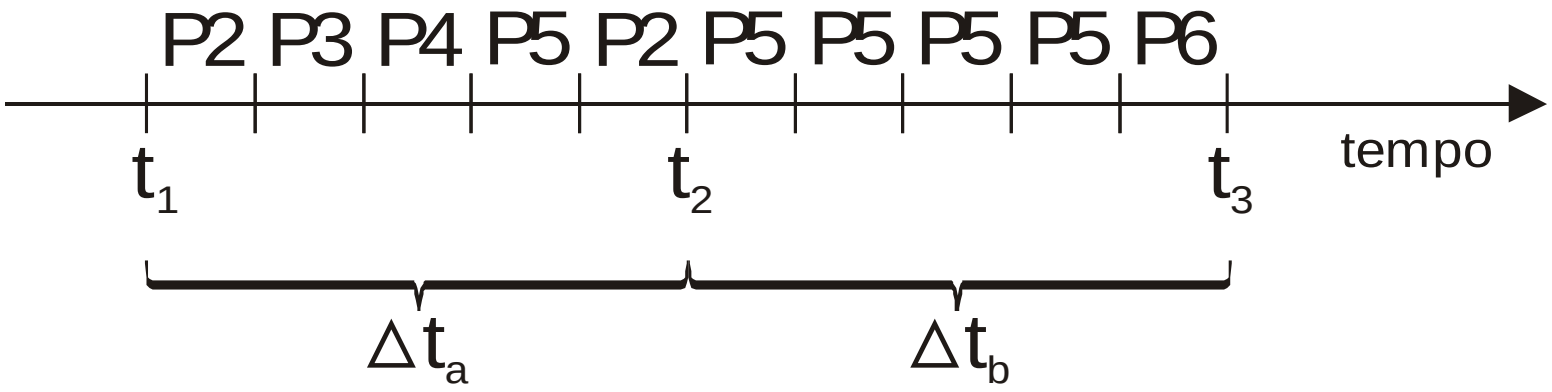
- O intervalo de tempo Δt é denominado janela do working set.

- o working set de um processo é uma função do tempo e do tamanho da janela do working set.

- Na janela o número de páginas distintas referenciadas é o tamanho do working set.

Working Set

- Tamanho do working set



Working Set

- Apesar de o conceito de working set ser bastante intuitivo, sua implementação não é simples por questões de desempenho.

- o modelo de working set deve ser implementado somente em sistemas que utilizam a política de alocação de páginas variável, onde o limite de páginas reais não é fixo.

- Uma maneira de implementar o modelo de working set é analisar a taxa de paginação de cada processo, conhecida como estratégia de frequência de page fault.

Algoritmos de Substituição de Páginas

- O maior problema na gerência de memória virtual por paginação não é decidir quais páginas carregar para a memória principal, mas quais liberar.

- Quando um processo necessita de uma nova página e não existem frames disponíveis, o sistema deverá selecionar, dentre as diversas páginas alocadas na memória, qual deverá ser liberada pelo processo.

Algoritmos de Substituição de Páginas

- Os algoritmos de substituição de páginas têm o objetivo de selecionar os frames que tenham as menores chances de serem referenciados em um futuro próximo.

- A partir do princípio da localidade, a maioria dos algoritmos tenta prever o comportamento futuro das aplicações em função do comportamento passado.

Algoritmos de Substituição de Páginas

- A partir do princípio da localidade, a maioria dos algoritmos tenta prever o comportamento futuro das aplicações em função do comportamento passado.

- Avaliando o número de vezes que uma página foi referenciada, o momento em que foi carregada para a memória principal e o intervalo de tempo da última referência.

Algoritmos de Substituição de Páginas

- Ótimo
  - Selecionar para substituição uma página que não será mais referenciada no futuro ou que levará o maior intervalo de tempo para ser novamente utilizada.
  - Na prática, impossível de ser implementado pois o SO não tem como conhecer o comportamento futuro das aplicações.
  - Essa estratégia é utilizada apenas como modelo comparativo na análise de outros algoritmos de substituição.

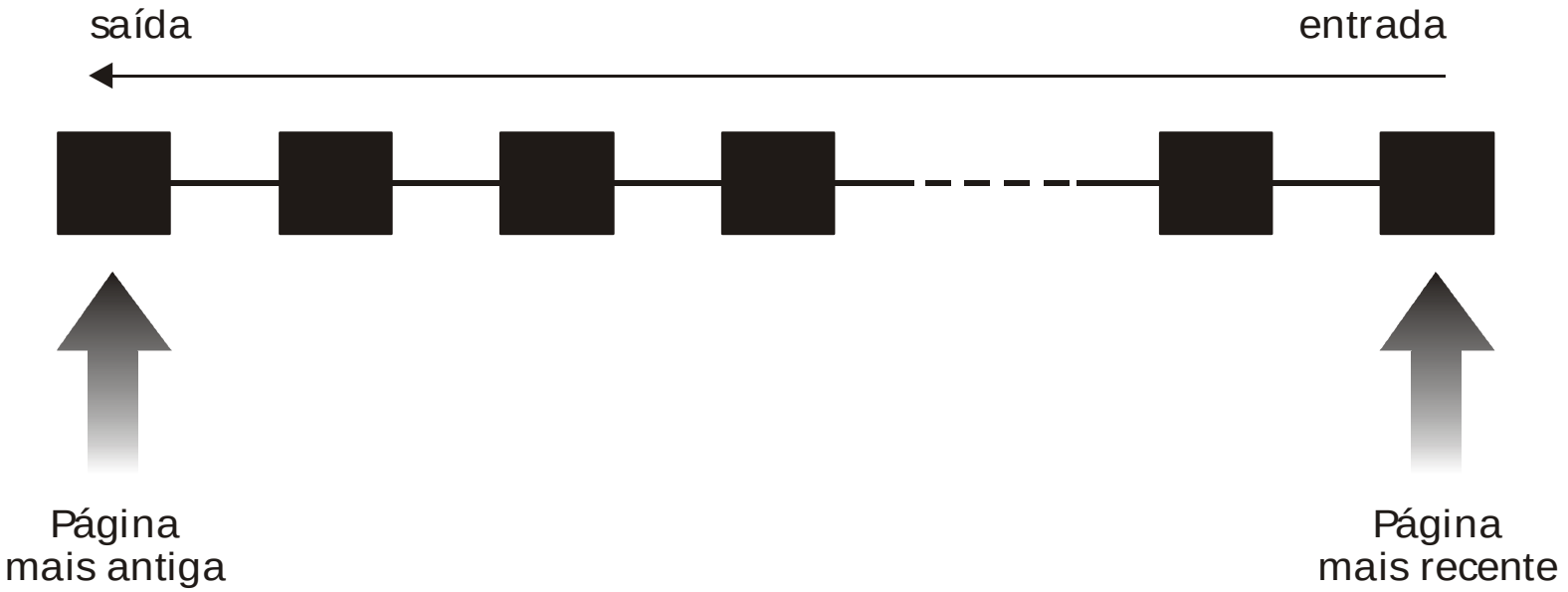
Algoritmos de Substituição de Páginas

- Aleatório
  - Não utiliza nenhum critério de seleção
  - Raramente Implementado

