Gerenciamento de memória

Prof. Marcos Ribeiro Quinet de Andrade
Universidade Federal Fluminense – UFF
Instituto de Ciência e Tecnologia – ICT

Gerenciamento de Memória

- Hierarquia de Memória
- Alocação particionada estática e dinâmica
- Gerenciamento dos espaços livres
- *Swapping*
- Memória virtual
  - Paginação e segmentação

Gerenciamento de Memória

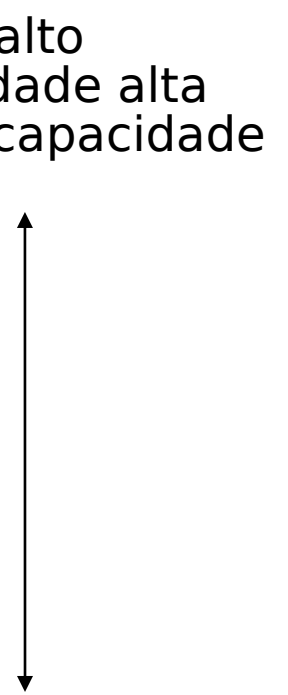
- Memória – recurso muito importante;
- Tendência atual do software
  - Lei de *Parkinson*: “Os programas se expandem para preencher a memória disponível para eles” (adaptação);
- Requisitos:
  - Muito grande;
  - Rápida;
  - Não volátil;
  - Baixo custo.

Gerenciamento de Memória

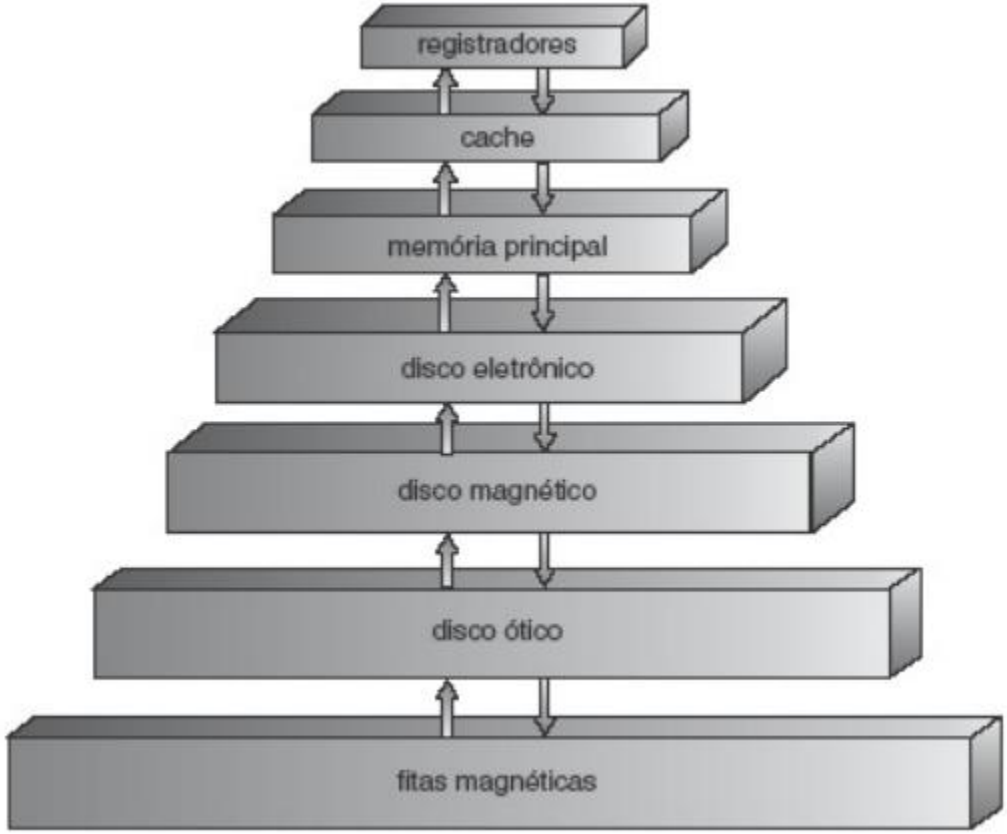
- Hierarquia de Memória:
  - Registradores
  - Cache – vários sub-níveis
  - Memória Principal
  - Memórias secundárias (Disco e fita magnética, memória SSD, mídias ópticas)

Hierarquia de Memória

Custo alto
Velocidade alta
Baixa capacidade



Custo baixo
Velocidade baixa
Capacidade elevad



Hierarquia de Memória

- Cache
  - Pequena quantidade – K/Megabytes (por exemplo, um Intel i7 de sétima geração tem 12 MB de cache L3 compartilhada, mas apenas 256 KB de cache L2 para cada *core*);
  - Alto custo por byte;
  - Muito rápida;
  - Volátil.
- Memória Principal
- Memória Secundária

Hierarquia de Memória

- Cache
- Memória Principal
  - Quantidade intermediária – Mega/Gigabytes;
  - Custo médio por byte;
  - Velocidade média;
  - Volátil.
- Memória Secundária

Hierarquia de Memória

- Cache
- Memória Principal
- Memória Secundária
  - Grande quantidade – Giga/Terabytes;
  - Baixo custo por byte;
  - Lenta;
  - Não volátil.

Hierarquia de Memória

- Para cada tipo de memória:
  - Gerenciar espaços livres/ocupados;
  - Alocar processos/dados na memória;
  - Localizar dado.
- Entre os níveis de memória:
  - Gerenciar trocas de blocos de dados.

Gerenciamento de Memória

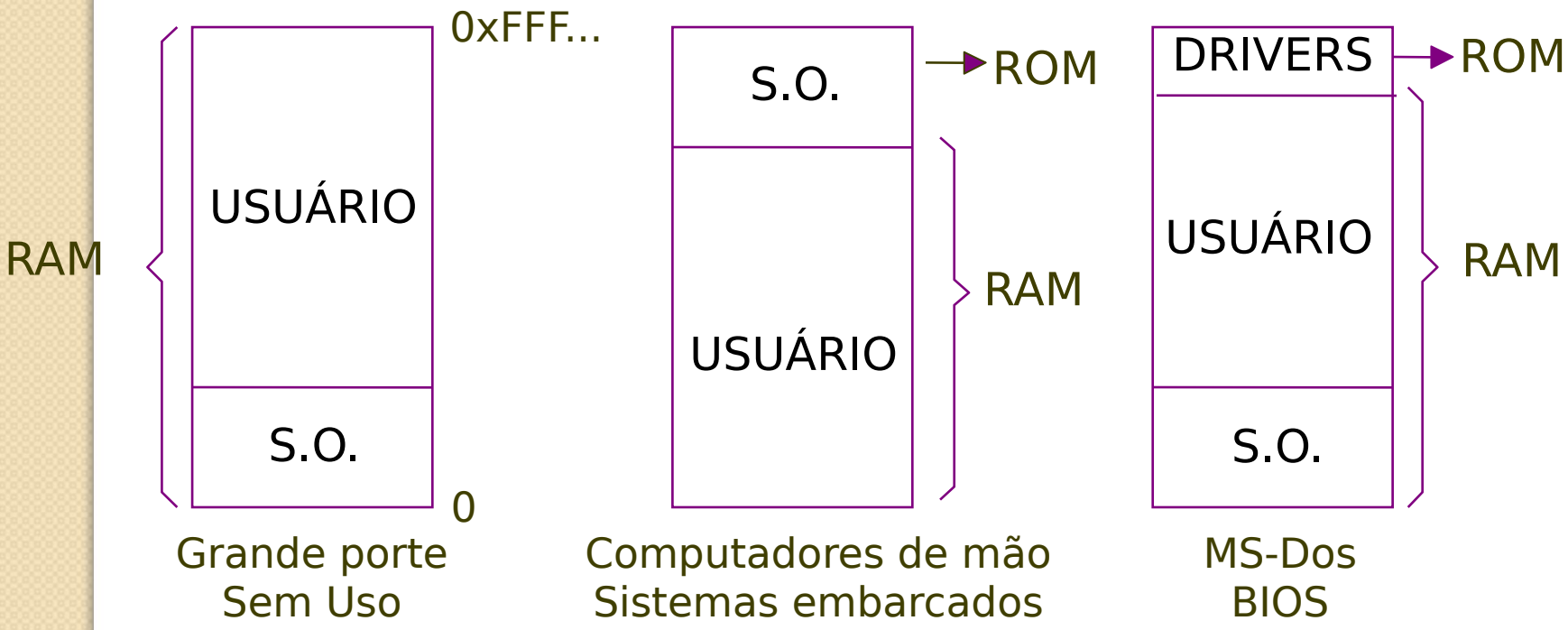
- **Gerenciador de memória**: responsável por alocar e liberar espaços na memória para os processos em execução; também responsável por gerenciar chaveamento entre os níveis de memória:
  - Memória principal e secundária;
  - Memória principal e cache.

Gerenciamento de Memória

- Tipos básicos de gerenciamento:
  - **Com paginação** (chaveamento): Processos são movidos entre a memória principal e o disco; artifício usado para resolver o problema da falta de memória;
    - Se existe MP suficiente não há necessidade de se ter paginação;
  - **Sem paginação**: não há chaveamento;

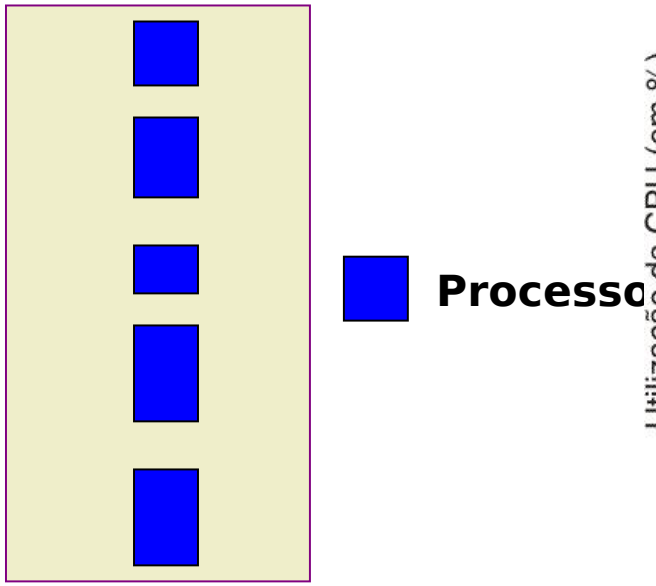
Gerenciamento de Memória

- Monoprogramação:
  - Sem paginação: gerenciamento mais simples;
- Apenas um processo na memória;

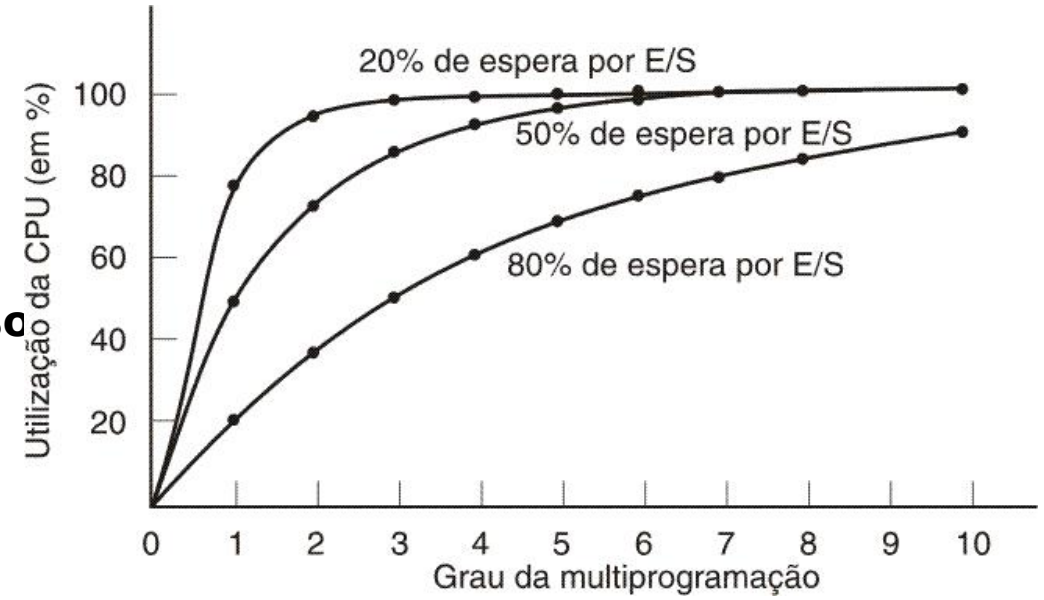


Gerenciamento de Memória

- Modelo de Multiprogramação:
  - Múltiplos processos sendo executados;
  - Eficiência da UCP;



Memória Principal



Necessidade de Particionamento da Memória Principal

Gerenciamento de Memória

- **Multiprogramação**
  - vários processos na memória:
  - como proteger os processos uns dos outros?
  - como proteger o *kernel* de todos os processos?
  - como tratar a realocação?

- Todas as soluções envolvem equipar a UCP com um hardware especial
  - **MMU** (***memory management unit***);
    - Mais detalhes adiante (slide 39)

Gerenciamento de Memória

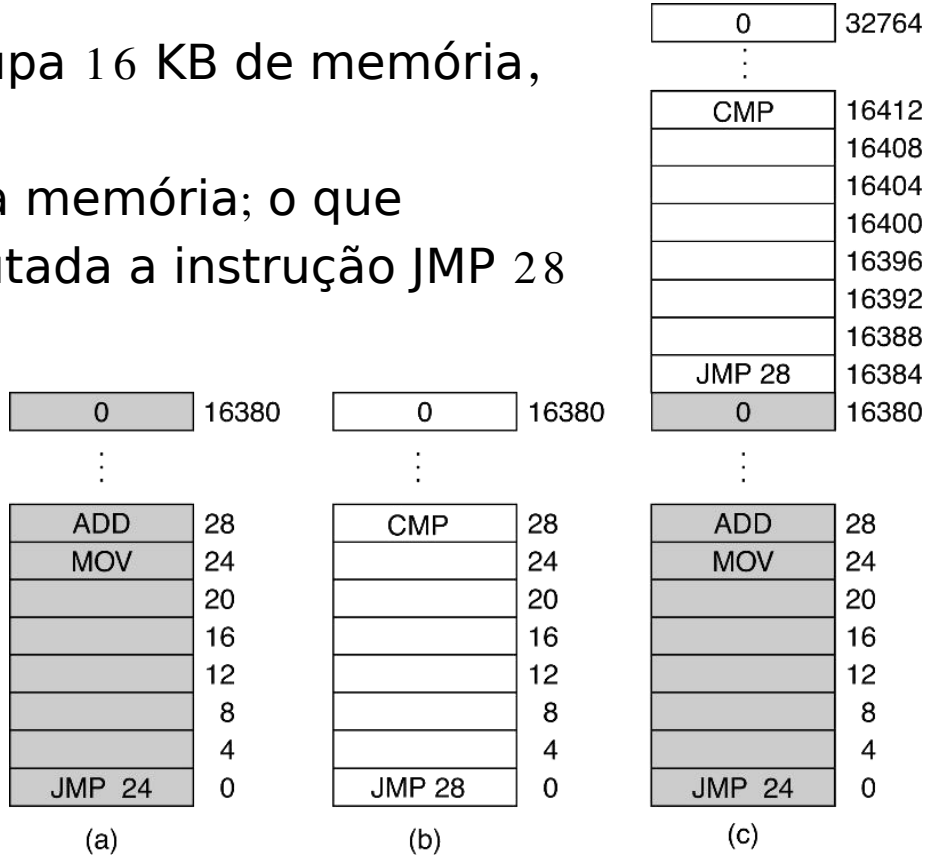
- **Multiprogramação**
  - vários processos na memória:
  - como proteger os processos uns dos outros?
  - como proteger o *kernel* de todos os processos?
  - como tratar a realocação?

- Todas as soluções envolvem equipar a UCP com um hardware especial
  - **MMU** (***memory management unit***);
    - Mais detalhes adiante (slide 39)

Gerenciamento de Memória

- **Sistemas sem abstração de memória**:

- (a) Programa P1, que ocupa 16 KB de memória, a partir do endereço 0
- (b) Programa P2, que ocupa 16 KB de memória, a partir do endereço 0
- (c) P1 e P2 carregados na memória; o que acontece quando executada a instrução JMP 28 de P2?



