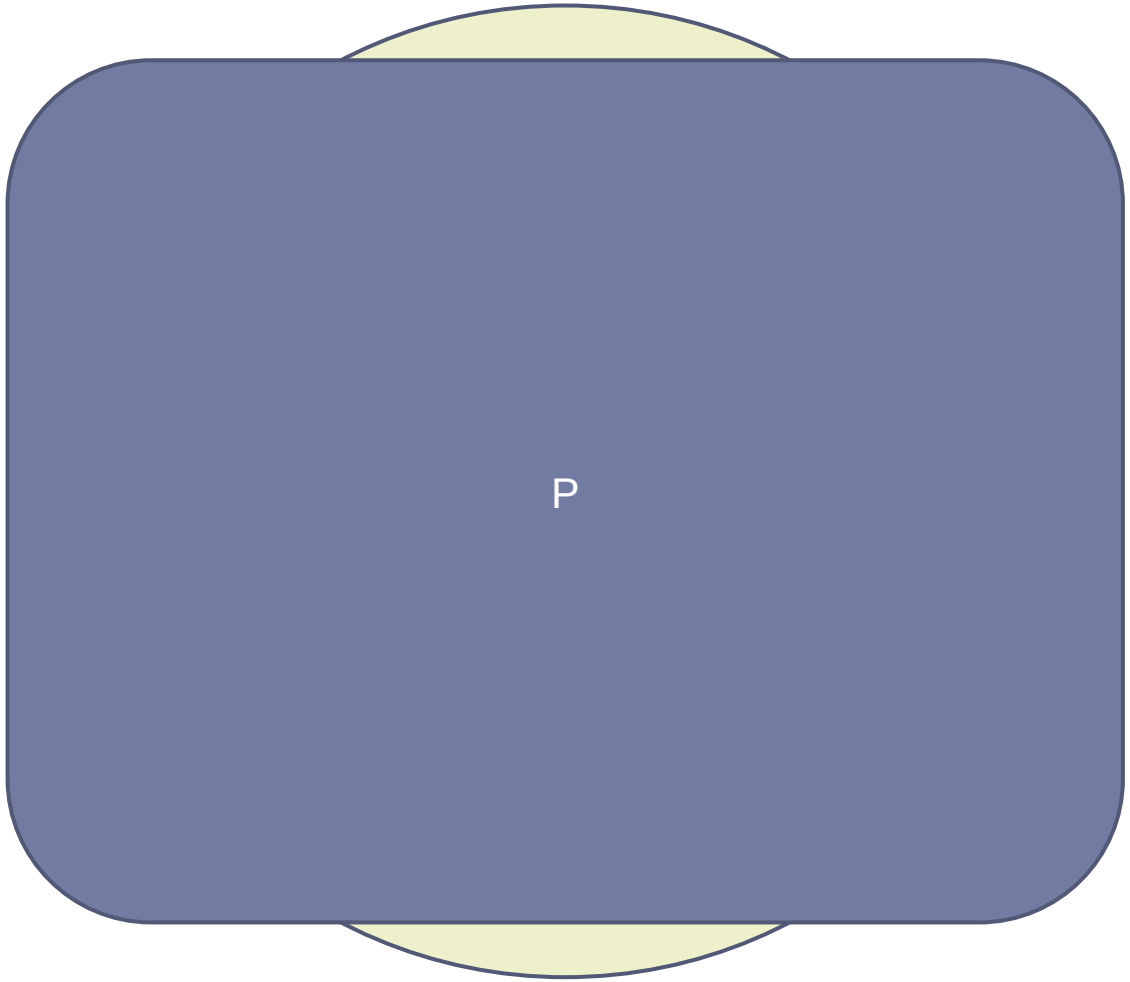
Sistemas Operacionais
Gerenciamento de Memória

Carlos Ferraz (cagf@cin.ufpe.br)
Jorge Cavalcanti Fonsêca (jcbf@cin.ufpe.br)