- FIFO (First-In-First-Out)
  - Página que primeiro foi utilizada será a primeira a ser escolhida, ou seja, o algoritmo seleciona a página que está há mais tempo na memória principal.
  - Páginas mais antigas estão no início e mais recentes no final

Algoritmos de Substituição de Páginas

- FIFO



Algoritmos de Substituição de Páginas

- LFU (Least-Frequently-Used)
  - Seleciona a página menos referenciada, ou seja, o frame menos utilizado.
  - Mantém um contador com o número de referências para cada página na memória.
  - A página que possuir o contador com o menor número de referências será escolhida.
  - É raramente implementado, servindo apenas de base para outros algoritmos de substituição.

Algoritmos de Substituição de Páginas

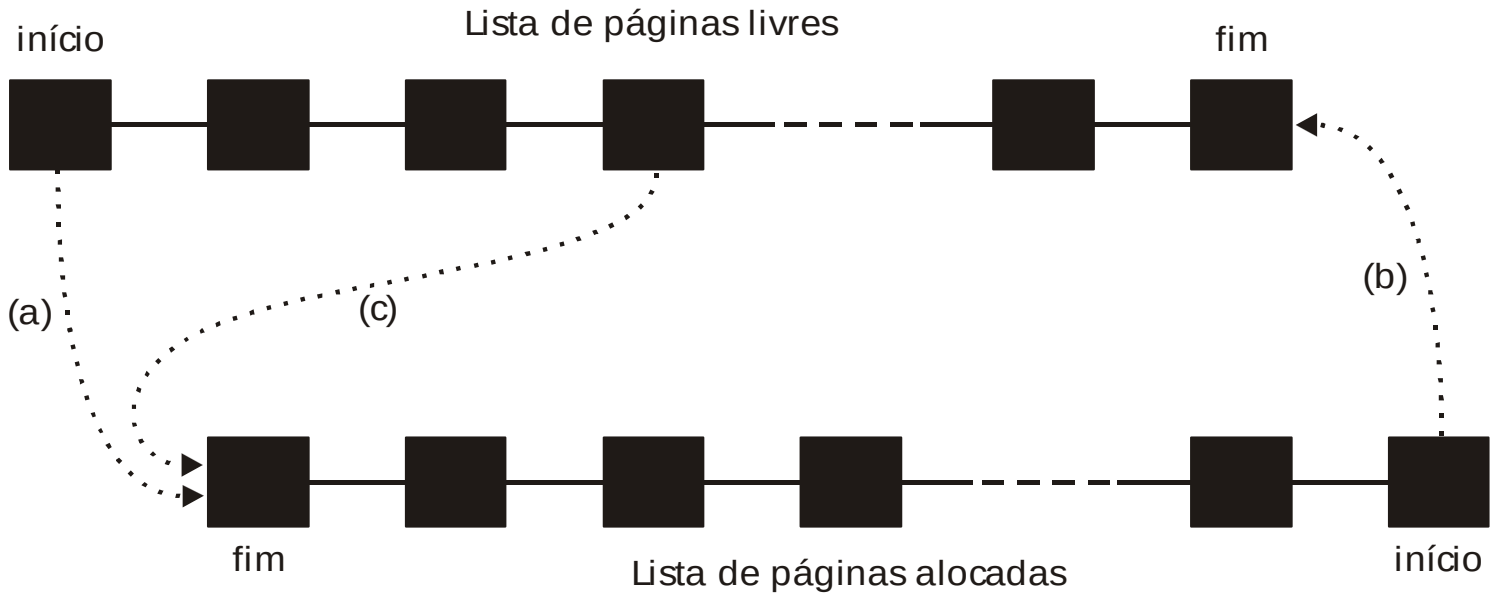
- LRU (Least-Recently-Used)
  - Seleciona a página que está na memória há mais tempo sem ser referenciada.
  - Princípio da Localidade (não foi usada recentemente não será futuramente).
  - É necessário que cada página tenha associado o momento do último acesso, que deve ser atualizado a cada referência a um frame.
  - Alto custo de implementação

Algoritmos de Substituição de Páginas

- FIFO com Buffer de Páginas
  - Combina uma lista de páginas alocadas (LPA) com uma lista de páginas livres (LPL).
  - A LPA organiza as páginas há mais tempo na memória principal (início da lista).
  - A LPL organiza os frames livres na memória principal → Frame livre a mais tempo no início da lista.

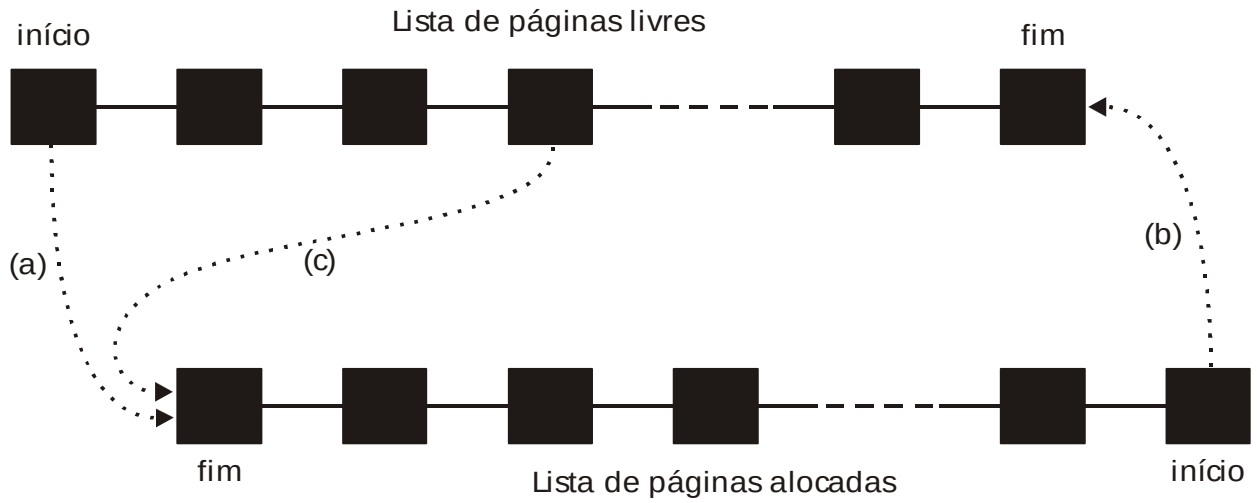
Algoritmos de Substituição de Páginas

- FIFO com Buffer de Páginas
  - Sempre que um processo necessita alocar uma nova página, o sistema utiliza a primeira página da LPL, colocando-a no final da LPA.