Gerenciamento de Memória

- A realocação estática simplesmente considera o primeiro endereço válido de uma área de memória atribuída a um programa como uma constante, que é somada a cada endereço de programa durante o processo de **carregamento**

- No exemplo anterior, o programa P2, carregado a partir do endereço 16384, a instrução "JMP 28" torna-se "JMP 16412"

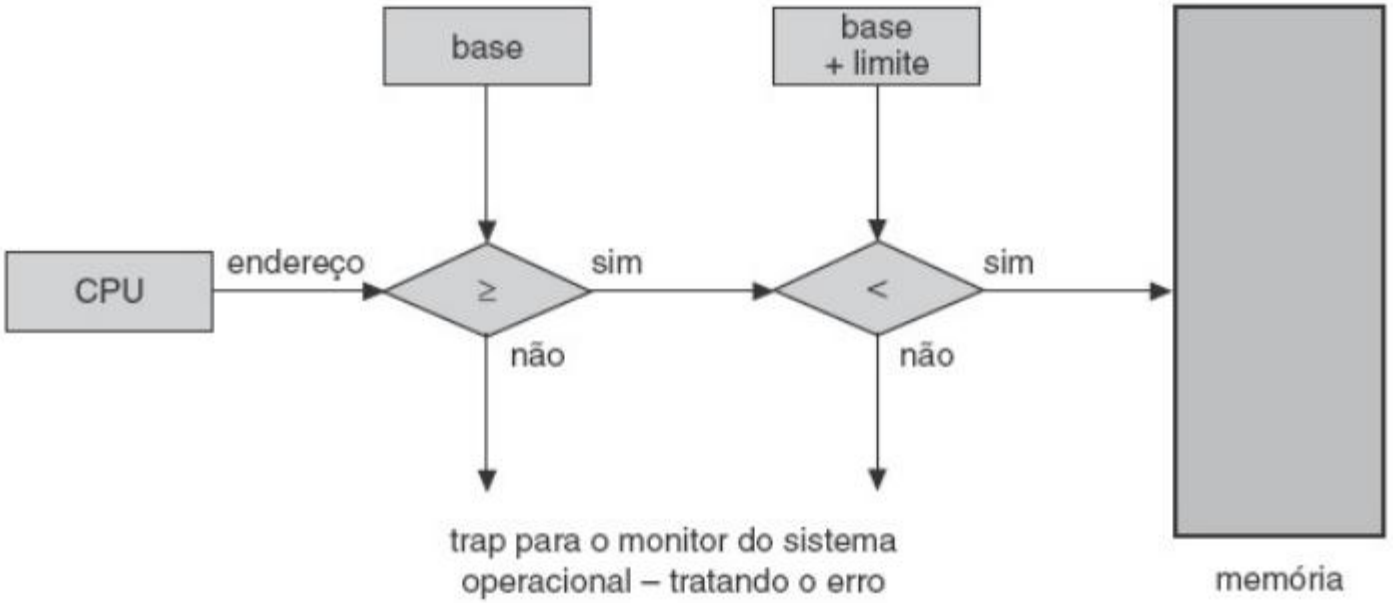
- Apesar de ser uma solução simples, não é considerada de funcionamento geral e torna o processo de carregamento mais lento

Gerenciamento de Memória

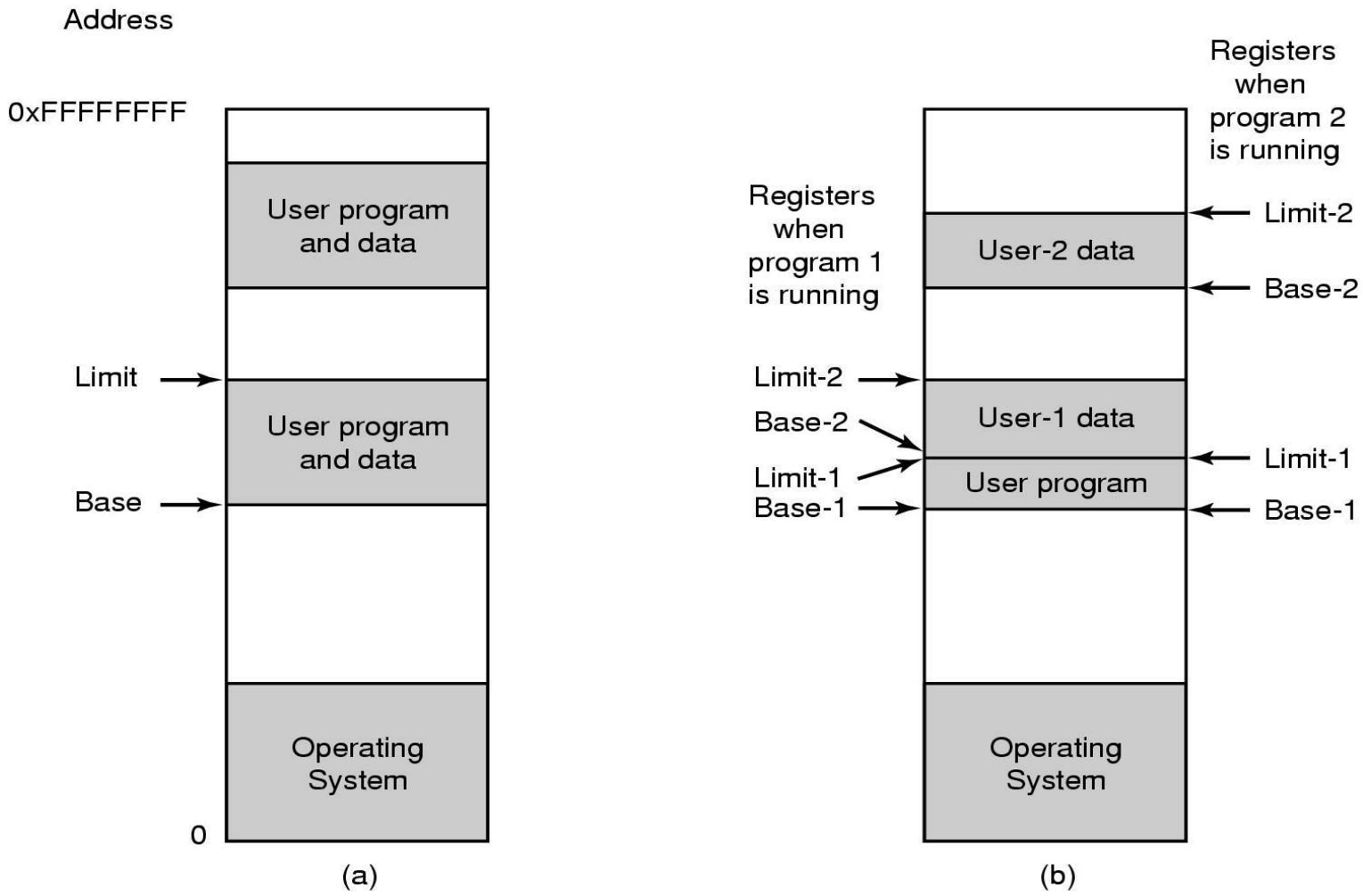
- A proteção na **relocação dinâmica** é feita em tempo de execução. Nesta proteção, são empregados um registrador base e um registrador limite
- O registrador base contém o valor do menor endereço físico. O registrador limite contém a faixa dos endereços lógicos
- Com registradores base e limite, cada endereço lógico deve ser menor que o conteúdo armazenado no registrador limite
- Esse endereço, se válido, é então calculado dinamicamente adicionando-se o valor contido no registrador base. O endereço calculado é então enviado à memória

Gerenciamento de Memória

Hardware de suporte para os registradores base e limite:



Gerenciamento de Memória – Registradores base e limite



Gerenciamento de Memória – Partições

- Particionamento da memória pode ser realizado de duas maneiras:
  - Partições fixas (alocação estática);
  - Partições variáveis (alocação dinâmica);

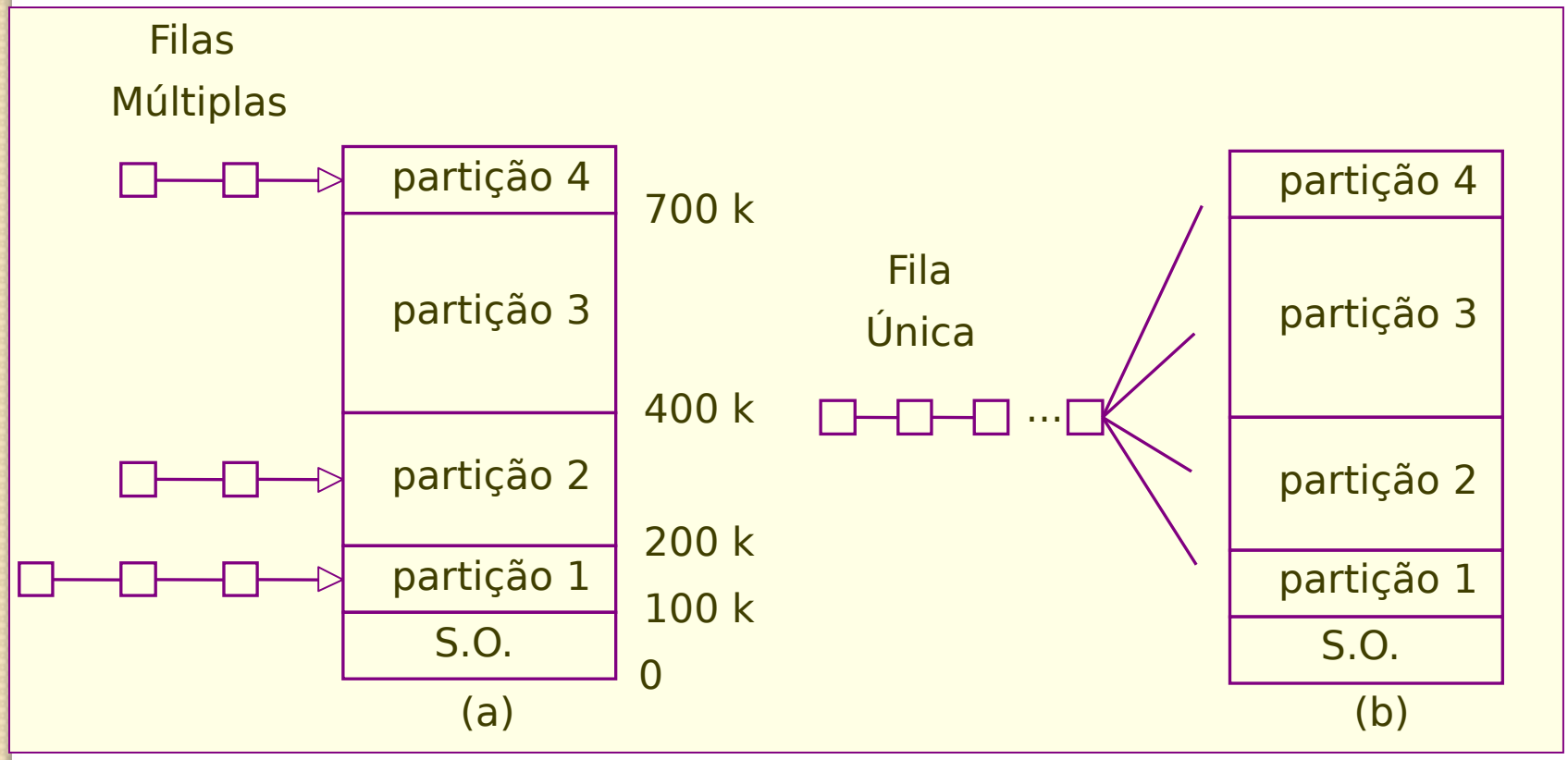
- **Partições Fixas**
  - Tamanho e número de partições são fixos (estáticos);
  - Não é atrativo, porque partições fixas tendem a desperdiçar memória (Qualquer espaço não utilizado é literalmente perdido)
  - Mais simples;

Gerenciamento de Memória – Partições Fixas

- Filas múltiplas:
  - Problema: filas não balanceadas;
- Fila única:
  - Melhor utilização da memória, pois procura o melhor processo para a partição considerada;
  - Problema: processos menores são prejudicados;

Gerenciamento de Memória – Partições Fixas

- Divisão da Memória em Partições Fixas:



Gerenciamento de Memória – Partições Fixas

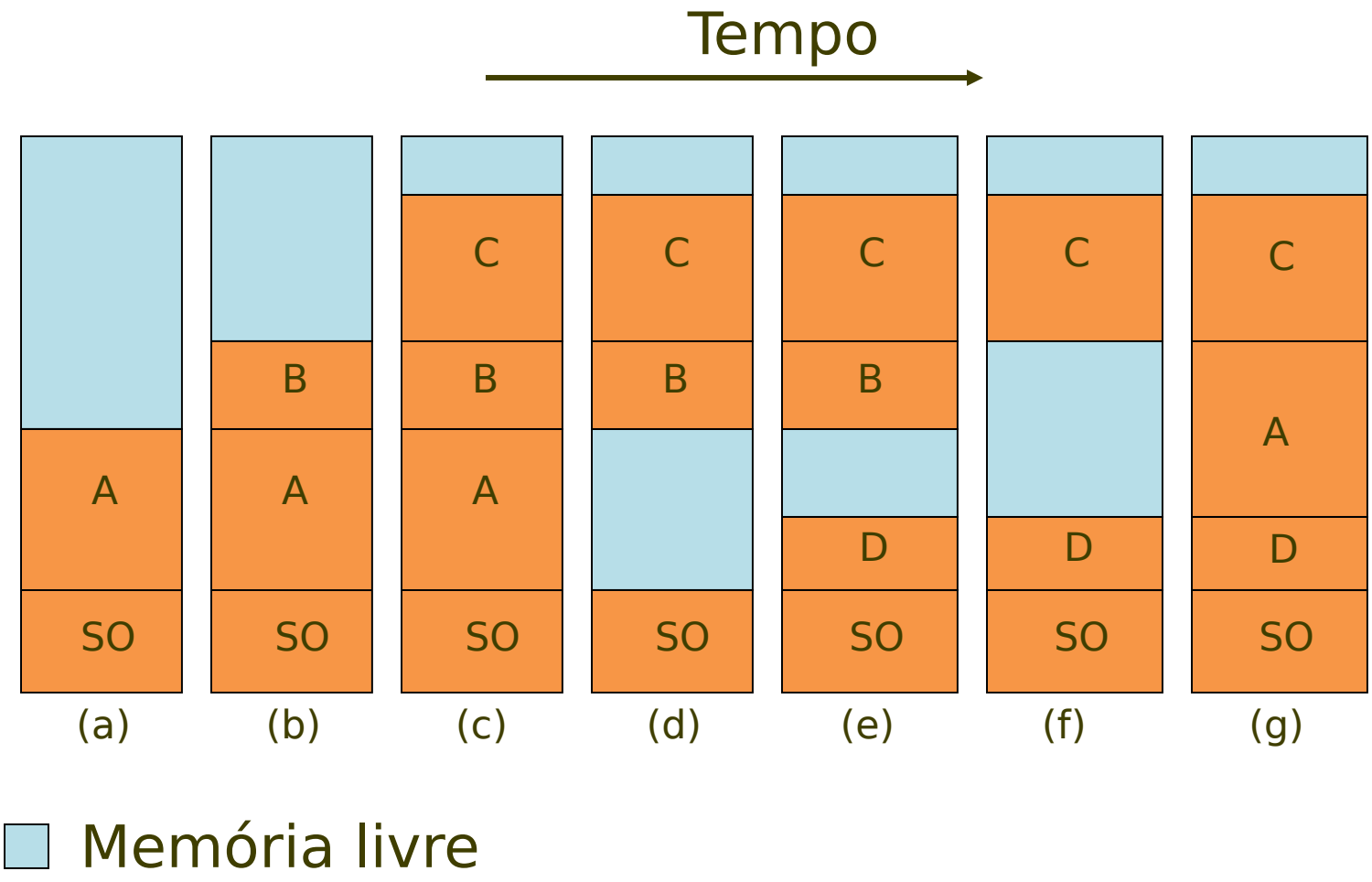
- Partições Fixas: problemas com fragmentação
  - **Interna**: desperdício dentro da área alocada para um processo;
    - Ex.: processo de tamanho 40K ocupando uma partição de 50k;
  - **Externa**: desperdício fora da área alocada para um processo;
    - Duas partições livres: PL1 com 25k e PL2 com 100k, e um processo de tamanho 110K para ser executado;
    - Livre: 125K, mas o processo não pode ser executado;