Memória Física vs. Memória do Programa



Tamanho dos softwares aumenta mais rápido que o tamanho das memórias

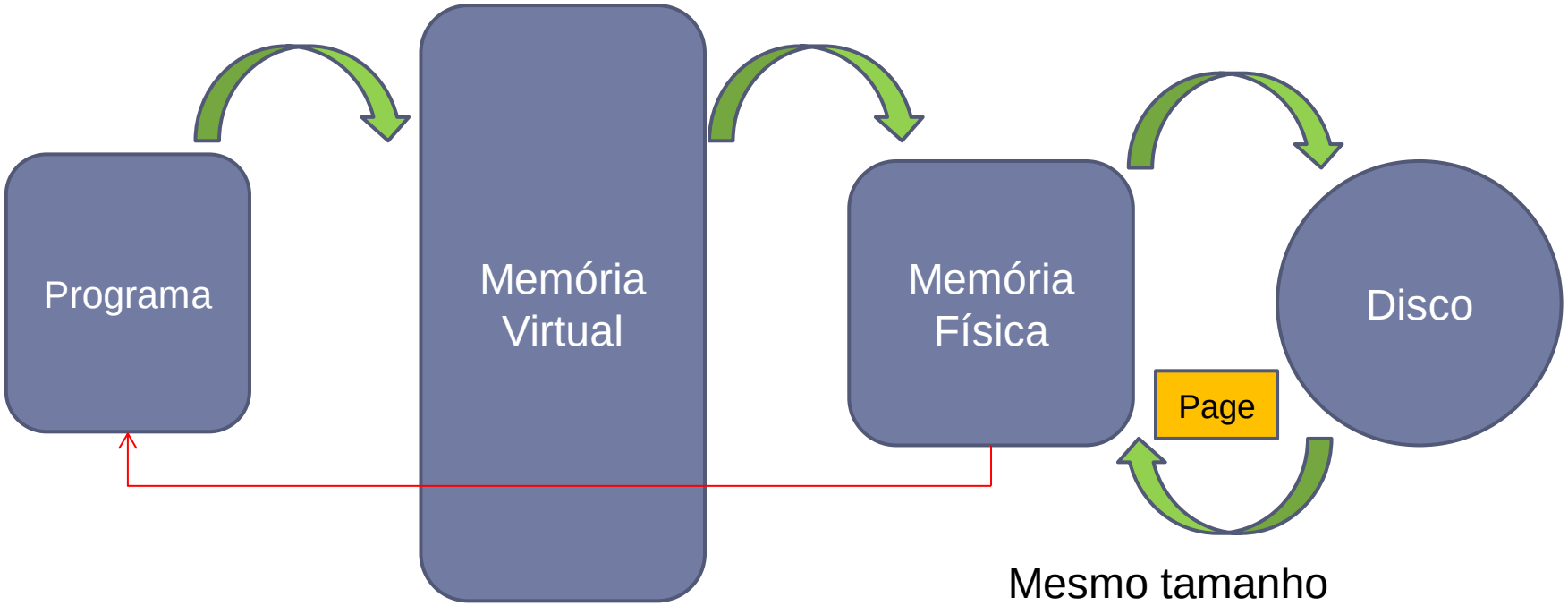
O que fazer?

- 1960 – Divisão manual de programas em módulos chamados sobreposições
  - Gerenciador de Sobreposições
    - S.O.
  - Divisão do programa em sobreposições
    - Programador
    - Alto nível de complexidade → sujeita a erros

- Era preciso deixar essa tarefa com o Sistema Operacional

Memória Virtual

- Programa → Espaço de endereçamento → Páginas



Na prática a memória virtual é uma evolução dos registradores base-limite

→ Paginação ←

Memória Virtual

- Paginação usa a MMU (*Memory Management Unit*)