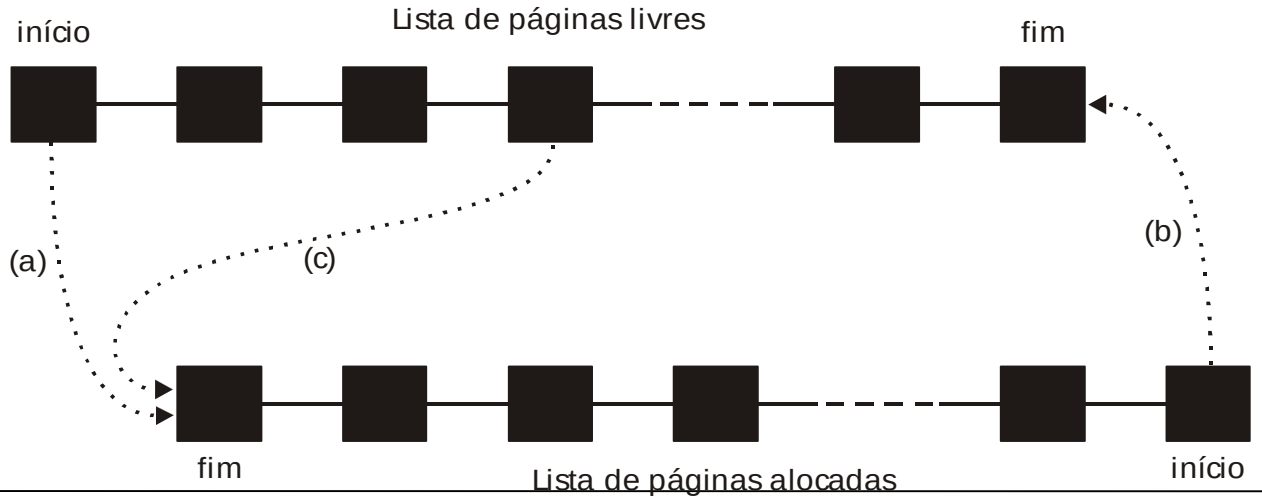
Algoritmos de Substituição de Páginas

- FIFO com Buffer de Páginas
  - Caso o processo tenha que liberar uma página, o mecanismo de substituição seleciona o frame em uso há mais tempo na memória, isto é, o primeiro da LPA, colocando-o no final da LPL



Algoritmos de Substituição de Páginas

- FIFO com Buffer de Páginas
  - a página que entrou na LPL continua disponível na memória por um determinado intervalo de tempo.
  - Caso esta página seja novamente referenciada e ainda não tenha sido alocada, basta retirá-la da LPL e devolvê-la ao processo.

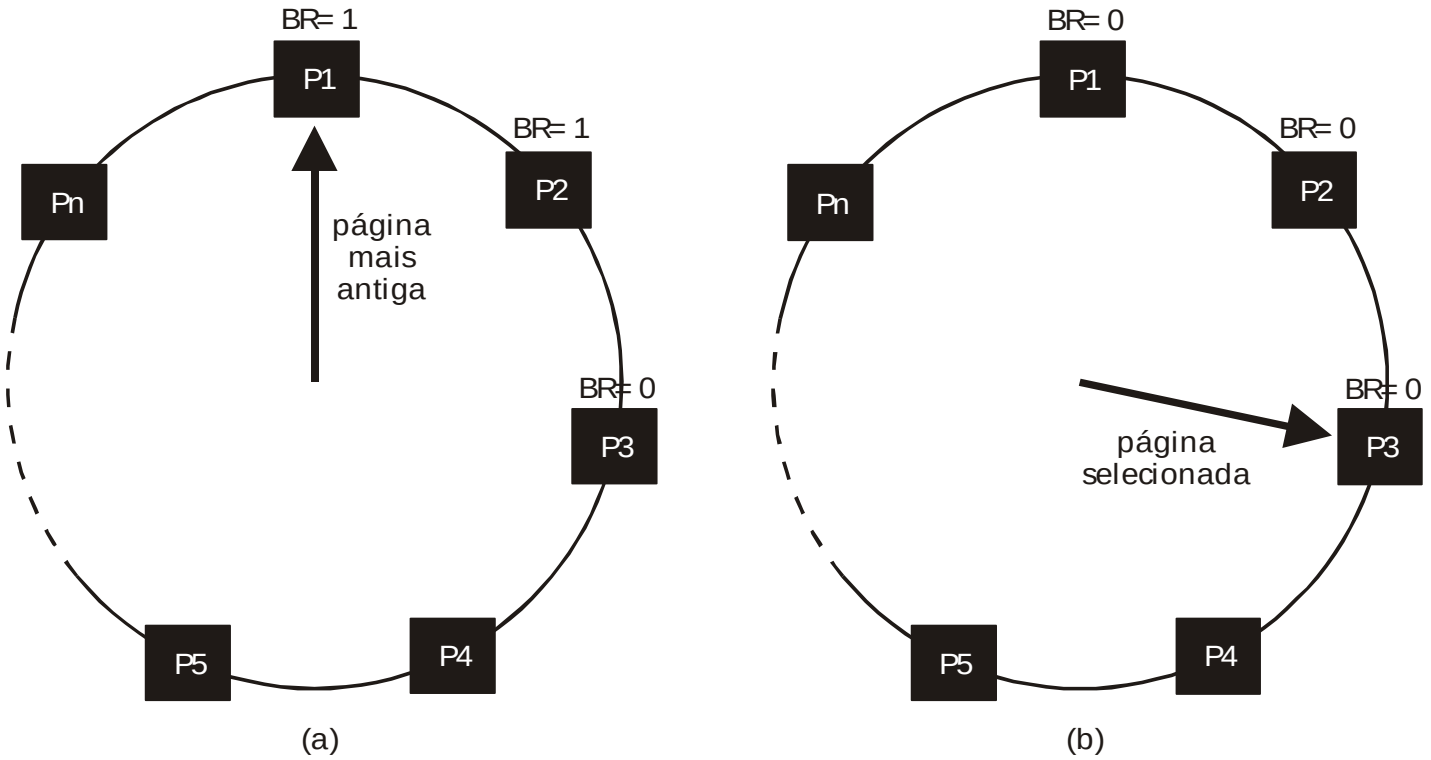


Algoritmos de Substituição de Páginas

- FIFO Circular (Clock)
  - Utiliza como base o FIFO, porém as páginas alocadas na memória estão em uma estrutura de lista circular, semelhante a um relógio.
  - Há um ponteiro que guarda a posição da página mais antiga na lista.
  - Cada página possui associado um bit de referência (BR), indicando se a página foi recentemente referenciada.