Gerenciamento de Memória – Partições Variáveis

- **Partições Variáveis**:
  - Tamanho e número de partições variam;
  - Otimiza a utilização da memória, mas complica a alocação e liberação da memória;
  - Partições são alocadas dinamicamente;
  - SO mantém na memória uma lista com os espaços livres;
  - Menor fragmentação interna e grande fragmentação externa;
    - Solução: Compactação;

Gerenciamento de Memória – Partições Variáveis

- Partições Variáveis:



Gerenciamento de Memória

- Minimizar espaço de memória inutilizados
  - Compactação: necessária para recuperar os espaços perdidos por fragmentação; no entanto, muito custosa para a UCP;
- Técnicas para alocação dinâmica de memória
  - *Bitmaps*;
  - Listas Encadeadas;

Gerenciamento de Memória

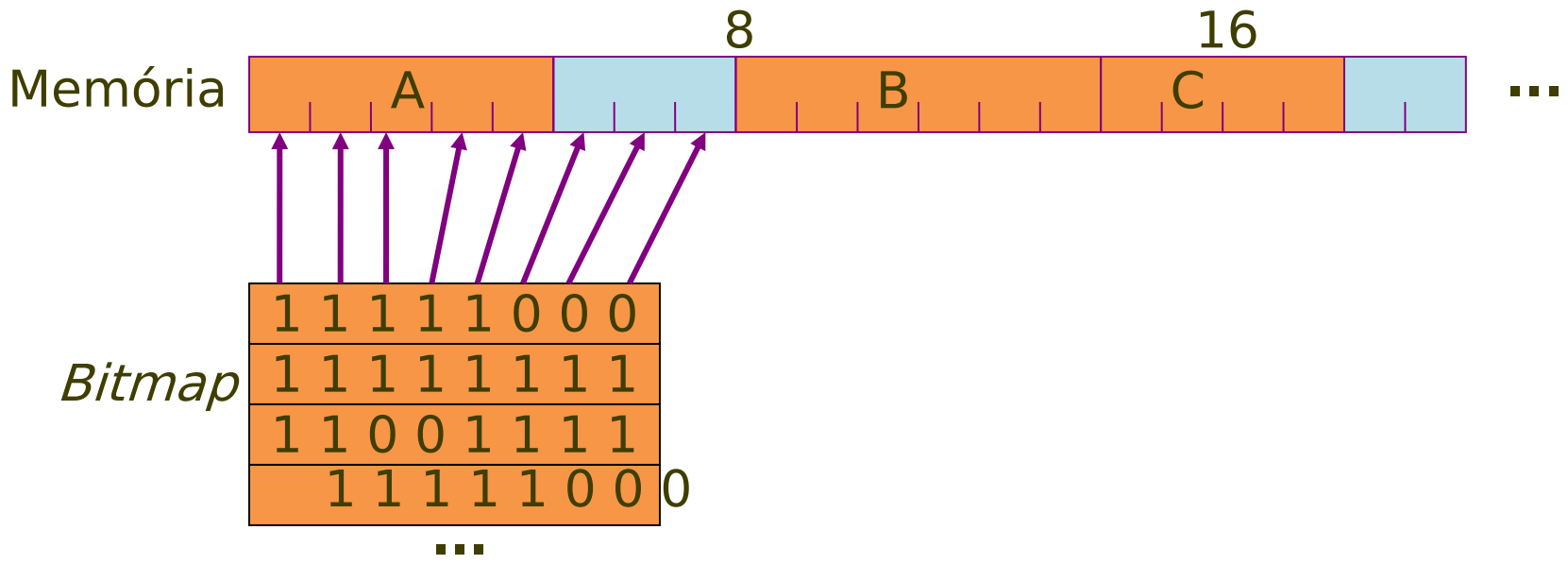
- Técnica com *Bitmaps*
  - Memória é dividida em unidades de alocação em kbytes;
  - Cada unidade corresponde a um *bit* no *bitmap*:

    0 → livre

    1 → ocupado
  - Tamanho do *bitmap* depende do tamanho da unidade e do tamanho da memória;
  - Ex.:
    - unidades de alocação pequenas → *bitmap* grande;
    - unidades de alocação grandes → perda de espaço;

Gerenciamento de Memória

- Técnica com *Bitmaps*:



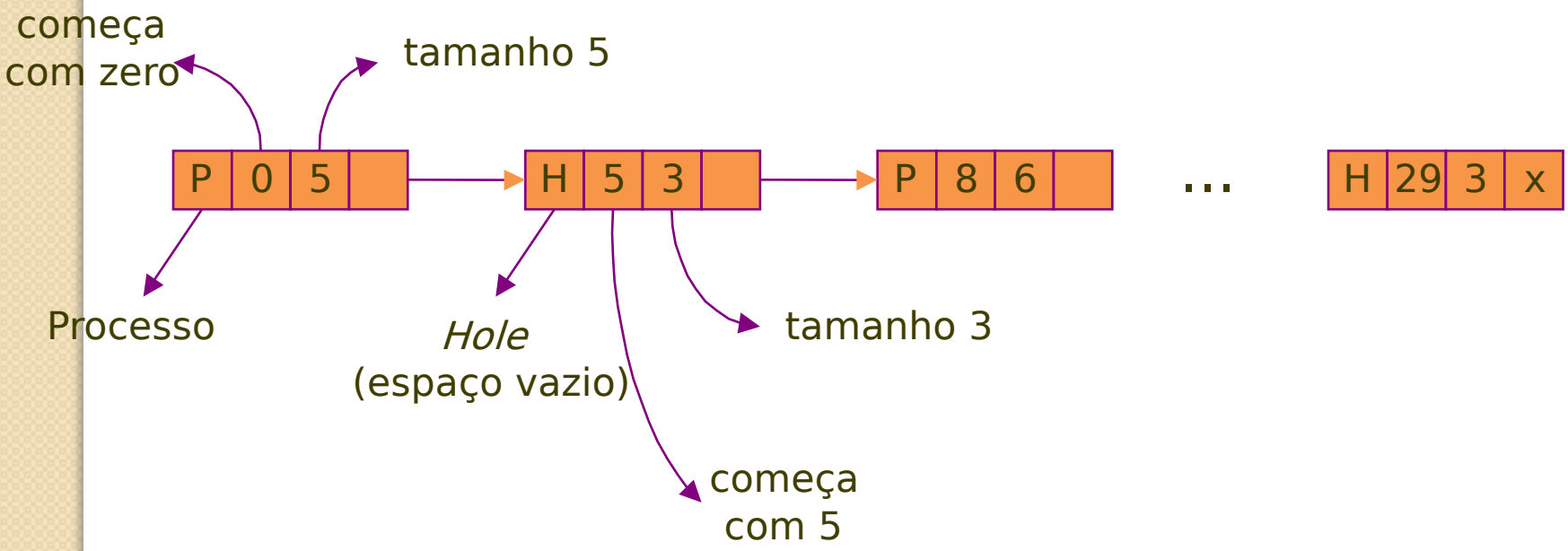
Memória ocupada

Memória livre

Gerenciamento de Memória

- Técnica com Listas Encadeadas:
  - Uma lista para os espaços vazios e outra para os espaços cheios, ou uma lista para ambos!

"espaço → segmento"



Gerenciamento de Memória

- Algoritmos de Alocação: quando um novo processo é criado:
  - *FIRST FIT*
    - 1º segmento é usado;
    - Rápido, mas pode desperdiçar memória por fragmentação;
  - *NEXT FIT*
    - 1º segmento é usado;
    - Mas na próxima alocação inicia busca do ponto que parou anteriormente;
    - Possui desempenho inferior;

Gerenciamento de Memória

- *BEST FIT*
  - Procura na lista toda e aloca o espaço que mais convém;
  - Menor fragmentação;
  - Mais lento;
- *WORST FIT*
  - Aloca o maior espaço disponível;
  - Gera ‘fragmentos’ grandes, mais fáceis de serem utilizados por outros processos;
- *QUICK FIT*
  - Mantém listas separadas para alguns dos tamanhos de segmentos de memória disponíveis em geral mais solicitados;

Gerenciamento de Memória

- Cada algoritmo pode manter listas separadas para processos e para espaços livres
  - Vantagem:
    - Aumenta desempenho;
  - Desvantagens:
  - Aumenta complexidade quando espaço de memória é liberado – gerenciamento das listas;
    - Fragmentação;

Gerenciamento de Memória

- **O que fazer quando não existe espaço suficiente para todos os processos ativos?**
- ***Swapping***
  - Chaveamento de processos inteiros entre a memória principal e o disco
- **Overlays → Memória Virtual**
  - Programas são divididos em pedaços menores
  - Pedaços são chaveados entre a memória principal e o disco

Gerenciamento de Memória

- ***Swapping***:
  - chaveamento de processos inteiros entre a memória principal e o disco;
  - Transferência do processo da memória principal para a memória secundária (normalmente disco): s*wap out*;
  - Transferência do processo da memória secundária para a memória principal: s*wap in*;
  - Pode ser utilizado tanto com partições fixas quanto com partições variáveis;

Gerenciamento de Memória
Memória Virtual

- Programas maiores que a memória eram divididos em pedaços menores chamados *overlays*;
  - Programador define áreas de *overlay*;
    - Vantagem: expansão da memória principal;
    - Desvantagem: custo muito alto;
- **Memória Virtual**
  - Sistema operacional é responsável por dividir o programa em *overlays*;
  - Sistema operacional realiza o chaveamento desses pedaços entre a memória e o disco;

Gerenciamento de Memória
Memória Virtual

- Sistema operacional é responsável por dividir o programa em *overlays*;
- Sistema operacional realiza o chaveamento desses pedaços entre a memória principal e o disco;
- Década de 60: ATLAS → primeiro sistema com MV (Universidade Manchester – Reino Unido);
- 1972: sistema comercial: IBM System/370;

Gerenciamento de Memória
Memória Virtual

- Com MV existe a sensação de se ter mais memória principal do que realmente a que se tem presente;
- O hardware muitas vezes implementa funções da gerência de memória virtual:
  - SO deve considerar características da arquitetura;

Gerenciamento de Memória
Memória Virtual

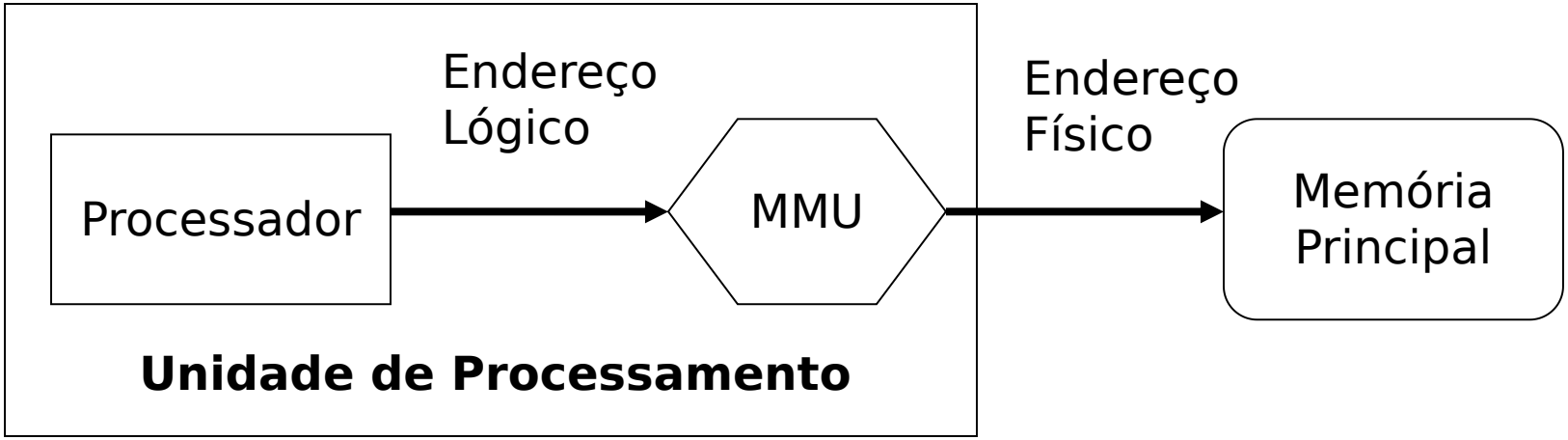
- O **Espaço de Endereçamento Virtual** de um processo é formado por todos os endereços virtuais que esse processo pode gerar;
- O **Espaço de Endereçamento Físico** de um processo é formado por todos os endereços físicos/reais aceitos pela memória principal;

Gerenciamento de Memória
Memória Virtual

- Um processo em Memória Virtual faz referência a endereços virtuais e não a endereço reais de memória principal;
- No momento da execução de uma instrução, o endereço virtual é traduzido para um endereço real, pois a CPU manipula apenas endereços reais da memória principal → MAPEAMENTO;

Gerenciamento de Memória
Mapeamento

- MMU (*Memory Management Unit*): Realiza mapeamento dos endereços lógicos (usados pelos processos) para endereços físicos;



Gerenciamento de Memória
Memória Virtual

- Técnicas de MV:
  - Paginação:
    - Blocos de tamanho fixo chamados de **páginas**;
    - SO mantém uma lista de todas as páginas;
    - Endereços Virtuais formam o espaço de endereçamento virtual;
    - O espaço de endereçamento virtual é dividido em páginas virtuais;
    - Mapeamento entre endereços reais e virtuais realizado pela MMU;
  - Segmentação:
    - Blocos de tamanho arbitrário chamados **segmentos**;

Gerenciamento de Memória
Memória Virtual – Paginação

- Memória Principal e Memória Secundária são organizadas em páginas de mesmo tamanho;
- Página é a unidade básica para transferência de informação;
- Tabela de páginas: responsável por armazenar informações sobre as páginas virtuais:
  - argumento de entrada: número da página virtual;
  - argumento de saída (resultado): número da página real (ou moldura de página – *page frame*);

Gerenciamento de Memória
Memória Virtual

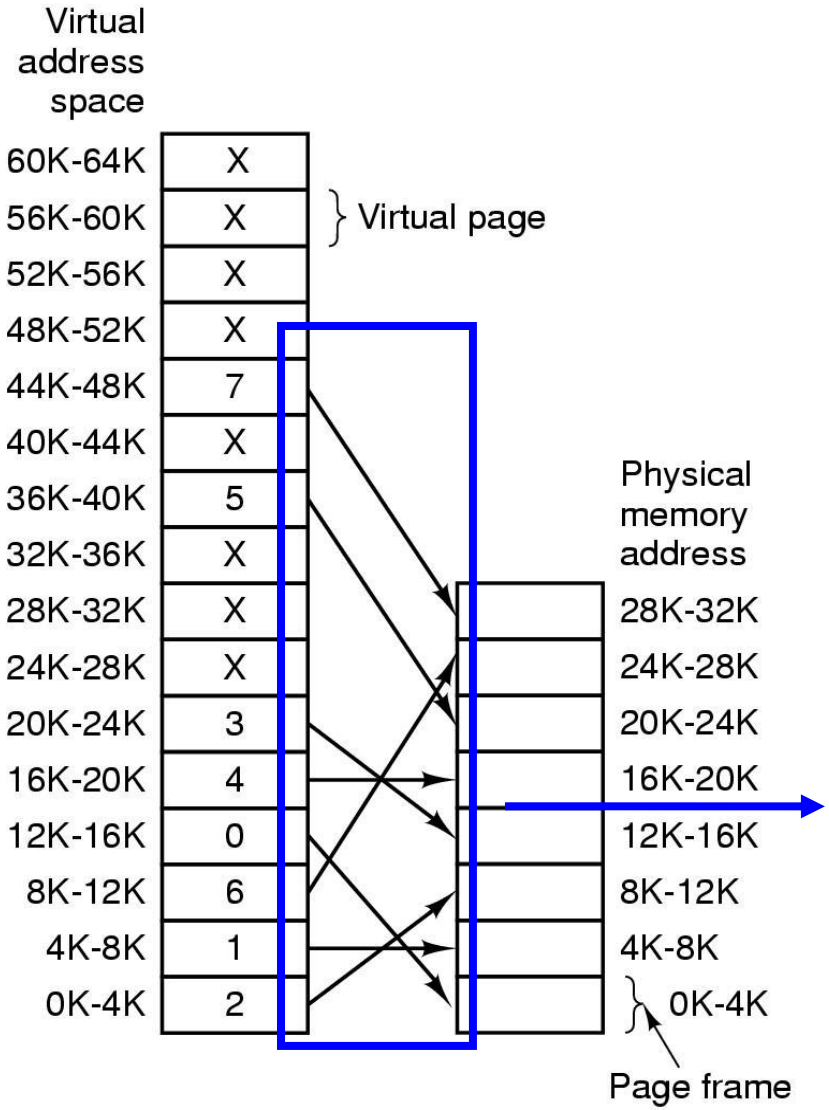
- Exemplo:
  - Páginas de 4Kb
    - 4096 bytes/endereços (0-4095);
  - 64Kb de espaço virtual;
  - 32Kb de espaço real;