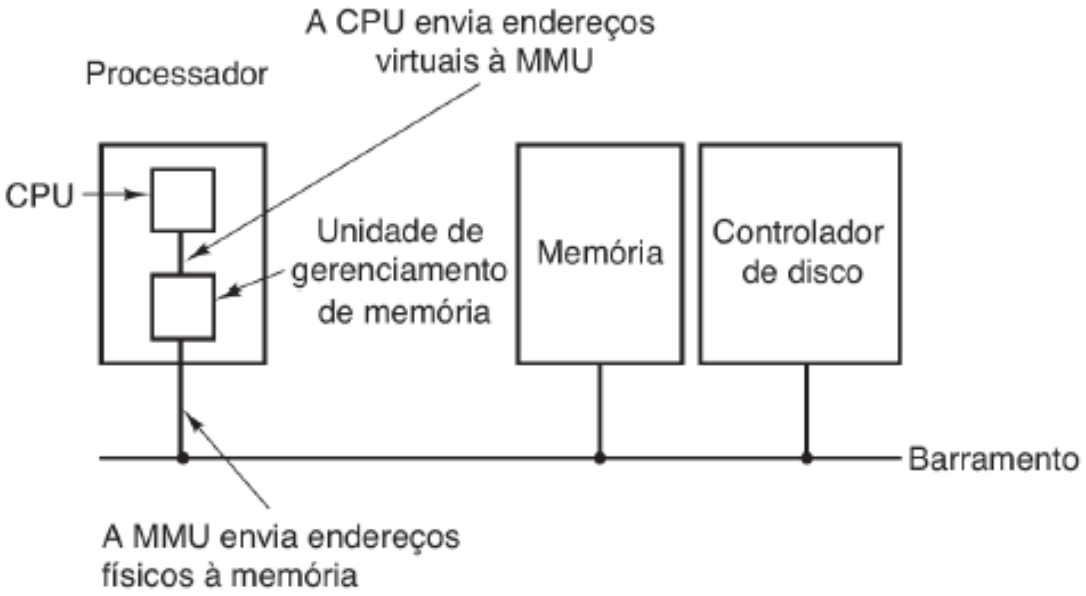
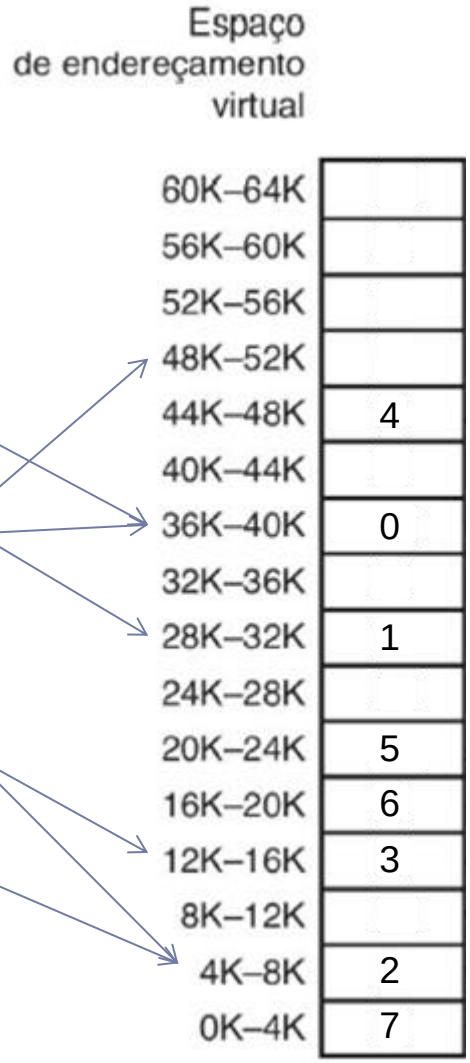