Algoritmos de Substituição de Páginas

- FIFO Circular (Clock)



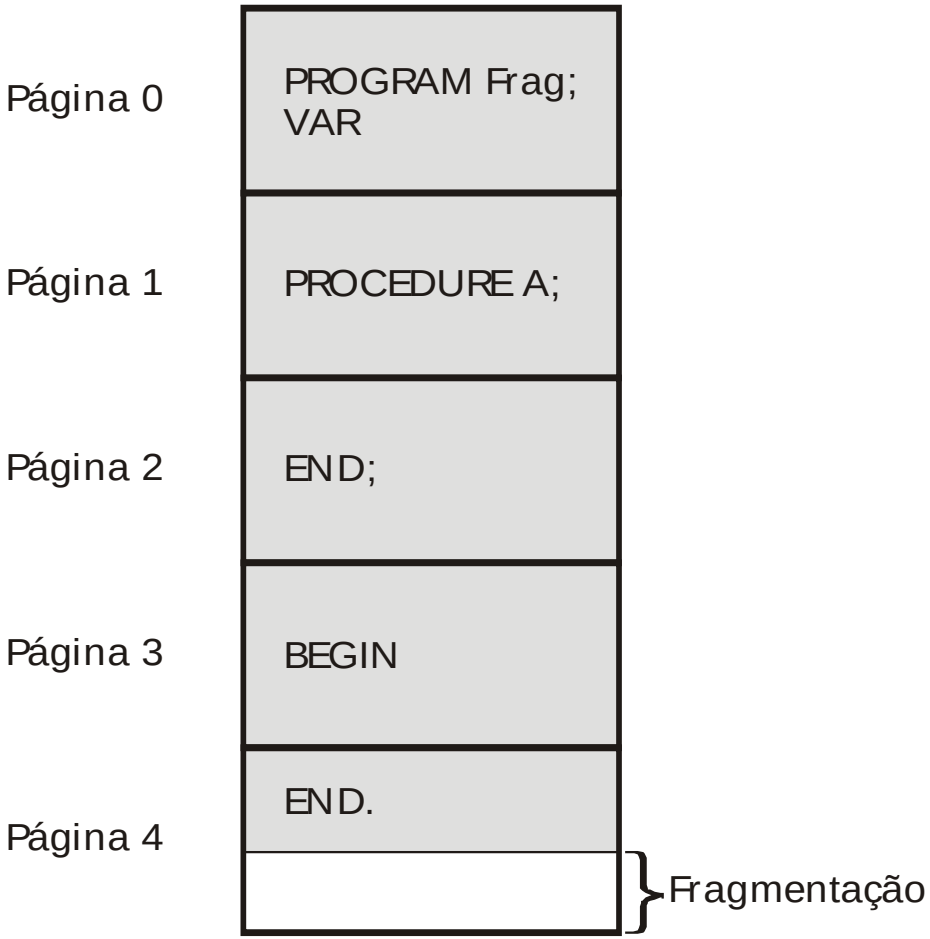
Tamanho de Página

- O tamanho da página tem impacto direto sobre o número de entradas na tabela de páginas e consequentemente, no tamanho da tabela e no espaço ocupado na memória principal.

- Apesar de páginas grandes tornarem menor o tamanho das tabelas de páginas, ocorre o problema da fragmentação interna.

Tamanho de Página

- Fragmentação interna.



Tamanho de Página

- O tamanho da página tem impacto direto sobre o número de entradas na tabela de páginas e consequentemente, no tamanho da tabela e no espaço ocupado na memória principal.

- Apesar de páginas grandes tornarem menor o tamanho das tabelas de páginas, ocorre o problema da fragmentação interna.

- Quanto maior o tamanho de página, maiores as chances de ter na memória código pouco referenciado, ocupando espaço desnecessariamente.

- Tempos de leitura e gravação na memória secundária.

Translation Lookaside Buffer

- O mapeamento implica pelo menos dois acessos à memória principal: o primeiro à tabela de páginas e o outro à própria página.

- Sempre que um endereço virtual precisa ser traduzido, a tabela de mapeamento deve ser consultada para se obter o endereço do frame e, posteriormente, acessar o dado na memória principal.

- O TLB é uma memória especial que funciona como uma memória cache, mantendo apenas as traduções dos endereços virtuais das páginas mais recentemente referenciadas.

Translation Lookaside Buffer

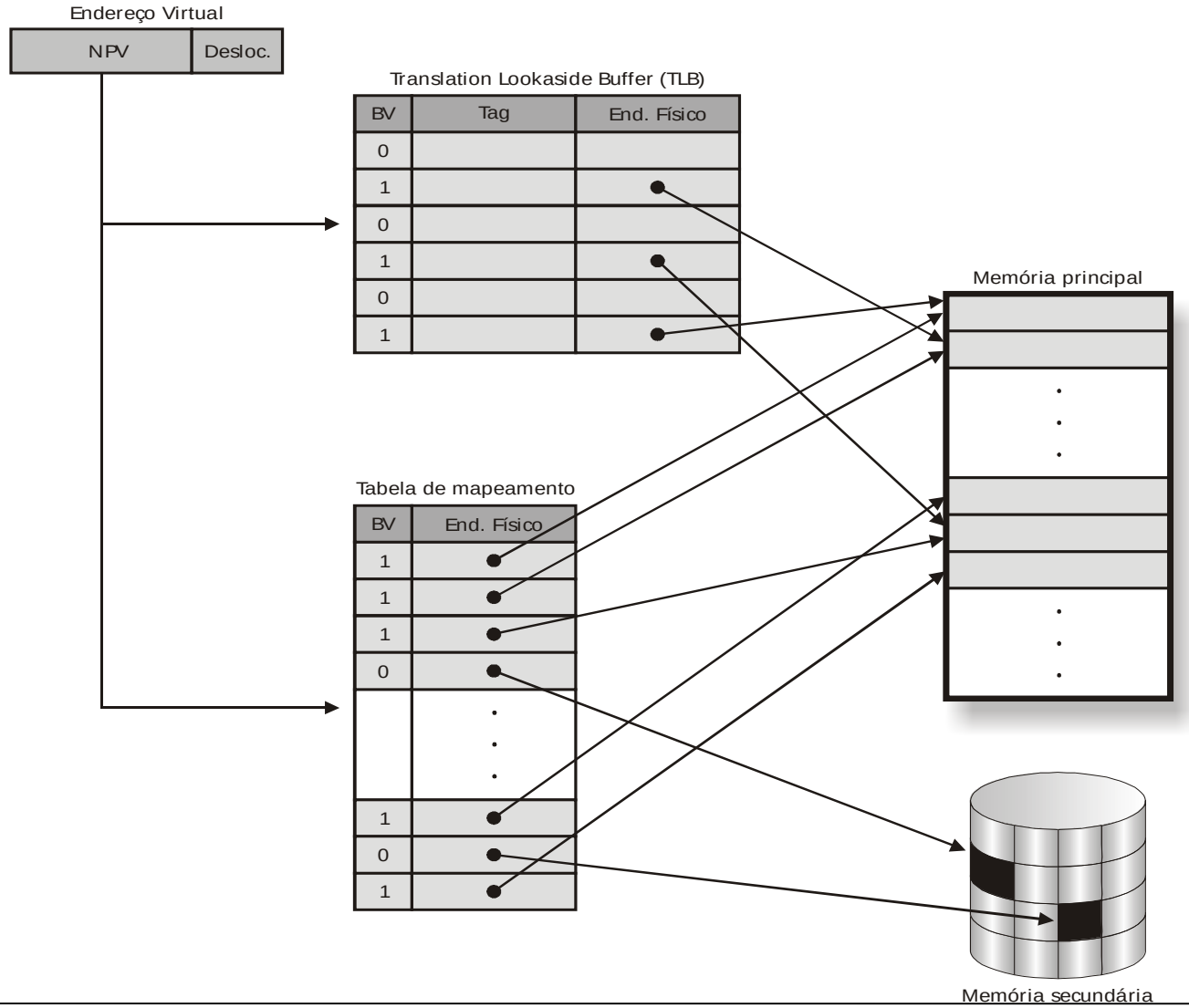
- O TLB utiliza o esquema de mapeamento associativo, que permite verificar simultaneamente em todas as suas entradas a presença do endereço virtual.