  - Temos:
    - 16 páginas virtuais;
    - 8 páginas reais;

Gerenciamento de Memória
Memória Virtual – Paginação

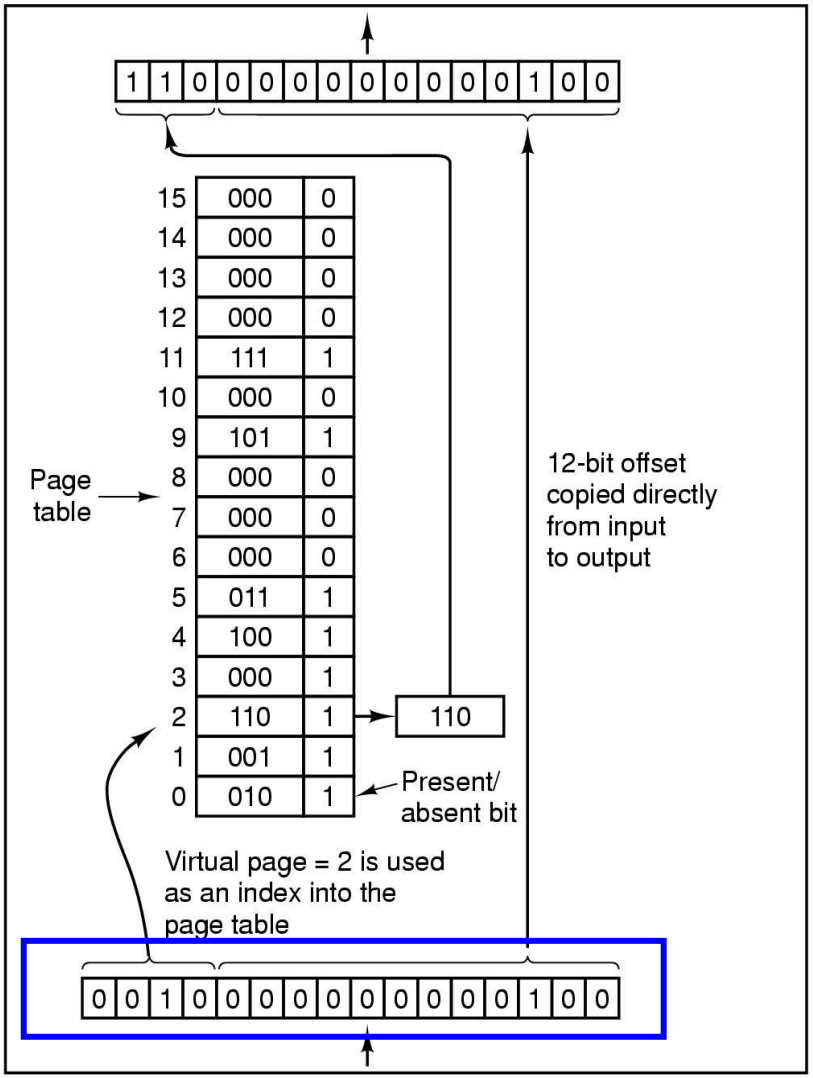
- Problemas:
  - Fragmentação interna;
  - Definição do tamanho das páginas;
    - Geralmente a MMU que define e não o SO;
    - Páginas maiores: leitura mais eficiente, tabela menor, mas maior fragmentação interna;
    - Páginas menores: leitura menos eficiente, tabela maior, mas menor fragmentação interna;
    - Tamanhos possíveis entre 512 bytes a 64 KB;
- Mapa de bits ou uma lista encadeada com as páginas livres;

Gerenciamento de Memória
Endereço Virtual → Endereço Real



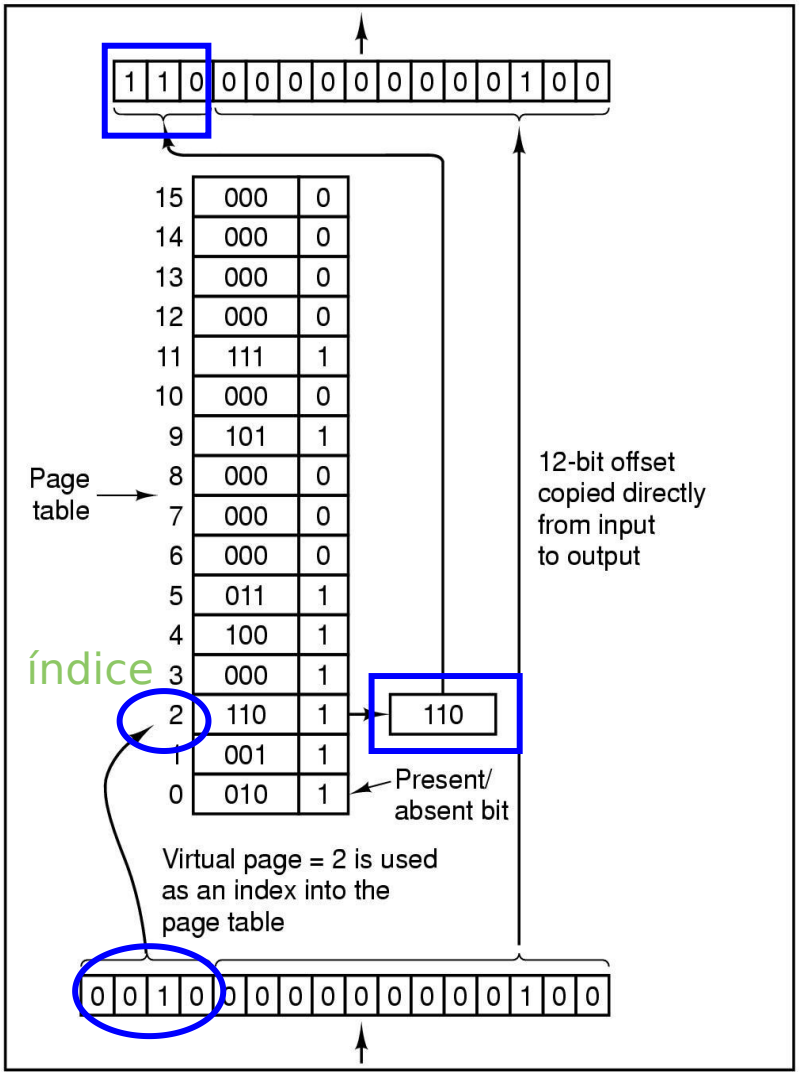
- Página virtual **mapeada** para página real;
- MMU realiza o mapeamento

Gerenciamento de Memória
Mapeamento da MMU



- Operação interna de uma MMU com 16 páginas de 4Kb;
- Endereço virtual de **16 bits**: 4 bits para nº de páginas e 12 para deslocamento;
- Com 4 bits é possível ter 16 páginas virtuais (2^4);
- 12 bits para deslocamento é possível endereçar os 4096 bytes;

Gerenciamento de Memória
Mapeamento da MMU



- Número da página virtual é usado como índice;
- Se página está na memória principal, então o nº da página real (110) é copiado para os três bits mais significativos do endereço de saída (real), juntamente com o deslocamento sem alteração;
- Endereço real com 15 bits é enviado à memória;

Gerenciamento de Memória
Memória Virtual – Paginação

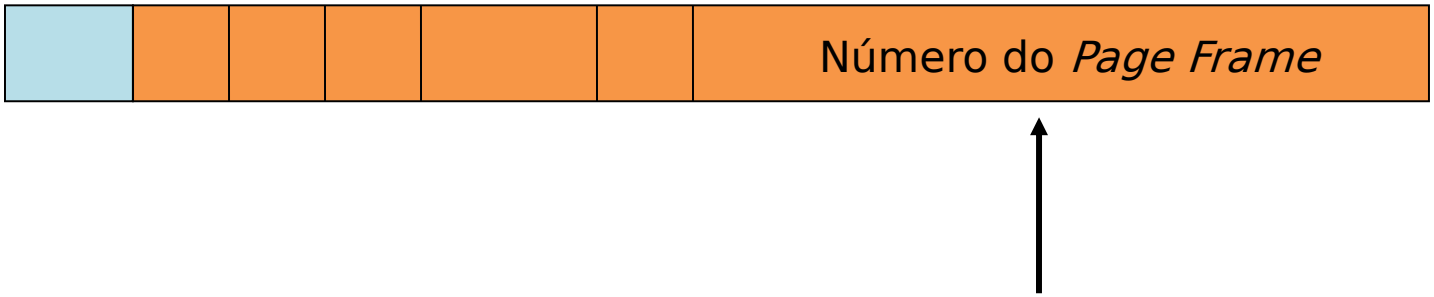
- A Tabela de páginas pode ser armazenada de três diferentes maneiras:
  - Em um conjunto de registradores, se a memória for pequena;
    - Vantagem: rápido
    - Desvantagem: precisa carregar toda a tabela nos registradores a cada chaveamento de contexto
  - Na própria MP→ MMU gerencia utilizando dois registradores:
    - Registrador Base da tabela de páginas (PTBR – *page table base register*): indica o endereço físico de memória onde a tabela está alocada;
    - Registrador Limite da tabela de páginas (PTLR – *page table limit register*): indica o número de entradas da tabela (número de páginas);
    - Dois acessos à memória;

Gerenciamento de Memória
Memória Virtual – Paginação

- Em uma memória *cache* na MMU chamada **Memória Associativa**;
  - Também conhecida como TLB (*Translation Lookaside Buffer* – *buffer* de tradução dinâmica);
  - Hardware especial para mapear endereços virtuais para endereços reais sem ter que passar pela tabela de páginas na memória principal;
  - Melhora o desempenho;

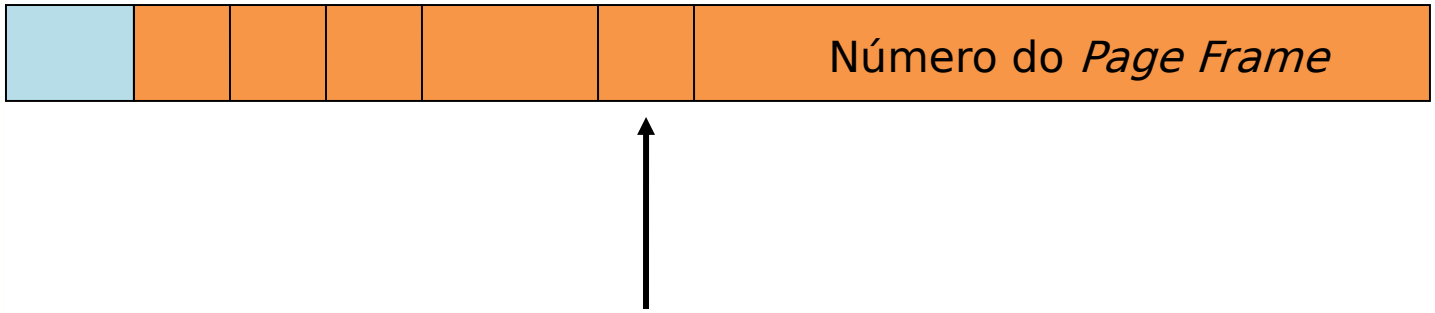
Gerenciamento de Memória
Memória Virtual – Paginação

- Estrutura de uma tabela de páginas (normalmente 32 bits)



Gerenciamento de Memória
Memória Virtual – Paginação

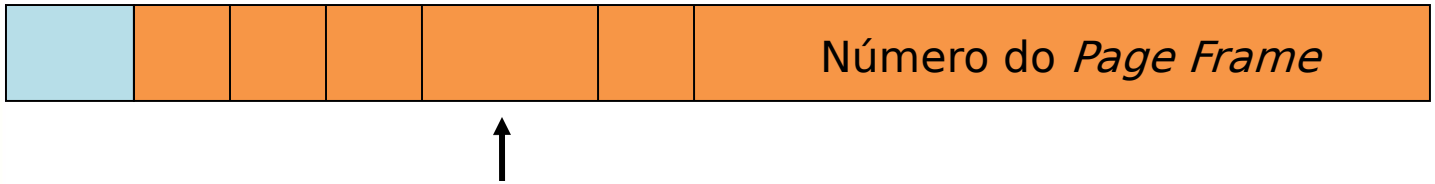
- Estrutura de uma tabela de páginas (normalmente 32 bits)



Bit de Residência:
Se valor igual 1, então entrada válida para uso;
Se valor igual 0, então entrada inválida, pois página virtual correspondente não está na memória;

Gerenciamento de Memória
Memória Virtual – Paginação

- Estrutura de uma tabela de páginas (normalmente 32 bits)



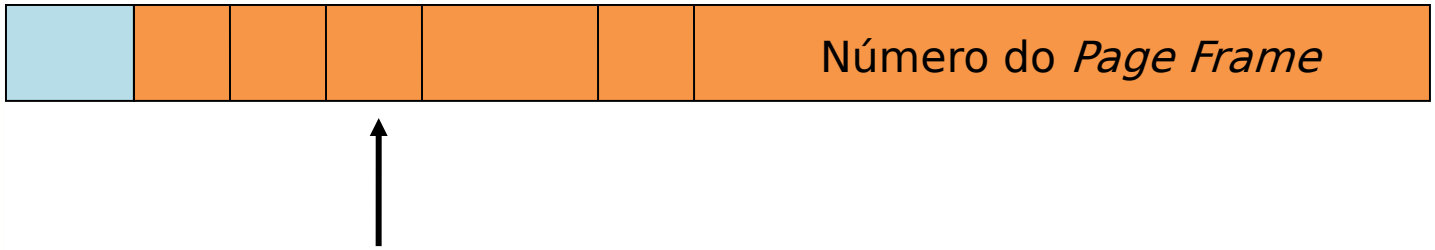
Bits de Proteção:
Indicam tipos de acessos permitidos:

1 bit → 0 – leitura/escrita
1 – leitura

3 bits → 0 – Leitura
1 – Escrita
2 - Execução

Gerenciamento de Memória
Memória Virtual – Paginação

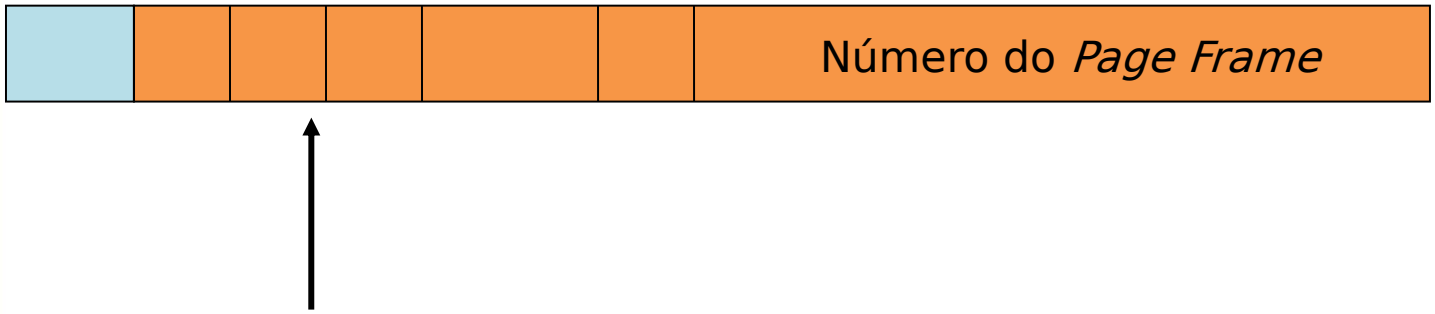
- Estrutura de uma tabela de páginas (normalmente 32 bits)



Bit de Modificação (Bit M):
Controla o uso da página;
Se valor igual a 1, página foi escrita;
página é copiada para o disco
Se valor igual a 0, página não foi modificada;
página não é copiada para o disco;