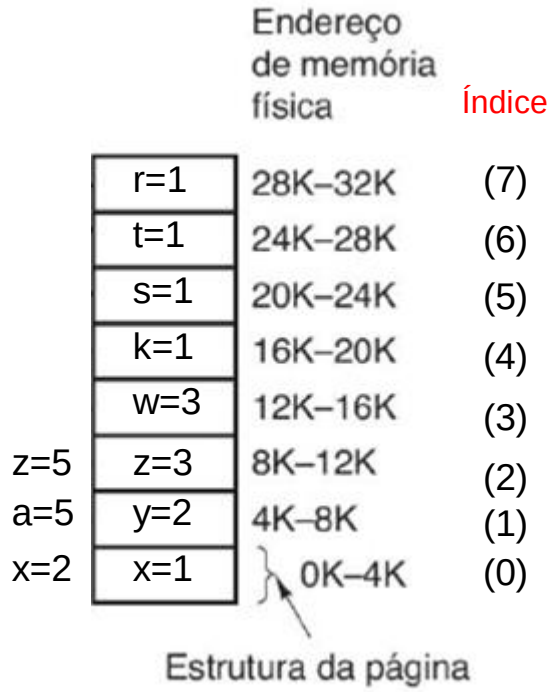
Figura 3.8 A posição e a função da MMU. Aqui a MMU é mostrada como parte do chip da CPU (processador) porque isso é comum atualmente. Contudo, em termos lógicos, poderia ser um chip separado, como ocorria no passado.

Paginação

- int x = 1
- int y = 2;
- int z = x + y;
- int w = z;
- x = x + 1;
- z += x;
- .
- .
- .
- int a = 5;