- Na tradução de um endereço virtual → TLB → Tabela de mapeamento → se "page fault" → carrega da memória secundária.

- TLB hit: endereço virtual (tag) está em cache.

- TLB miss: endereço virtual não está na cache.

Translation Lookaside Buffer



Translation Lookaside Buffer

- Campos da TLB

Campo	Descrição
Tag	Endereço virtual sem o deslocamento.
Modificação	Bit que indica se a página foi alterada.
Referência	Bit que indica se a página foi recentemente referenciada, sendo utilizada para a realocação de entradas na TLB.
Proteção	Define a permissão de acesso à página.
Endereço físico	Posição do frame na memória principal.

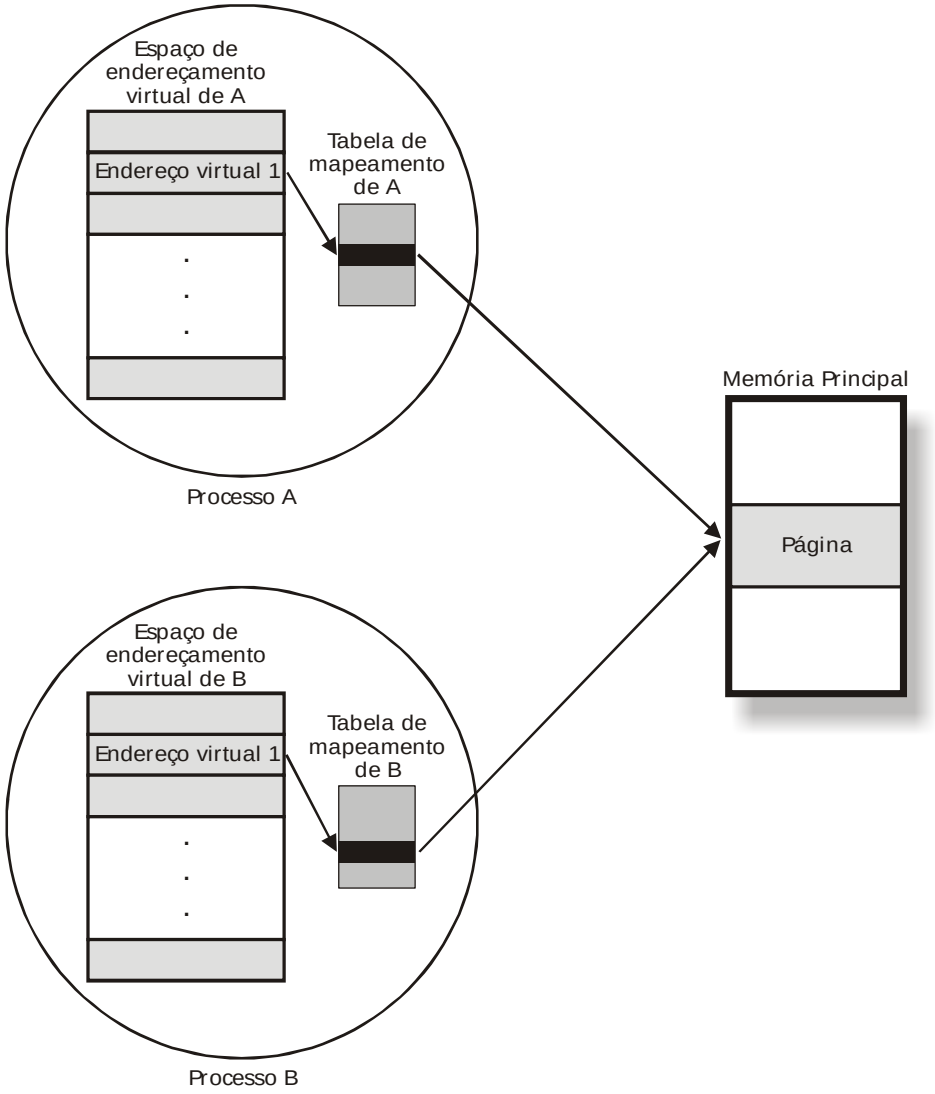
Compartilhamento de Memória

- Em sistemas que implementam memória virtual → implementação da reentrância → compartilhamento de código entre diversos processos.

- Para isso, basta que as entradas das tabelas de mapeamento dos processos apontem para os mesmos frames na memória principal.

- Apesar de os processos compartilharem as mesmas páginas de código, cada um possui sua própria área de dados em páginas independentes.

Compartilhamento de Memória



Compartilhamento de Memória

- O mecanismo de paginação permite que processos façam o mapeamento de uma mesma área na memória.

- A única preocupação da aplicação é garantir o sincronismo no acesso à região compartilhada, evitando problemas de inconsistência.

Memória Virtual por Segmentação

- É a técnica de gerência de memória onde o espaço de endereçamento virtual é dividido em blocos de tamanhos diferentes chamados segmentos.

- Na técnica de segmentação, um programa é dividido logicamente em sub-rotinas e estruturas de dados, que são alocadas em segmentos na memória principal.