Gerenciamento de Memória
Memória Virtual – Paginação

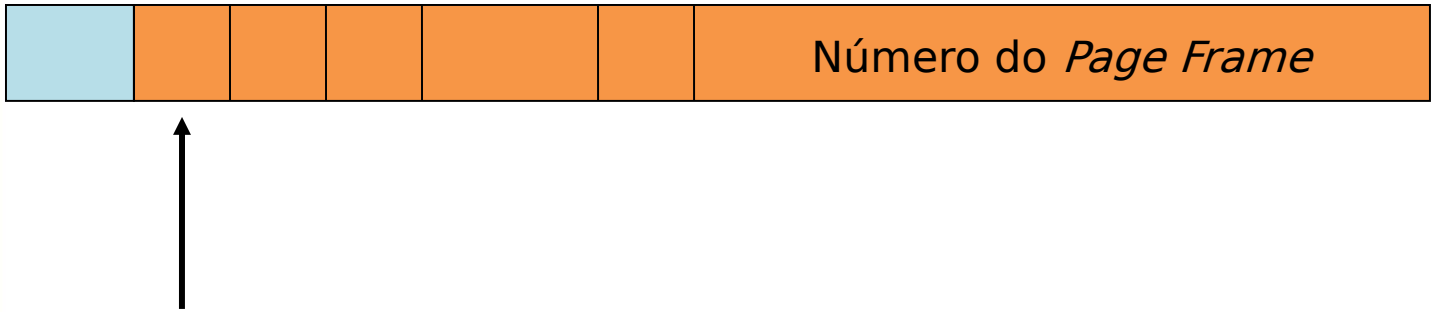
- Estrutura de uma tabela de páginas (normalmente 32 bits)



Bit de Referência (Bit R):
Controla o uso da página;
Auxilia o SO na escolha da página que deve deixar a MP;
Se valor igual a 1, página foi referenciada (leitura/escrita);
Se valor igual a 0, página não referenciada;

Gerenciamento de Memória
Memória Virtual – Paginação

- Estrutura de uma tabela de páginas (normalmente 32 bits)



Bit de *Cache*:
Necessário quando os dispositivos de entrada/saída são mapeados na memória e não em um endereçamento específico de E/S;

Gerenciamento de Memória
Alocação de Páginas

- Quantas páginas reais serão alocadas a um processo?
- Duas estratégias:
  - Alocação fixa ou estática: cada processo tem um número máximo de páginas reais, definido quando o processo é criado;
    - O limite pode ser igual para todos os processos;
    - Vantagem: simplicidade;
    - Desvantagens: (i) número muito pequeno de páginas reais pode causar muita paginação (troca de páginas da memória principal); (ii) número muito grande de páginas reais causa desperdício de memória principal;

Gerenciamento de Memória
Alocação de Páginas

- Alocação variável ou dinâmica: número máximo de páginas reais alocadas ao processo varia durante sua execução;
  - Vantagem: (i) processos com elevada taxa de paginação podem ter seu limite de páginas reais ampliado; (ii) processos com baixa taxa de paginação podem ter seu limite de páginas reais reduzido;
  - Desvantagem: monitoramento constante;

Gerenciamento de Memória
Busca de Página

- Política de busca de página: determina quando uma página deve ser carregada para a memória

- Três estratégias:
  - **Paginação simples**:
    - Todas as páginas virtuais do processo são carregadas para a memória principal;
    - Assim, sempre todas as páginas são válidas;
  - **Paginação por demanda** (*Demand Paging*):
    - Apenas as páginas referenciadas são carregadas na memória principal;
    - Quais páginas virtuais foram carregadas → Bit de controle (bit de residência);
    - Página inválida;
  - **Paginação antecipada** (*Antecipatory Paging*)
    - Carrega para a memória principal, além da página referenciada, outras páginas que podem ou não ser necessárias para o processo

Gerenciamento de Memória
Busca de Página

- **Página inválida**: MMU gera uma interrupção de proteção e aciona o sistema operacional;
  - Se a página está fora do espaço de endereçamento do processo, o processo é abortado;
  - Se a página ainda não foi carregada na memória principal, ocorre uma **falta de página** *(page fault)*;

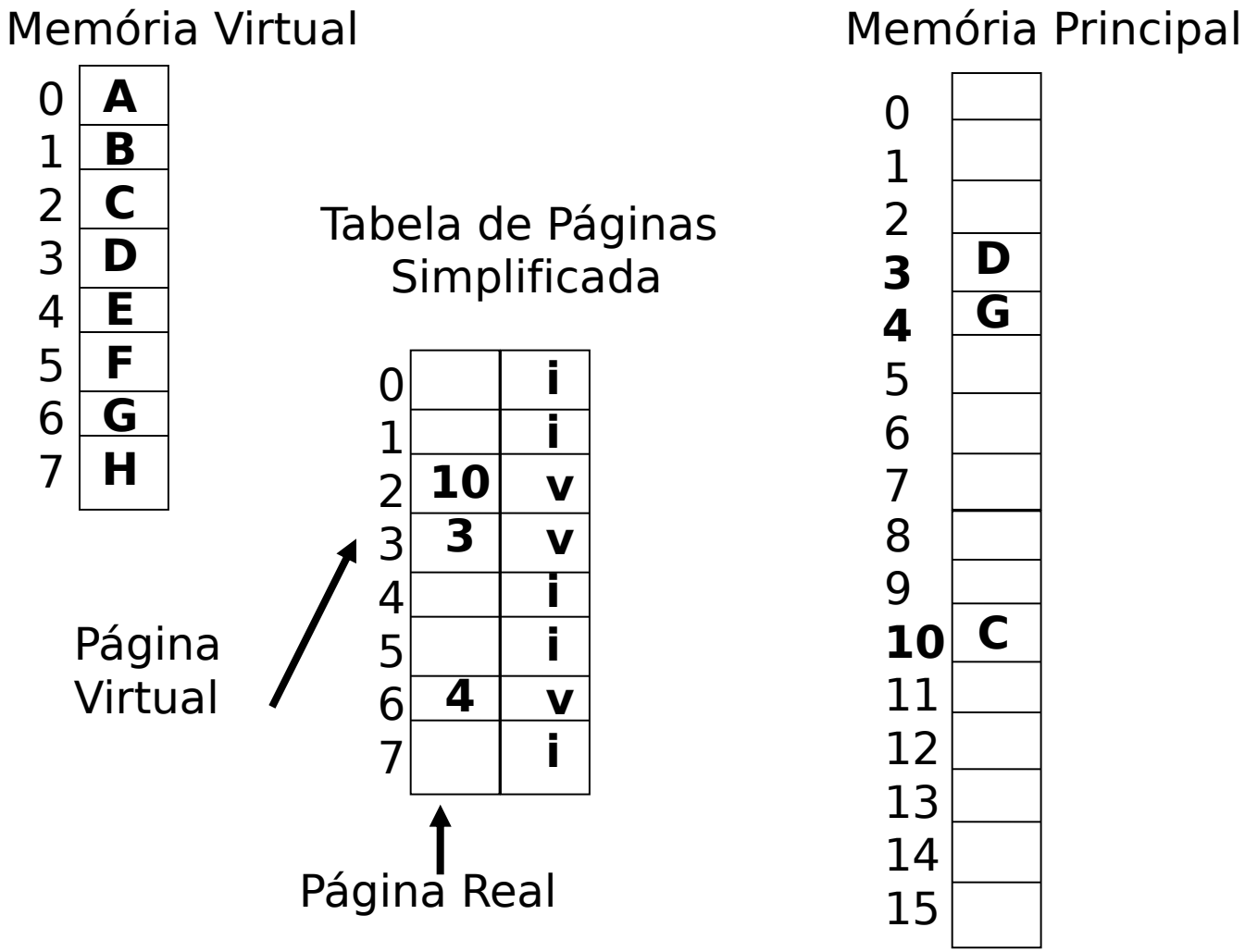
Gerenciamento de Memória
Busca de Página

- **Falta de Página**:
  - Processo é suspenso e seu descritor é inserido em uma **fila especial** – fila dos processos esperando uma página virtual;
  - Uma página real livre deve ser alocada;
  - A página virtual acessada deve ser localizada no disco;
  - Operação de leitura de disco, indicando o endereço da página virtual no disco e o endereço da página real alocada;

Gerenciamento de Memória
Busca de Página

- Após a leitura do disco:
  - Tabela de páginas do processo é corrigida para indicar que a página virtual agora está válida e está na página real alocada;
    - *Pager*: carrega páginas especificas de um processo do disco para a memória principal;
  - O descritor do processo é retirado da **fila especial** e colocado na fila do processador;

Gerenciamento de Memória
Troca de Páginas



Gerenciamento de Memória
Troca de Páginas

- Se todas as páginas estiverem ocupadas, uma página deve ser retirada: página vítima;
- Exemplo:
  - Dois processos $P1$ e $P2$, cada um com 4 páginas virtuais;
  - Memória principal com 6 páginas;

Gerenciamento de Memória
Troca de Páginas

Memória Virtual P1

0	A
1	B
2	C
3	D

Tabela de Páginas P1 Simplificada

0	1	v
1	5	v
2		i
3	0	v

Memória Principal

0	D
1	A
2	F
3	E
4	G
5	B

3 páginas de cada processo

Memória Virtual P2

0	E
1	F
2	G
3	H

Tabela de Páginas P2 Simplificada

0	3	v
1	2	v
2	4	v
3		i

- P2 tenta acessar página 3! Falta de Página!

Gerenciamento de Memória
Troca de Páginas

Memória Virtual P1

0	A
1	B
2	C
3	D

Tabela de Páginas P1 Simplificada

0	1	v
1	5	v
2		i
3	0	v

Memória Principal

0	D
1	A
2	F
3	E
4	H
5	B

Memória Virtual P2

0	E
1	F
2	G
3	H

Tabela de Páginas P2 Simplificada

0	3	v
1	2	v
2		i
3	4	v

3 páginas de cada processo

- Página 2 (virtual) é escolhida como vítima!

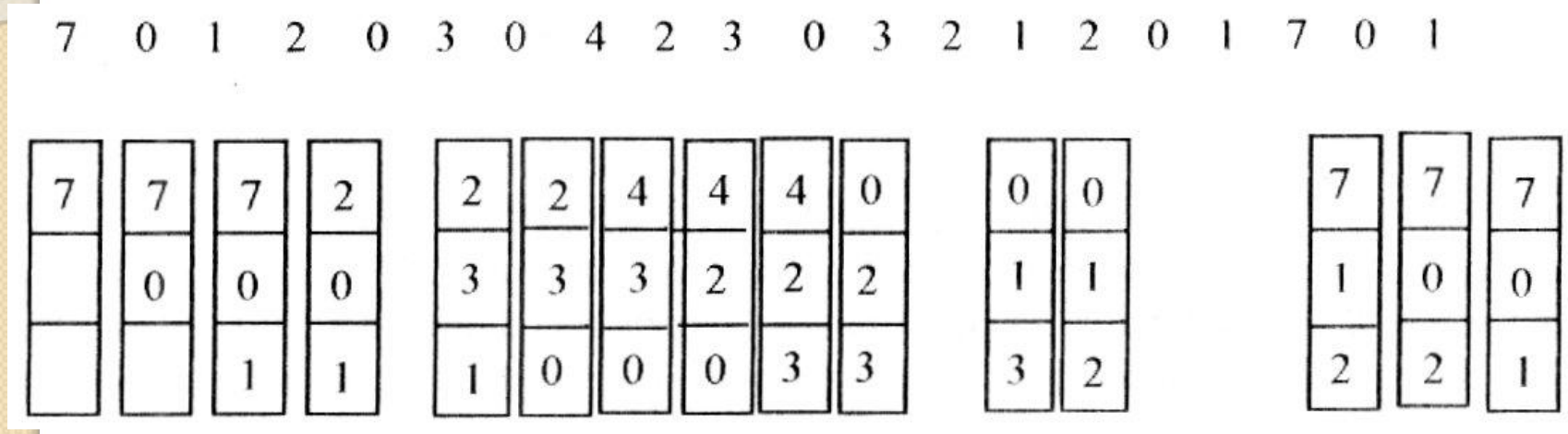
Algoritmos de Troca de Páginas Baseados Somente no Histórico

- Não dispõem de recursos que permitam o uso de informações sobre operações anteriores sobre cada uma das páginas, apenas um histórico linear de acessos
- São métodos mais simples, empregados em sistemas mais básicos; um processo de decisão simples geralmente é realizado rapidamente

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *First-in First-out Page Replacement* (FIFO)
  - SO mantém uma listas das páginas correntes na memória;
    - A página no início da lista é a mais antiga e a página no final da lista é a mais nova;
  - Simples, mas pode ser ineficiente, pois uma página que está em uso constante pode ser retirada;
  - Pouco utilizado;

Exemplo de FIFO

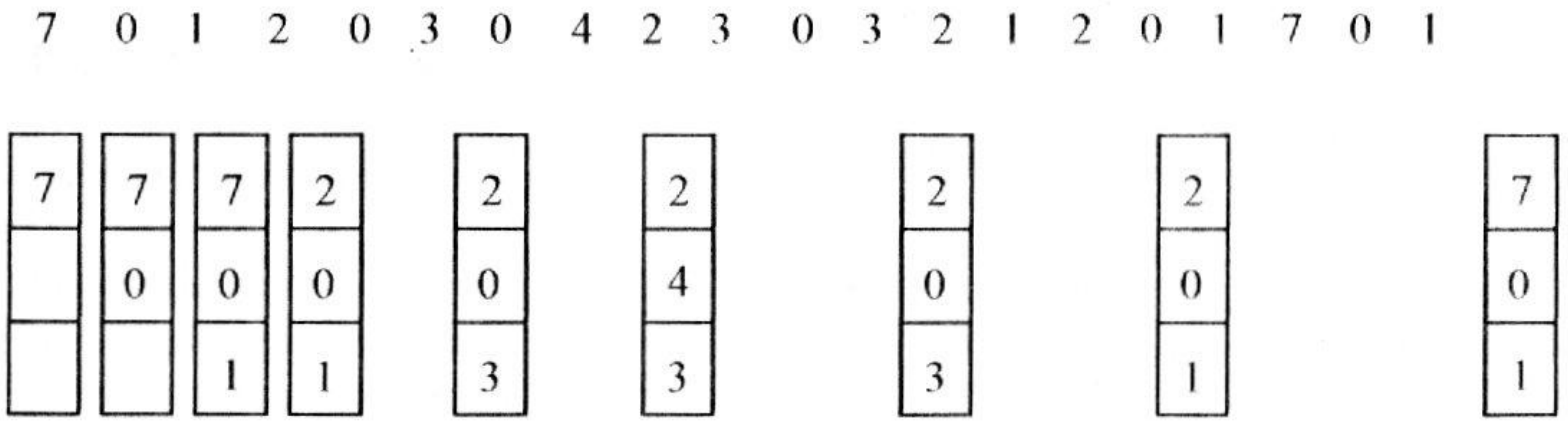


- Com esta sequência de referências temos 15 faltas de página

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo Ótimo (*Optimal Replacement – OPT*):
  - Retira da memória a página que tem menos chance de ser referenciada;
    - Praticamente impossível de se saber;
    - Impraticável;
    - Usado em simulações para comparação com outros algoritmos;

Exemplo do *Optimal Replacement*

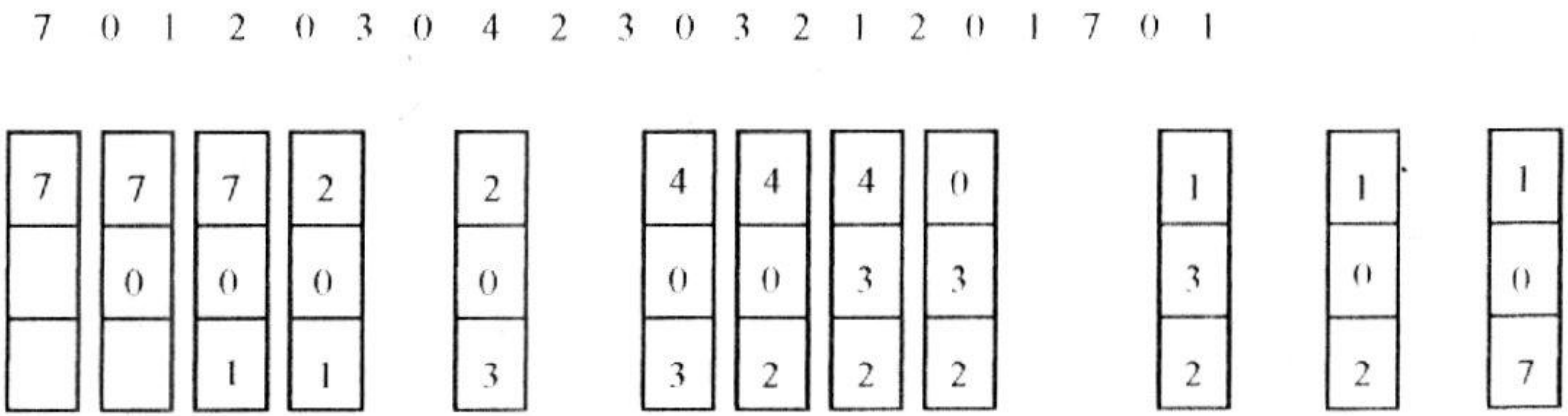


- Para a *string* de busca apresentada, são causadas nove faltas de página

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *Least Recently Used Page Replacement* (LRU)
  - Pode ser implementado tanto por hardware quanto por software:
    - Hardware: MMU deve suportar a implementação LRU;
      - Contador em hardware (64 *bits*) – conta instruções executadas;
      - Após cada referência à memória, o valor do contador é armazenado na entrada da tabela de páginas referente à página acessada;
      - Quando ocorre falta de página, o SO examina todos os contadores e escolhe a página que tem o menor valor

Exemplo do LRU



- Para a mesma string dos exemplos anteriores, aqui são causadas doze faltas de página

Algoritmo LRU

- O algoritmo LRU é bastante utilizado e considerado muito bom. O problema com ele está na sua forma de implementação. O sistema necessita manter uma lista das páginas da memória, ordenada por último uso. Há duas formas de implementação:

  - **Contador**: a cada entrada na tabela de páginas é associado um registrador de tempo de uso. Sempre que uma referência a pagina é feita, o valor do tempo é carregado no registrador. A página substituída deve ser aquela com o menor valor de tempo.
  - **Pilha**: nessa abordagem é mantida uma estrutura de pilha dos números das páginas. Quando a pagina é referenciada, ela é removida da pilha e colocada no topo. Dessa forma, o fundo da pilha sempre contém a página usada menos recentemente.