Falta de Página

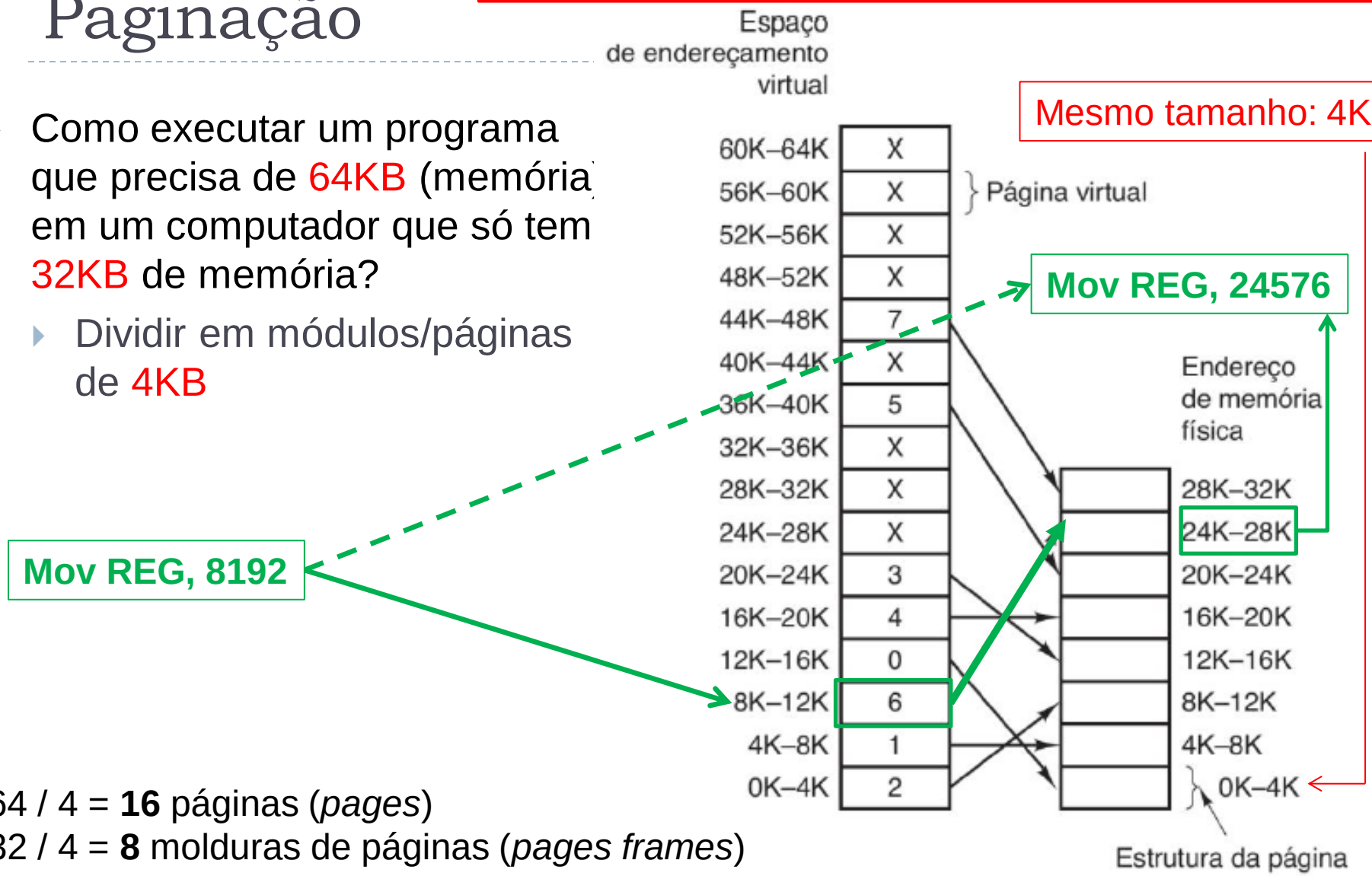


Disco (HD)

Paginação

Memória virtual 64K (2^{16}) > Memória real 32K (2^{15})

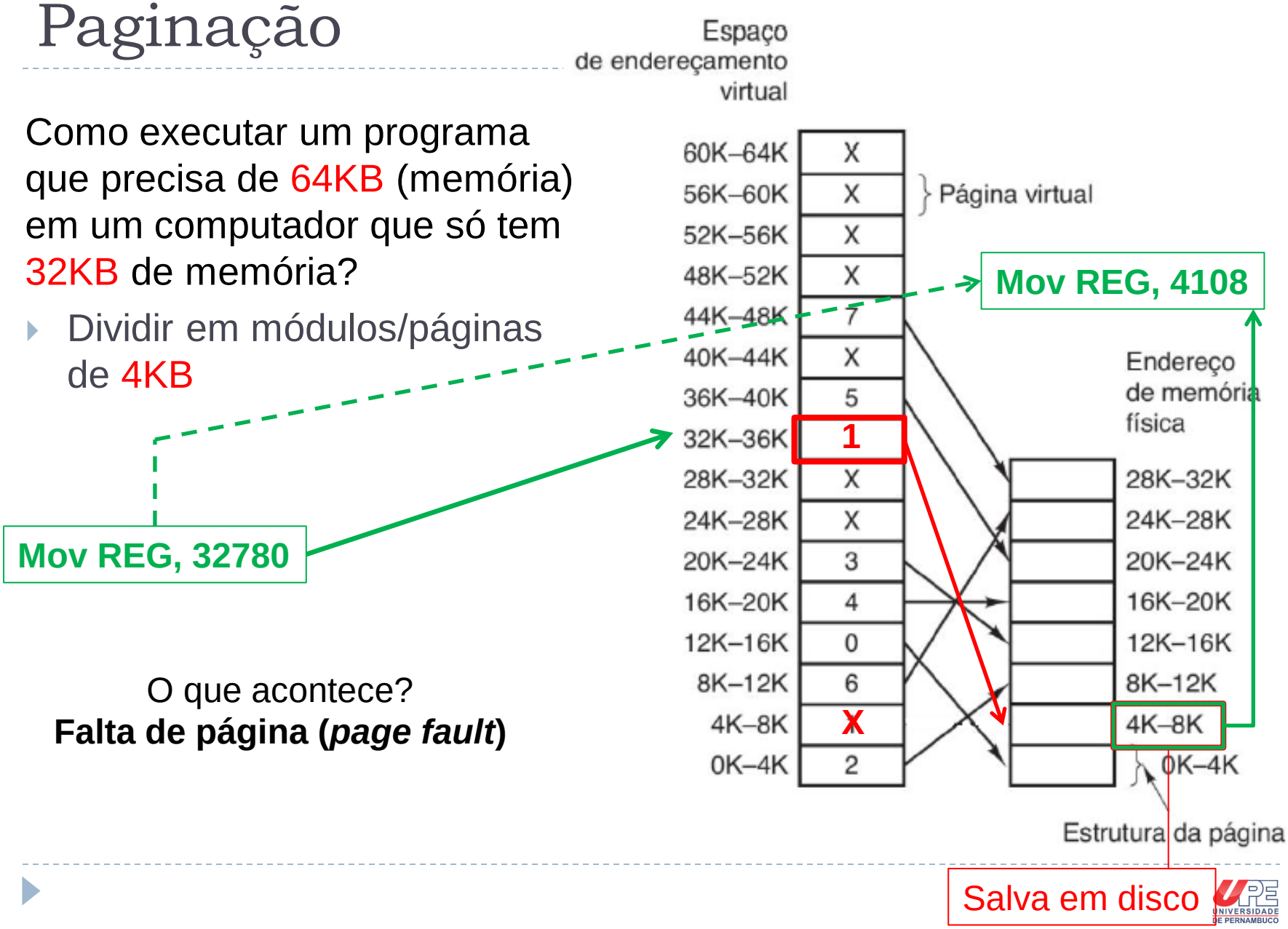
- Como executar um programa que precisa de 64KB (memória) em um computador que só tem 32KB de memória?
  - Dividir em módulos/páginas de 4KB