Memória Virtual por Segmentação

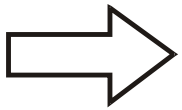
- Segmentação

```
PROGRAM Segmento;
    VAR A: ARRAY...
    C: ...
PROCEDURE X;
END;
FUNCTION Y;

END;

BEGIN

END.
```



Memória Virtual por Segmentação

- Na técnica de paginação o programa é dividido em páginas de tamanho fixo, sem qualquer ligação com sua estrutura.

- Na segmentação existe uma relação entre a lógica do programa e sua alocação na memória principal.

- A definição dos segmentos é realizada pelo compilador.

- A partir do código-fonte do programa, e cada segmento pode representar um procedimento, função, vetor ou pilha.

Memória Virtual por Segmentação

- O espaço de endereçamento virtual de um processo possui um número máximo de segmentos que podem existir.

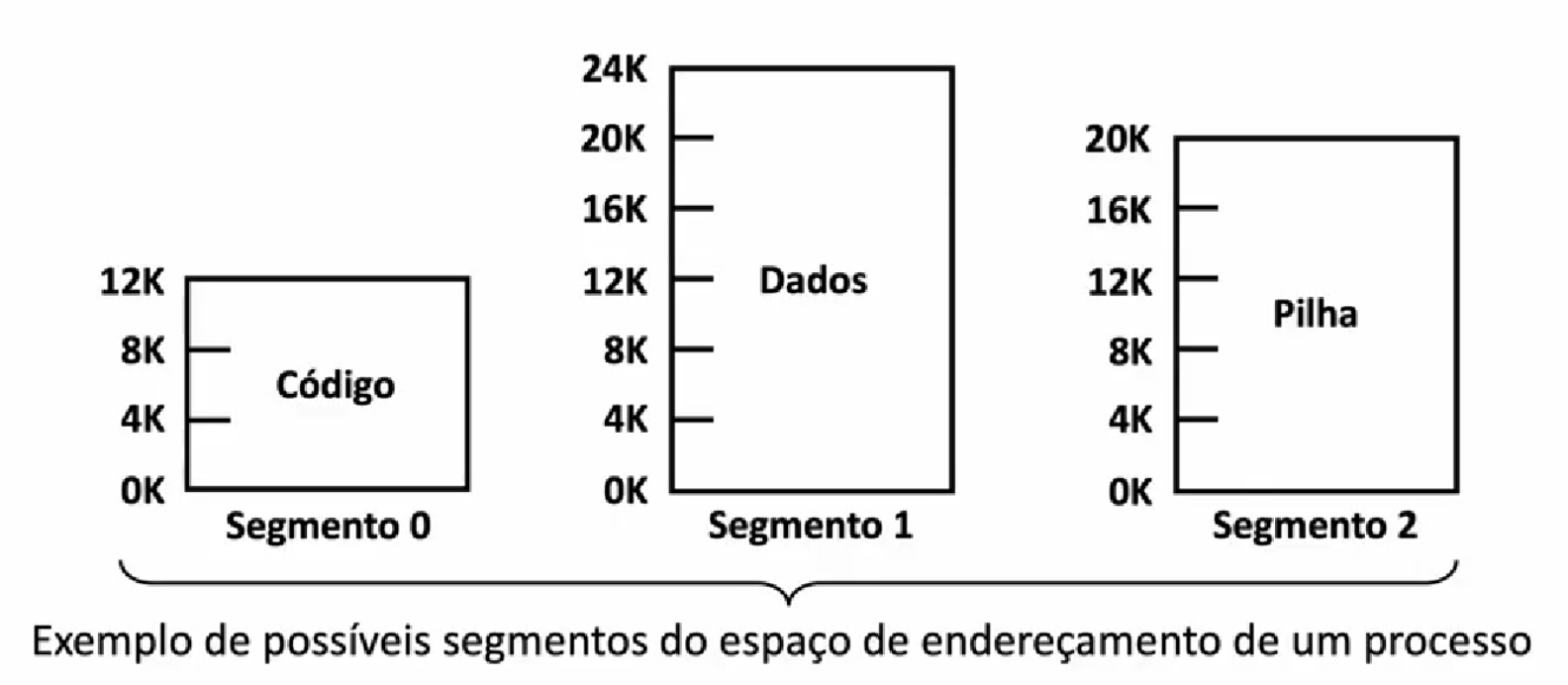
- O tamanho do segmento pode ser alterado durante a execução do programa, facilitando a implementação de estruturas de dados dinâmicas.

- Espaços de endereçamentos independentes → alteração em sub-rotinas → sem necessidade do programa principal e todas as suas sub-rotinas serem recompiladas.

- Isso não ocorrem em sistemas com paginação.

Memória Virtual por Segmentação

- Segmentação



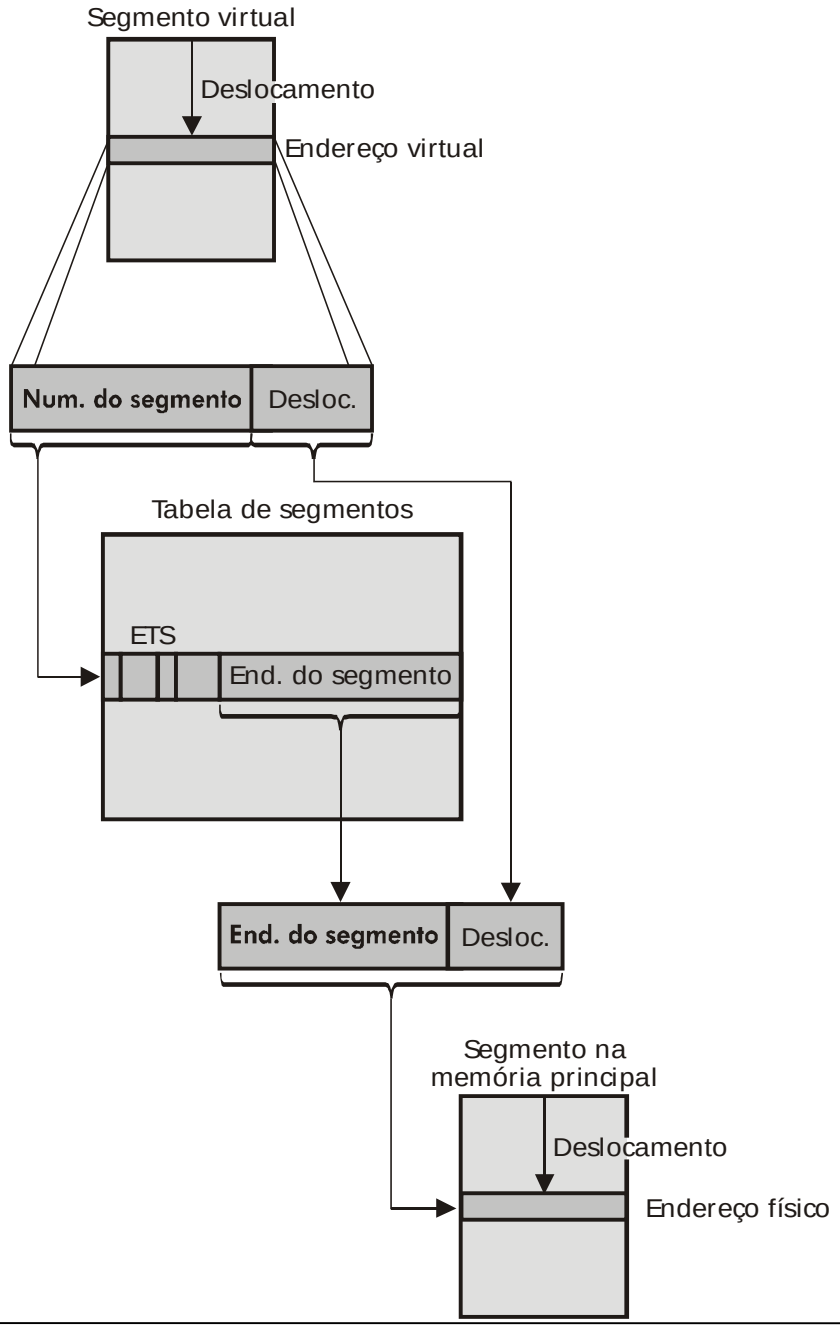
Exemplo de possíveis segmentos do espaço de endereçamento de um processo