Algoritmos de Troca de Páginas Empregando Informações Adicionais

- Com o propósito de aumentarem a eficiência do gerenciamento de memória através da minimização do número de trocas de páginas armazenadas na memória principal, alguns algoritmos podem se valer de duas informações adicionais no seu processo de decisão:

  - **Página referenciada**: uma página acessada recentemente é considerada referenciada e, teoricamente, tem uma maior chance de ser utilizada novamente do que uma página que não é referenciada a mais tempo, portanto, tende a ser “poupada” na escolha de uma página para troca. A definição de até quando uma página é considerada referenciada é crítica para o desempenho do algoritmo

Algoritmos de Troca de Páginas Empregando Informações Adicionais

- **Página modificada**: simplesmente identifica se alguma informação contida na página sofreu uma modificação. Em caso positivo, sua escolha para paginação necessitaria de um swap out, com uma operação de escrita na memória não volátil, tornando o processo mais demorado

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *Not Recently Used Page Replacement* (NRU) ou *Não Usada Recentemente* (NUR)
  - Troca as páginas não utilizadas recentemente:
  - 02 bits associados a cada página → R (referência) e M (modificação)
    - Classe 0 (R = 0 e M = 0) → não referenciada, não modificada;
    - Classe 1 (R = 0 e M = 1) → não referenciada, modificada;
    - Classe 2 (R = 1 e M = 0) → referenciada, não modificada;
    - Classe 3 (R = 1 e M = 1) → referenciada, modificada;
  - R e M são atualizados a cada referência à memória;

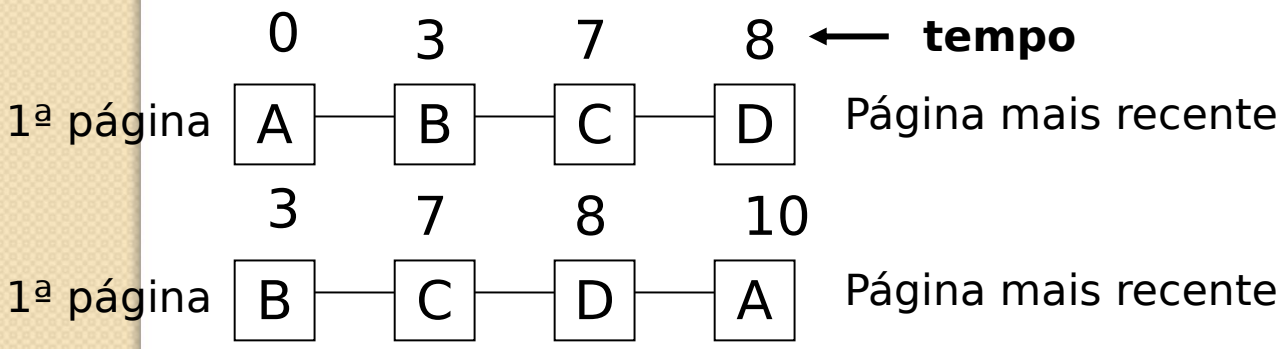
Gerenciamento de Memória
Troca de Páginas – Paginação

- NRU:
  - Periodicamente, o bit R é limpo para diferenciar as páginas que não foram referenciadas recentemente;
    - A cada *tick* do relógio ou interrupção de relógio;
    - Classe 3 → Classe 1;
  - Vantagens: fácil de entender, eficiente para implementar e fornece bom desempenho;

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo da Segunda Chance
  - FIFO + *bit* R;
  - Página mais velha é candidata em potencial;

Se o bit R==0, então página é retirada da memória, senão, R=0 e se dá uma nova chance à página colocando-a no final da lista;



Se página A com R==1; e
falta de página em tempo 10;
Então R=0 e página A vai para final da lista;

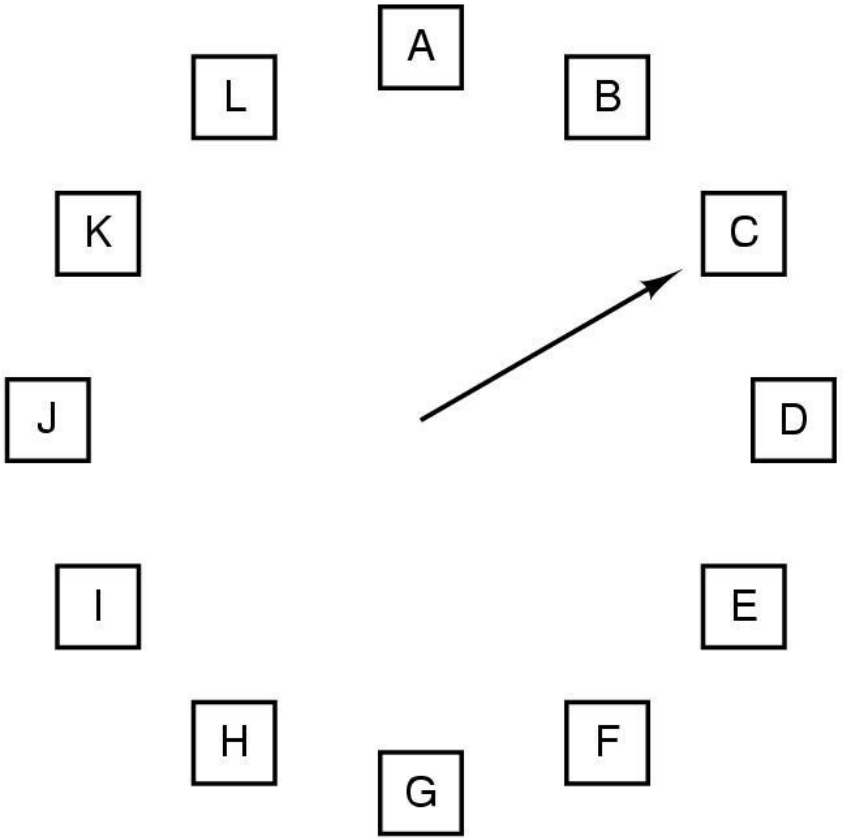
Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo do Relógio
  - Lista circular com ponteiro apontando para a página mais antiga
  - Algoritmo se repete até encontrar R=0;

Se R=0	Se R=1
- troca de página - desloca o ponteiro	- R = 0 - desloca o ponteiro - continua busca

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo do Relógio



When a page fault occurs, the page the hand is pointing to is inspected. The action taken depends on the R bit:

R = 0: Evict the page

R = 1: Clear R and advance hand

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *Least Recently Used Page Replacement* (LRU) ou *Menos Recentemente Usada* (MRU)
  - Troca a página menos referenciada/modificada recentemente;
  - Alto custo
    - Lista encadeada com as páginas que estão na memória, com as mais recentemente utilizadas no início e as menos utilizadas no final;
    - A lista deve ser atualizada a cada referência da memória;

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *Least Recently Used Page Replacement* (LRU)
  - Pode ser implementado tanto por hardware quanto por software:

  - Software: duas maneiras
    - NFU (*Not frequently used*) ou LFU (*least frequently used*);
    - *Aging* (Envelhecimento);

Gerenciamento de Memória
Troca de Páginas – Paginação

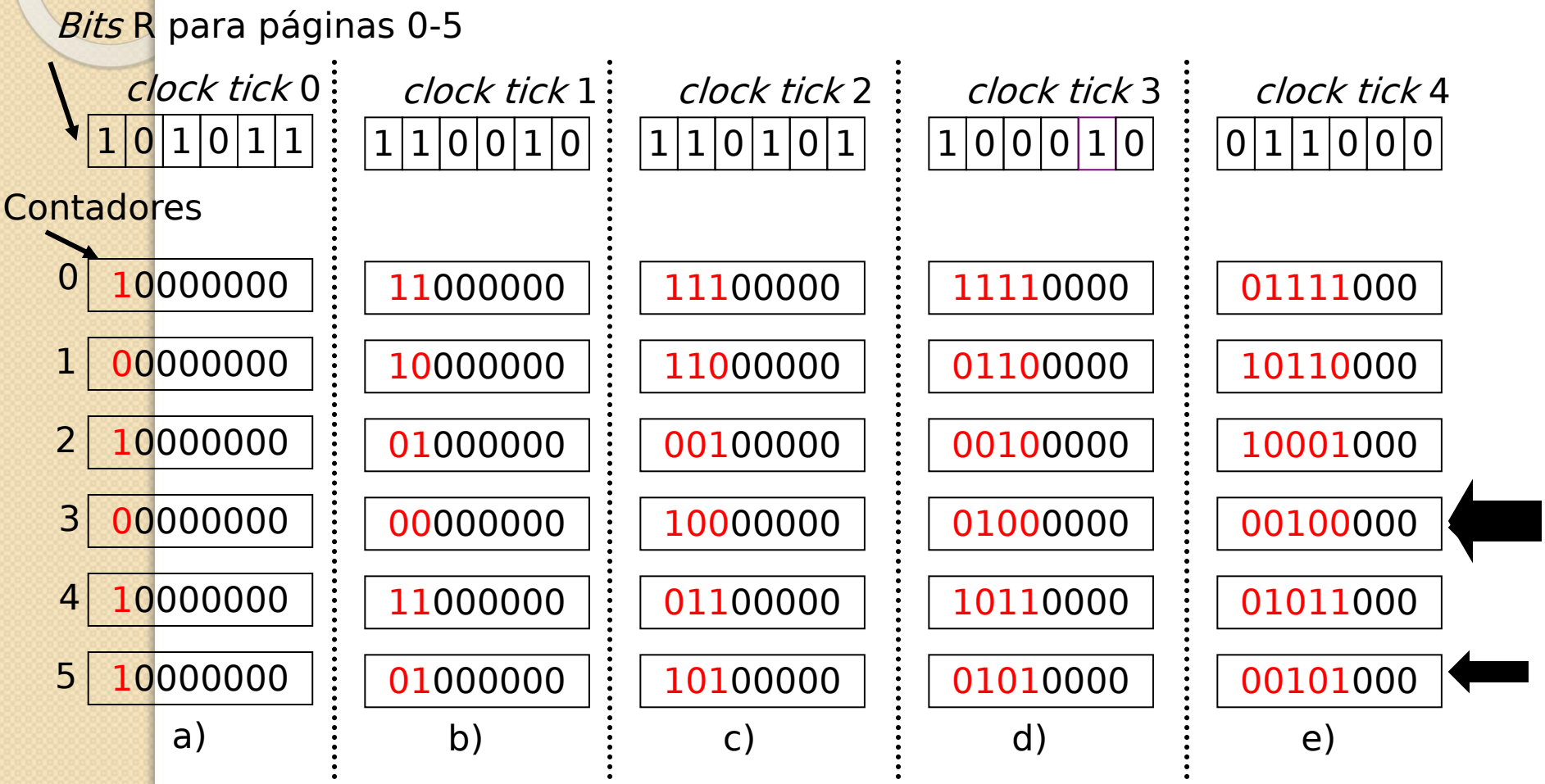
- Software: NFU ou LFU (*least*)
  - Para cada página existe um contador → iniciado com zero e incrementado a cada referência à pagina;
    - Página com menor valor do contador é candidata a troca;
    - Esse algoritmo não se esquece de nada
  - Problema: pode retirar páginas que estão sendo referenciadas com frequência

Gerenciamento de Memória
Troca de Páginas – Paginação

- Software: Algoritmo *aging* (envelhecimento)
  - Modificação do NFU, resolvendo o problema descrito anteriormente;
    - Além de saber **quantas** **vezes** a página foi referenciada, também controla **quando** ela foi referenciada;

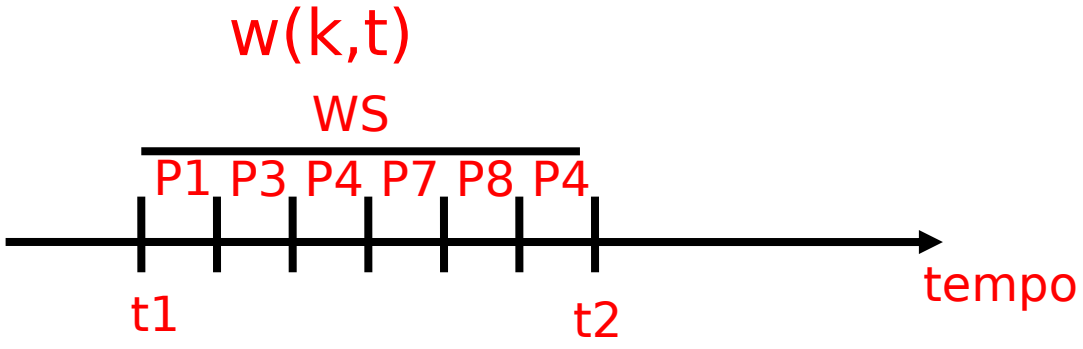
Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *aging*



Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *Working Set (WS)*:
  - Paginação por demanda → páginas são carregadas na memória somente quando são necessárias;
  - Pré-paginação → *Working set*
    - Carregar um conjunto de páginas que um processo está efetivamente utilizando (referenciando) em um determinado tempo ***t*** antes de ele ser posto em execução;



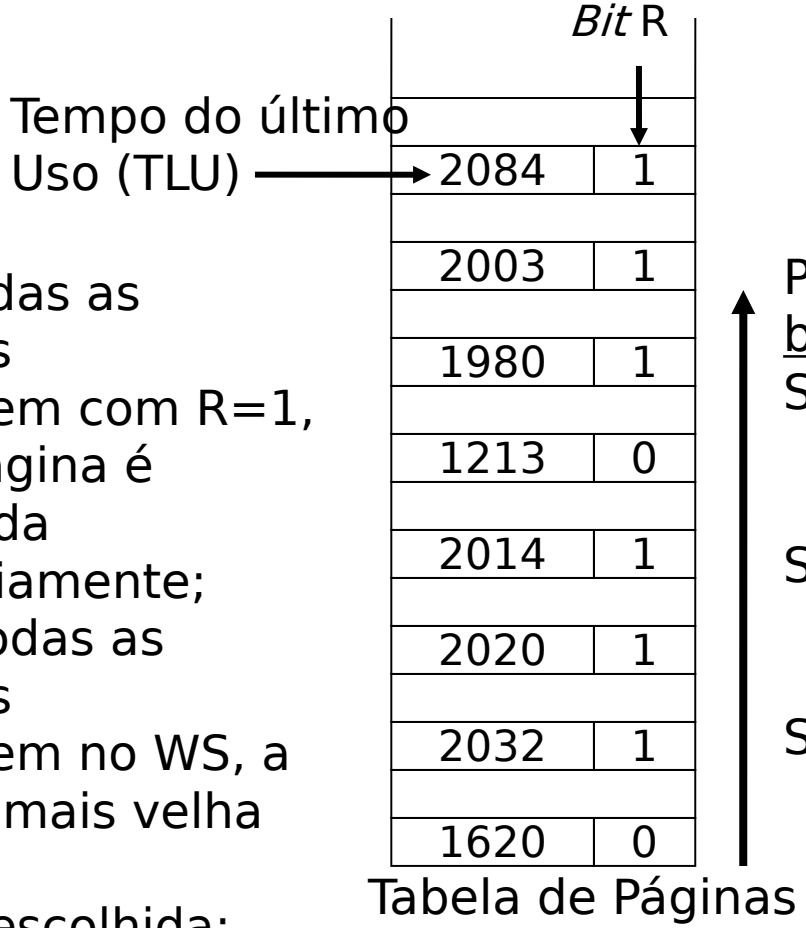
Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *Working Set (WS)*:
  - Objetivo principal: reduzir a falta de páginas
    - Um processo só é executado quando todas as páginas necessárias no tempo ***t*** estão carregadas na memória;
    - SO gerencia quais páginas estão no *Working Set;*
  - Para simplificar: o *working set* pode ser visto como o conjunto de páginas que o processo referenciou durante os últimos **t** segundos de tempo;
  - Utiliza *bit* R e o tempo de relógio (tempo virtual) da última vez que a página foi referenciada;