Memória Virtual por Segmentação

- O mecanismo de mapeamento é muito semelhante ao de paginação.
- Os segmentos são mapeados através de tabelas de mapeamento de segmentos (TMS).
- Os endereços são compostos pelo número do segmento virtual (NSV) e por um deslocamento.
- O NSV identifica unicamente o segmento virtual que contém o endereço, funcionando como um índice na TMS.

Memória Virtual por Segmentação

- O deslocamento indica a posição do endereço virtual em relação ao início do segmento no qual se encontra.


Segmento virtual
Deslocamento
Endereço virtual
Num. do segmento
Desloc.
Tabela de segmentos
ETS
End. do segmento
End. do segmento
Desloc.
Segmento na
memória principal
Deslocamento
Endereço físico

Memória Virtual por Segmentação

- Campos da ETS

Campo	Descrição
Tamanho	Especifica o tamanho do segmento.
Bit de validade	Indica se o segmento está na memória principal.
Bit de modificação	Indica se o segmento foi alterado.
Bit de referência	Indica se o segmento foi recentemente referenciado, sendo utilizado pelo algoritmo de substituição.
Proteção	Indica a proteção do segmento.

Memória Virtual por Segmentação

- Uma grande vantagem da segmentação em relação à paginação é a sua facilidade em lidar com estruturas de dados dinâmicas.

- O tamanho do segmento pode ser facilmente alternado na ETS.

- Estruturas de dados, como pilhas e listas encadeadas, podem aumentar e diminuir dinamicamente.

- Na paginação a expansão de um vetor implica na alocação de novas páginas → mudanças → ajuste na tabela de paginação.

Memória Virtual por Segmentação

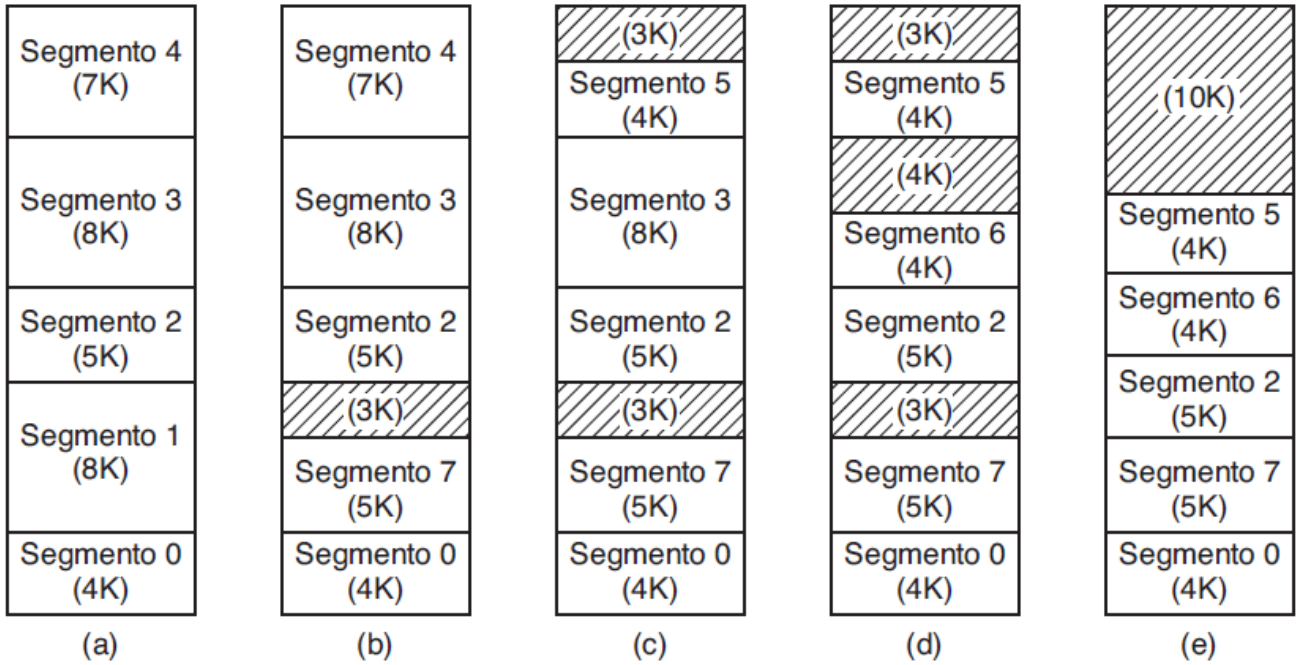
- Apenas os segmentos referenciados são transferidos da memória secundária para a memória principal.
- Para que a segmentação funcione de forma eficiente, os programas devem estar bem modularizados.
- Para alocar segmentos → o SO mantém uma tabela com as áreas livres e ocupadas da memória.
- Quando um novo segmento é referenciado, o sistema seleciona o espaço livre suficiente → carregar o segmento → memória.

Memória Virtual por Segmentação

- Na paginação ocorre o problema da fragmentação interna.

- Na segmentação surge o problema da fragmentação externa.

(a)-(d) Desenvolvimento da fragmentação externa. (e) Remoção da fragmentação externa.



Memória Virtual por Segmentação

- Na paginação ocorre o problema da fragmentação interna.
- Na segmentação surge o problema da fragmentação externa.
- Em sistemas com segmentação → proteção de memória é mais simples → cada segmento possui um conteúdo bem definido.
- O compartilhamento de memória é mais simples que na paginação → a tabela mapeia estruturas lógicas e não páginas.

Memória Virtual por Segmentação

- Paginação x segmentação

Característica	Paginação	Segmentação
Tamanho dos blocos de mémoria	Iguais	Diferentes
Proteção	Complexa	Mais simples
Compartilhamento	Complexo	Mais simples
Estruturas de dados dinâmicas	Complexo	Mais simples
Fragmentação interna	Pode existir	Não existe
Fragmentação externa	Não existe	Pode existir
Programação modular	Dispensável	Indispensável
Alteração do programa	Mais trabalhosa	Mais simples

Memória Virtual por Segmentação com Paginação

- Técnica de gerência de memória na qual o espaço de endereçamento é dividido em segmentos e, por sua vez, cada segmento dividido em páginas.

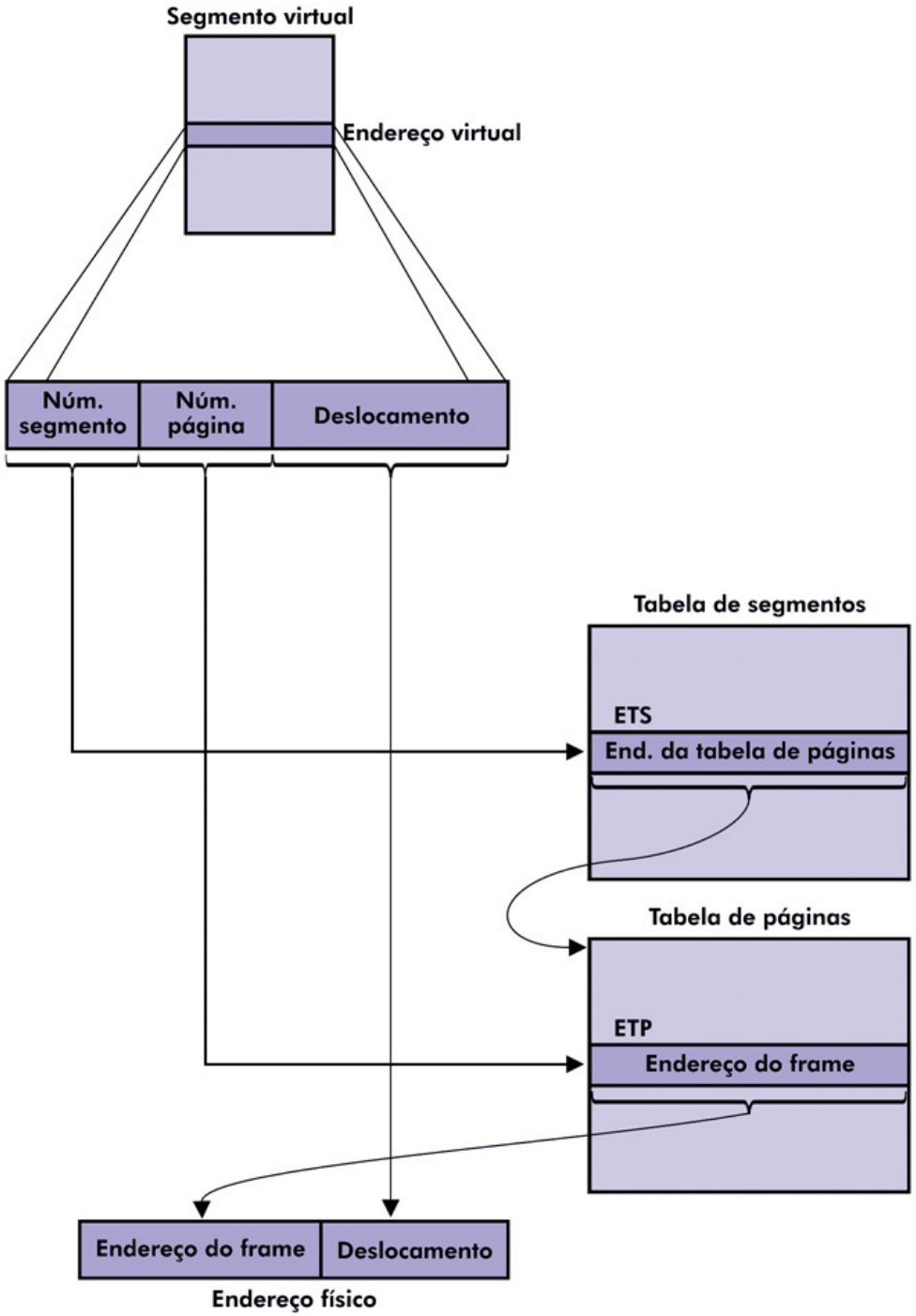
- um endereço virtual é formado pelo número do segmento virtual (NSV), um número de página virtual (NPV) e um deslocamento.

- Através do NSV, obtém-se uma entrada na tabela de segmentos, que contém informações da tabela de páginas do segmento.

Memória Virtual por Segmentação com Paginação

- O NPV identifica unicamente a página virtual que contém o endereço, funcionando como um índice na tabela de páginas.

- O deslocamento indica a posição do endereço virtual em relação ao início da página na qual se encontra.


Segmento virtual
Endereço virtual
Núm. segmento
Núm. página
Deslocamento
Tabela de segmentos
ETS
End. da tabela de páginas
Tabela de páginas
ETP
Endereço do frame
Endereço do frame
Deslocamento
Endereço físico

Memória Virtual por Segmentação com Paginação

- A aplicação continua sendo mapeada em segmentos de tamanhos diferentes.

- Por outro lado, o sistema trata cada segmento como um conjunto de páginas de mesmo tamanho.

- Um segmento não precisa estar contíguo na memória principal, eliminando o problema da fragmentação externa encontrado na segmentação pura.

Thrashing

- Excessiva transferência de páginas e/ou segmentos entre a memória principal e memória secundária.
- Esse problema está presente em sistemas que implementam tanto paginação como segmentação.
- Ocorre em dois níveis:
  - no próprio processo
  - no sistema

Thrashing

- No nível do processo, a excessiva paginação ocorre devido ao elevado número de page faults gerado pelo programa em execução.

- No nível do sistema ocorre quando existem mais processos competindo por memória principal que espaço disponível.

- O thrashing em sistemas que implementam segmentação também ocorre em dois níveis.

Thrashing

- Se existirem mais processos para serem executados que memória real disponível, a única solução é a expansão da memória principal.

- Este problema não ocorre apenas em sistemas que implementam memória virtual, mas também em sistemas com outros mecanismos de gerência de memória