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *Working Set*:

Tempo virtual atual (CVT): 2204
age = CVT – TLU
(Ex.: 2204-2084 = 120)
τ = múltiplos *clock ticks*



Percorrer as páginas examinando bit R;
Se (R==1)*
  página foi referenciada;
  faz TLU da página igual ao CVT;
Se (R==0 e *age* > τ)
  página não está no *working set*;
  remove a página;
Se (R==0 e *age* <= τ) **
  página está no *working set*;
  guarda página com maior *age*;

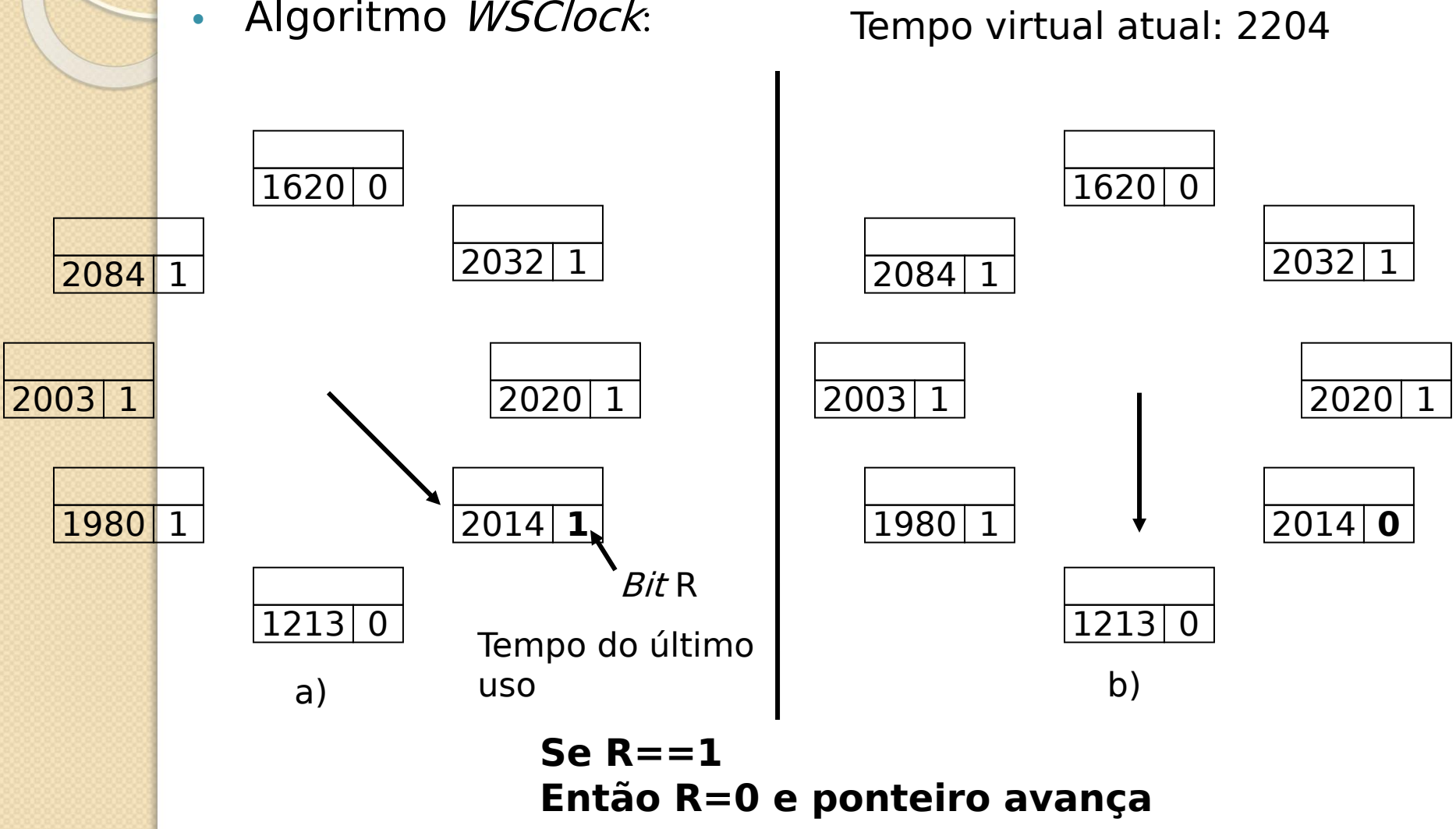
* Se todas as páginas estiverem com R=1, uma página é escolhida aleatoriamente;
** Se todas as páginas estiverem no WS, a página mais velha com R=0 é escolhida;

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *WSClock*:
  - *Clock* + *Working Set*;
  - Lista circular de molduras de páginas formando um anel a cada página carregada na memória;
  - Utiliza *bit R* e o tempo da última vez que a página foi referenciada;
  - *Bit M* utilizado para agendar escrita em disco;

Gerenciamento de Memória
Troca de Páginas – Paginação

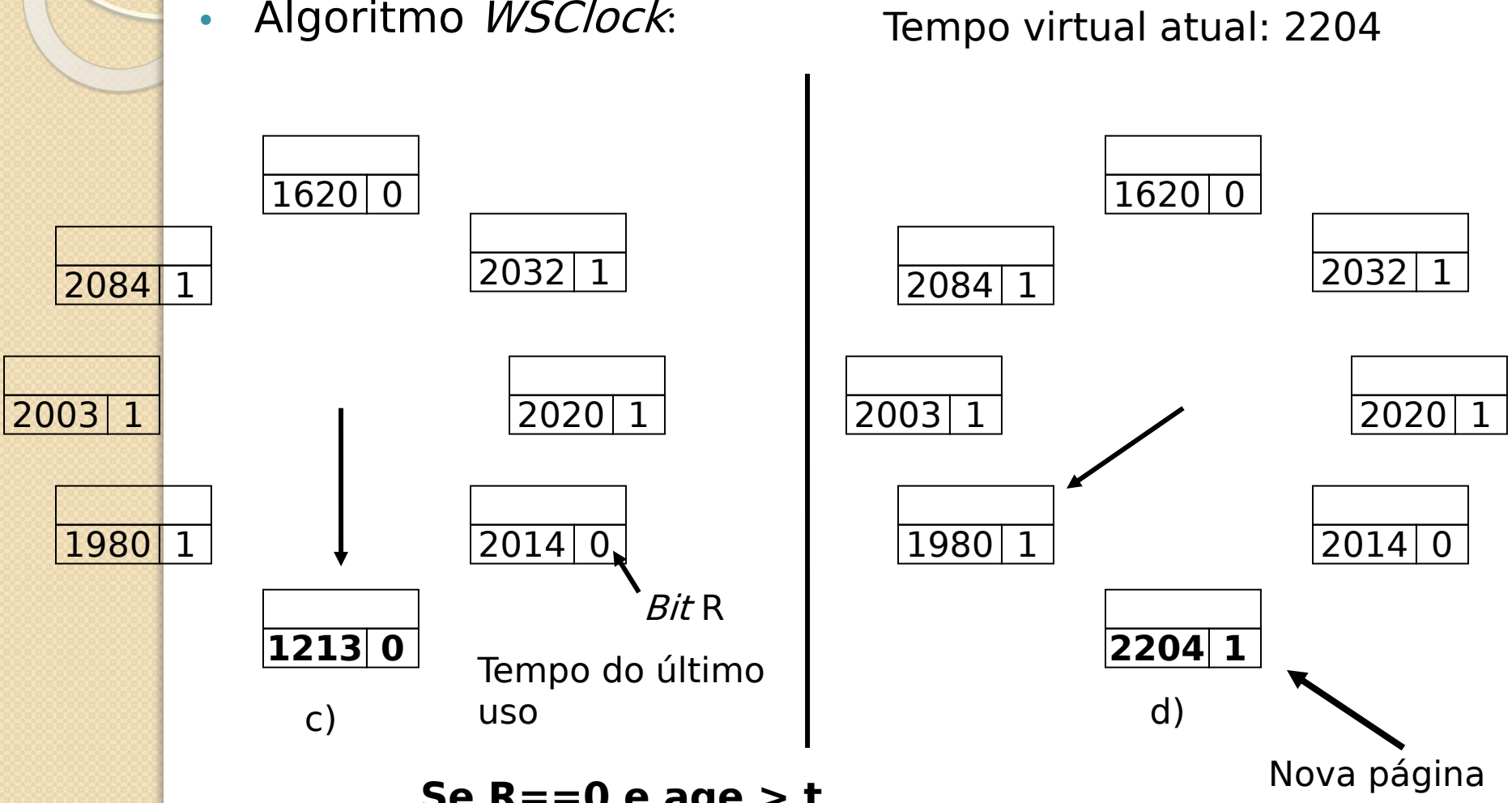
- Algoritmo *WSClock*:



Se R==1
Então R=0 e ponteiro avança

Gerenciamento de Memória
Troca de Páginas – Paginação

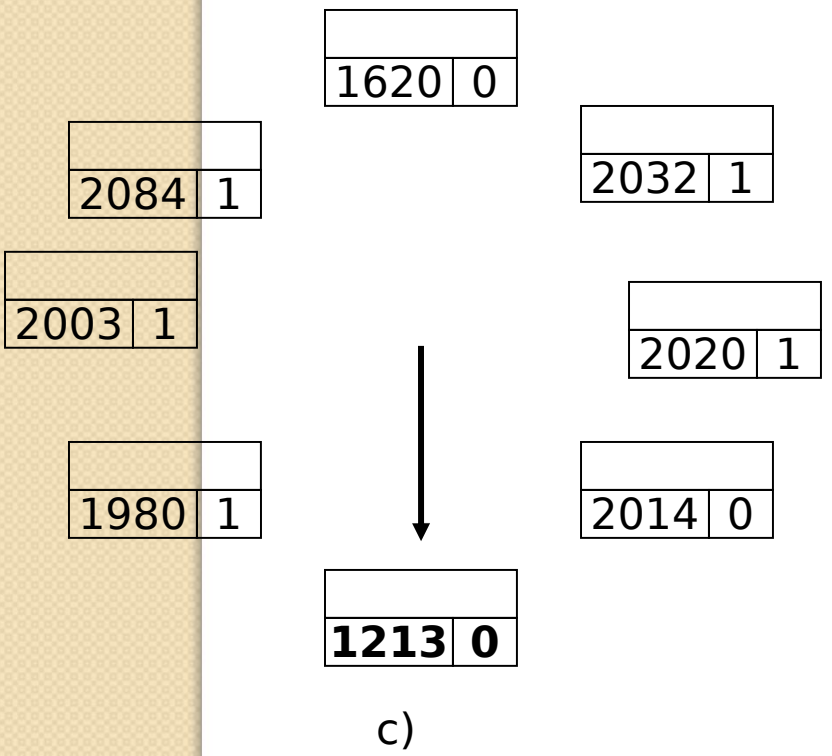
- Algoritmo *WSClock*:



Se R==0 e age > t
e M==0 (não agenda escrita) e troca

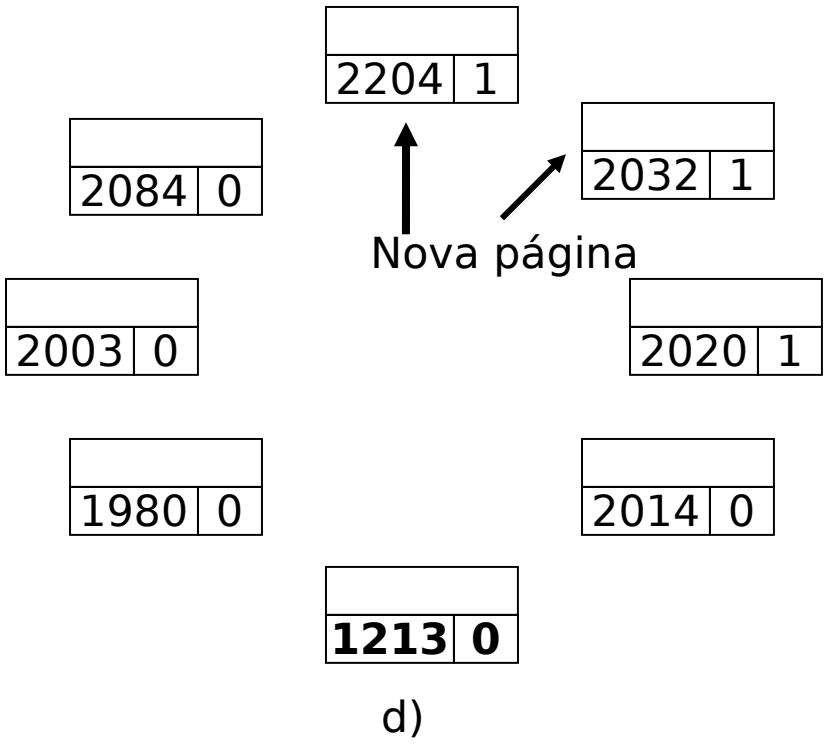
Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmo *WSClock*:



c)

Tempo virtual atual: 2204



d)

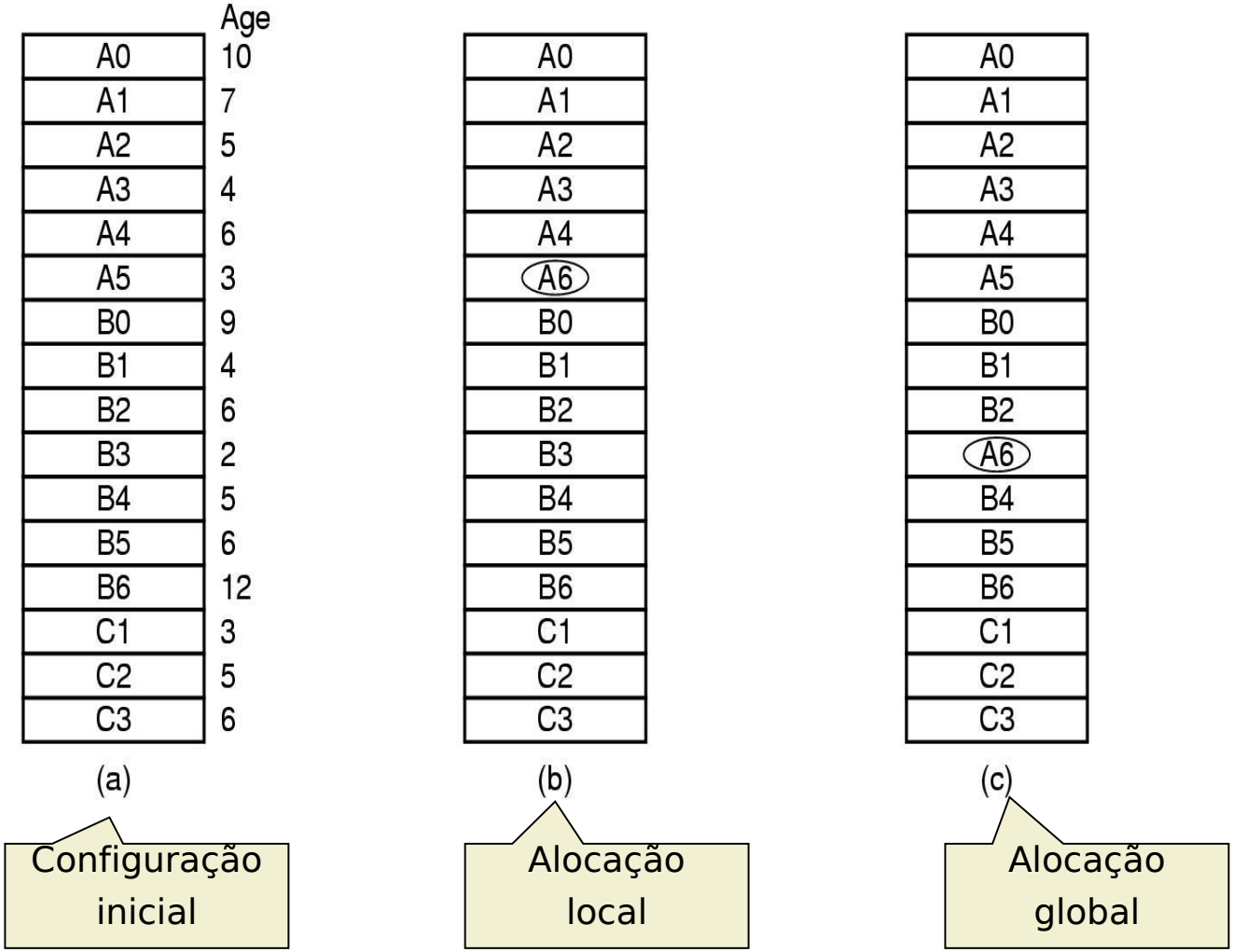
R==0 e age > t
M==1 (agenda escrita e continua procura)

Gerenciamento de Memória
Troca de Páginas

- **Política de Substituição Local**: páginas dos próprios processos são utilizadas na troca;
  - Dificuldade: definir quantas páginas cada processo pode utilizar;
- **Política de Substituição Global**: páginas de todos os processos são utilizadas na troca;
  - Problema: processos com menor prioridade podem ter um número muito reduzido de páginas, e com isso, acontecem muitas faltas de páginas;

Gerenciamento de Memória
Troca de Páginas

Falta de Página no Processo A



Gerenciamento de Memória
Troca de Páginas

- Política de alocação local (número fixo de páginas/processo) permite somente política de substituição local de páginas
- Política de alocação global (número variável de páginas/processo) permite tanto a política de substituição de páginas local quanto global

Gerenciamento de Memória
Troca de Páginas – Paginação

- Algoritmos de substituição local:
  - *Working Set*;
  - *WSClock*;

- Algoritmos de substituição local/global:
  - Ótimo;
  - NRU;
  - FIFO;
  - Segunda Chance;
  - LRU;
  - Relógio;

Gerenciamento de Memória
Implementação da Paginação

- Até agora, vimos somente como uma página é selecionada para remoção. Mas onde a página descartada da memória é colocada?

- **Memória Secundária**
  - A área de troca (*swap area*) é gerenciada como uma lista de espaços disponíveis;
  - O endereço da área de troca de cada processo é mantido na tabela de processos;
    - Cálculo do endereço: MMU;

Gerenciamento de Memória
Implementação da Paginação

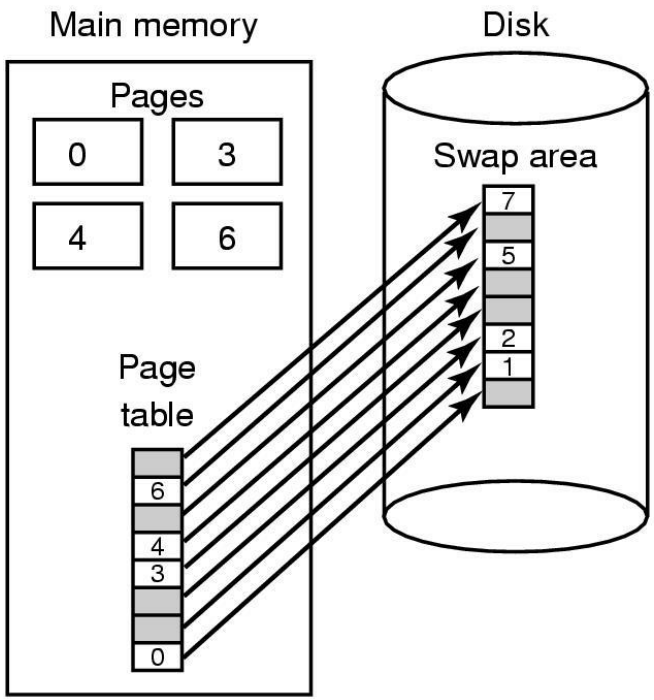
- Memória Secundária
  - Possibilidade A – Assim que o processo é criado, ele é copiado todo para sua área de troca no disco, sendo carregado para memória quando necessário;
    - Área de troca diferente para dados, pilha e programa, pois área de dados pode crescer e a área de pilha crescerá certamente;

Gerenciamento de Memória Implementação da Paginação

- Memória Secundária (cont.)
  - Possibilidade B – Nada é alocado antecipadamente, espaço é alocado em disco quando a página for enviada para lá. Assim, processo na memória RAM não fica “amarrado” a uma área específica;

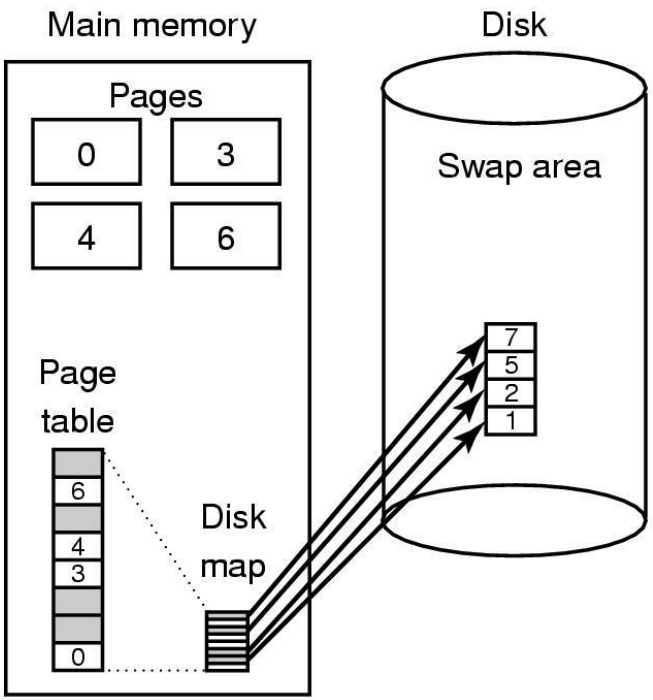
Gerenciamento de Memória
Implementação da Paginação

- Como fica a memória secundária:



(a)

Área de troca estática



(b)

Área de troca dinâmica

Gerenciamento de Memória Virtual
Segmentação

- Segmentação: Visão do programador/compilador
  - Tabelas de segmentos com ***n*** linhas, cada qual apontando para um segmento de memória;
  - Vários espaços de endereçamento;
  - Endereço real → base + deslocamento;
  - Alocação de segmentos segue os algoritmos já estudados:
    - *FIRST-FIT*;
    - *BEST-FIT*;
    - *NEXT-FIT*;
    - *WORST-FIT*;
    - *QUICK-FIT*;

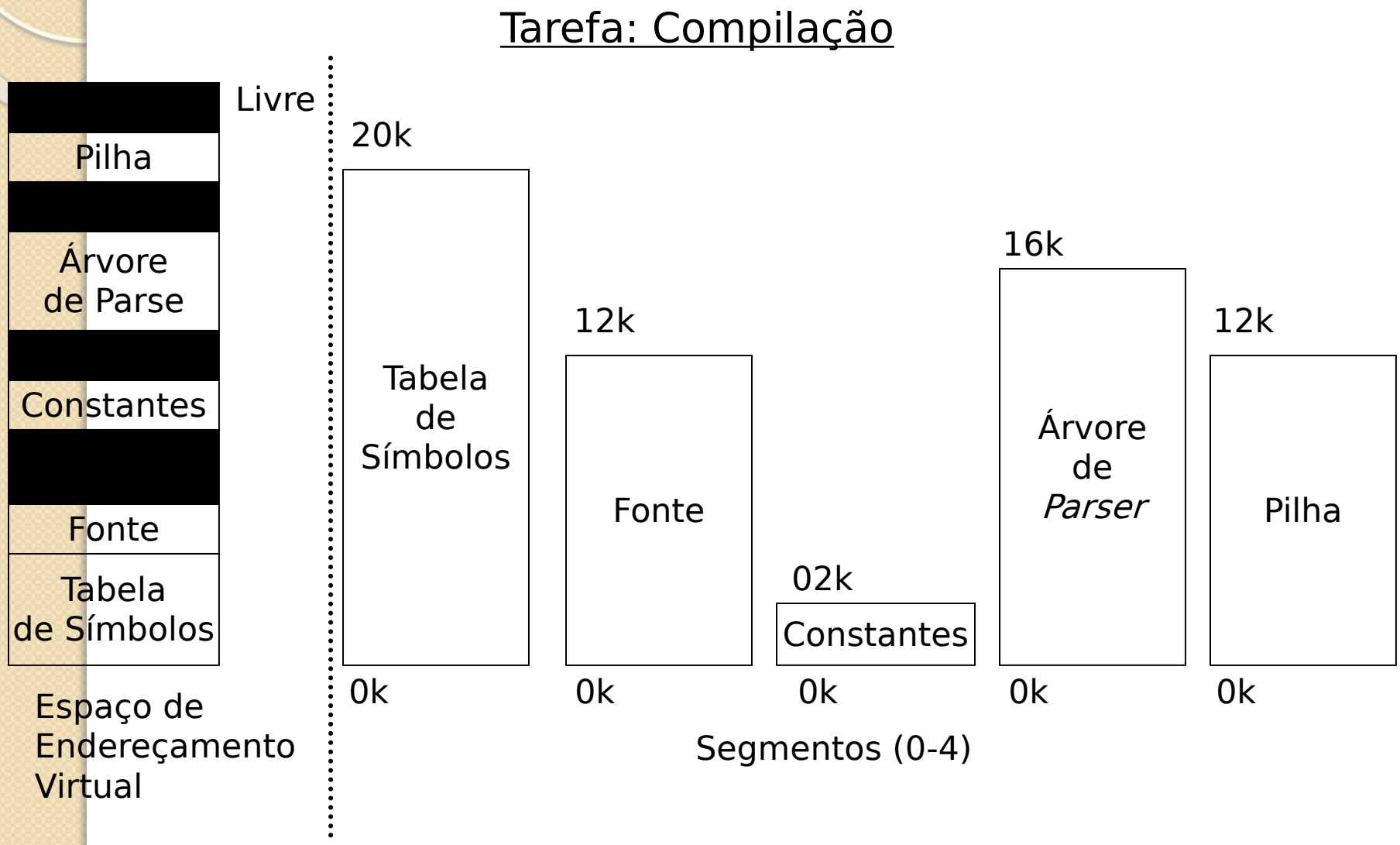
Gerenciamento de Memória Virtual
Segmentação

- Segmentação:
  - Facilita proteção dos dados;
  - Facilita compartilhamento de procedimentos e dados entre processos;
  - MMU também é utilizada para mapeamento entre os endereços lógicos e físicos;
    - Tabela de segmentos informa qual o endereço da memória física do segmento e seu tamanho;

Gerenciamento de Memória Virtual
Segmentação

- Segmentação:
  - Problemas encontrados: embora haja espaço na memória, não há espaço contínuo:
    - Política de relocação: um ou mais segmentos são relocados para abrir espaço contínuo;
    - Política de compactação: todos os espaços são compactados;
    - Política de bloqueio: fila de espera;
    - Política de troca: substituição de segmentos;
  - Sem fragmentação interna, com fragmentação externa;

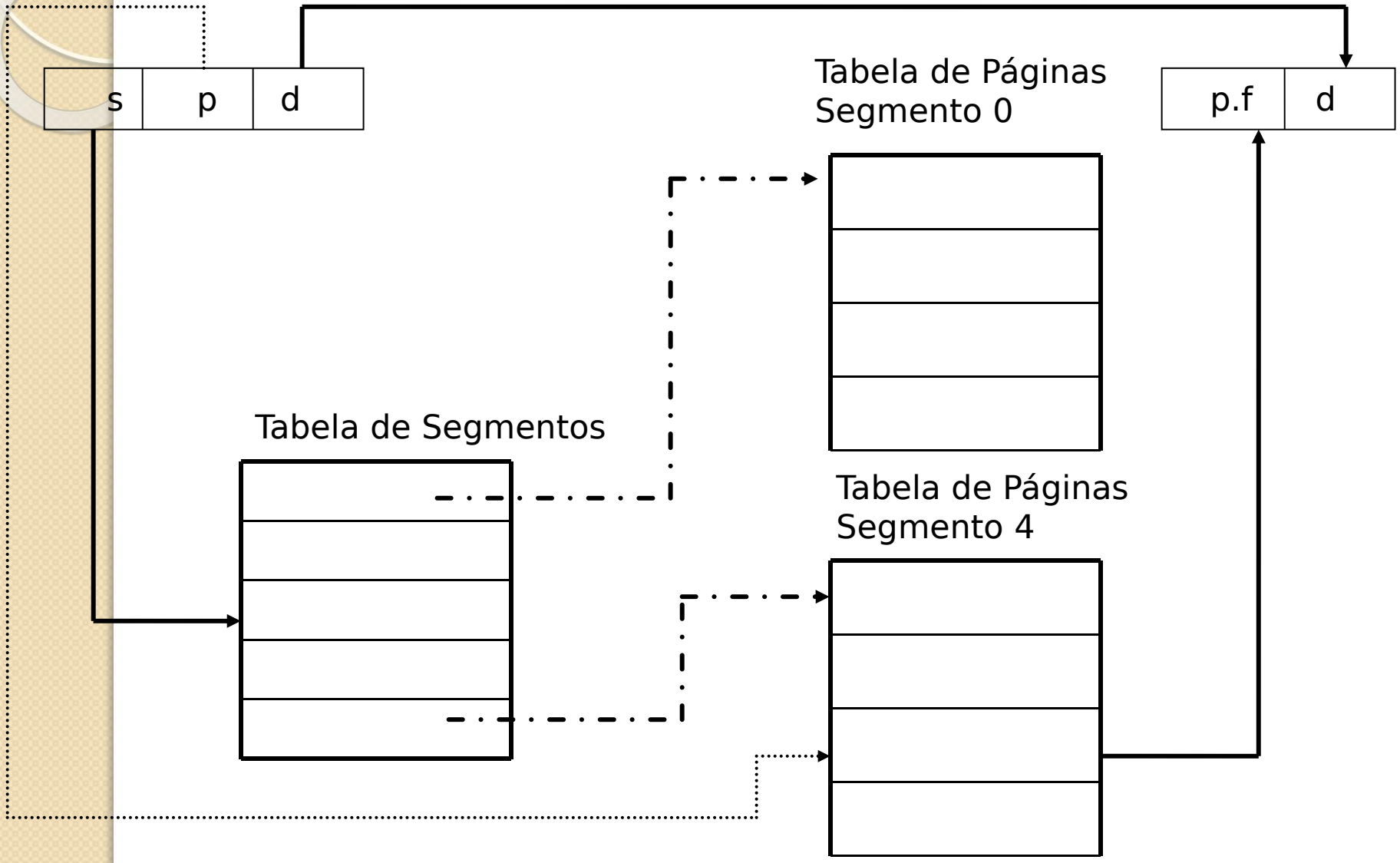
Gerenciamento de Memória Virtual
Segmentação



Gerenciamento de Memória Segmentação–Paginada

- Espaço lógico é formado por segmentos
  - Cada segmento é dividido em páginas lógicas;
  - Cada segmento possui uma tabela de páginas → mapear o endereço de página lógica do segmento em endereço de página física;
  - No endereçamento, a tabela de segmentos indica, para cada segmento, onde sua respectiva tabela de páginas está;
  - Multics, Pentium

Gerenciamento de Memória Segmentação–Paginada



Gerenciamento de Memória Virtual

Consideração	Paginação	Segmentação
Programador deve saber da técnica?	Não	Sim
Espaços de endereçamento existentes	1	Vários
Espaço total de endereço pode exceder memória física?	Sim	Sim
É possível distinguir procedimento de dados e protegê-los?	Não	Sim

Gerenciamento de Memória Virtual

Consideração	Paginação	Segmentação
Tabelas de tamanho variável podem ser acomodadas sem problemas?	Não	Sim
Compartilhamento de procedimentos entre usuário é facilitado?	Não	Sim
Por que?	Para obter espaço de endereçamento maior sem aumentar memória física	Para permitir que programas e dados possam ser divididos em espaços de endereçamento logicamente independentes; compartilhamento e proteção