64 / 4 = **16** páginas (*pages*)
32 / 4 = **8** molduras de páginas (*pages frames*)

Paginação

- Como executar um programa que precisa de 64KB (memória) em um computador que só tem 32KB de memória?
  - Dividir em módulos/páginas de 4KB

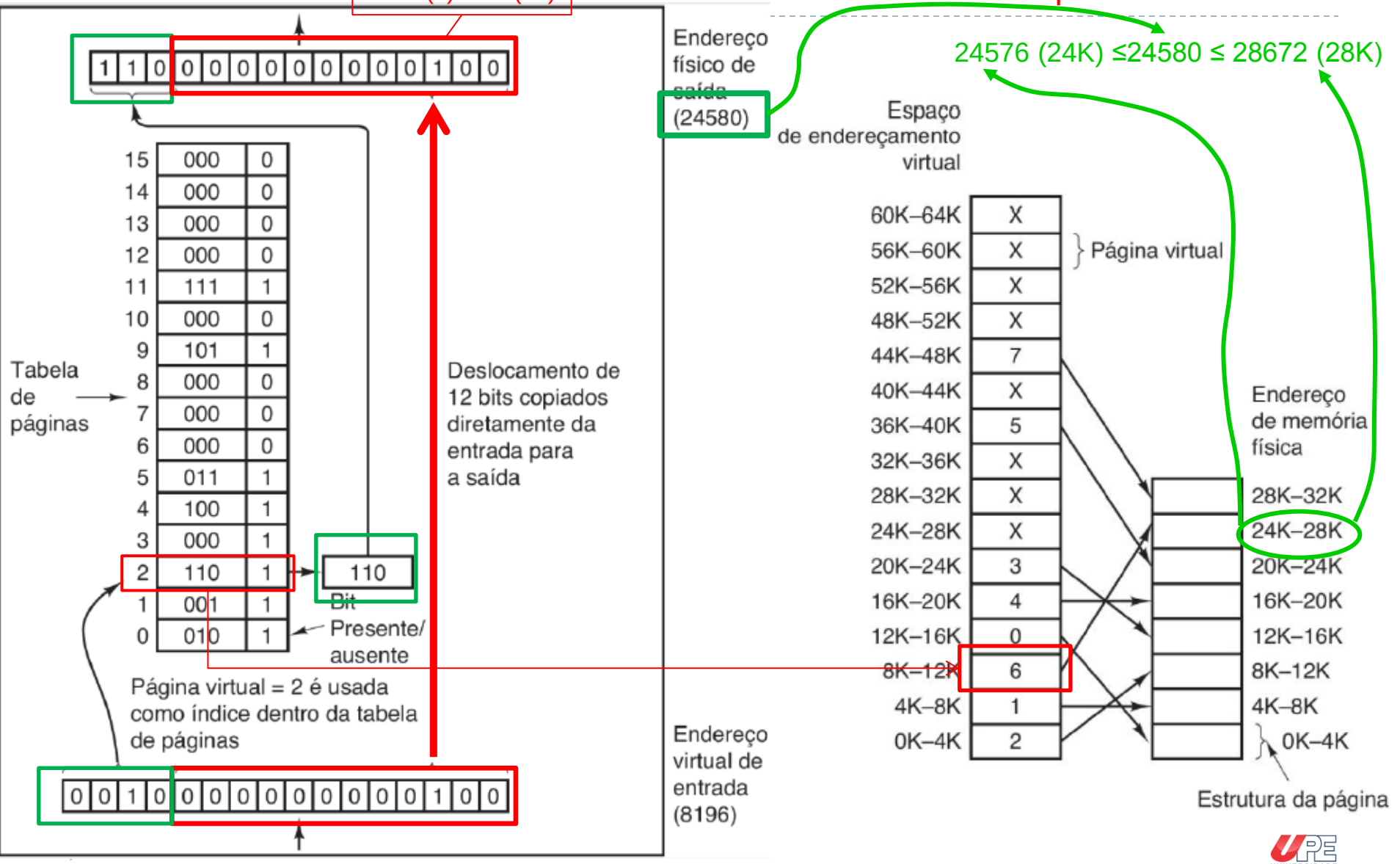


O que acontece?

Falta de página (*page fault*)

MMU

- 4096 = $2^{(12)}$
  - 12 bits para deslocamento



Estrutura de uma entrada na tabela de páginas

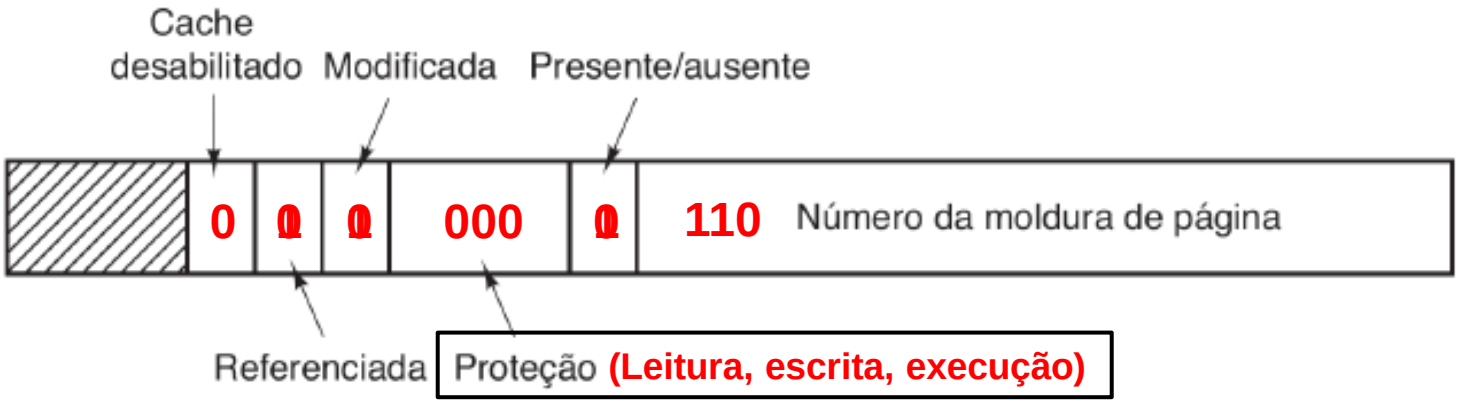
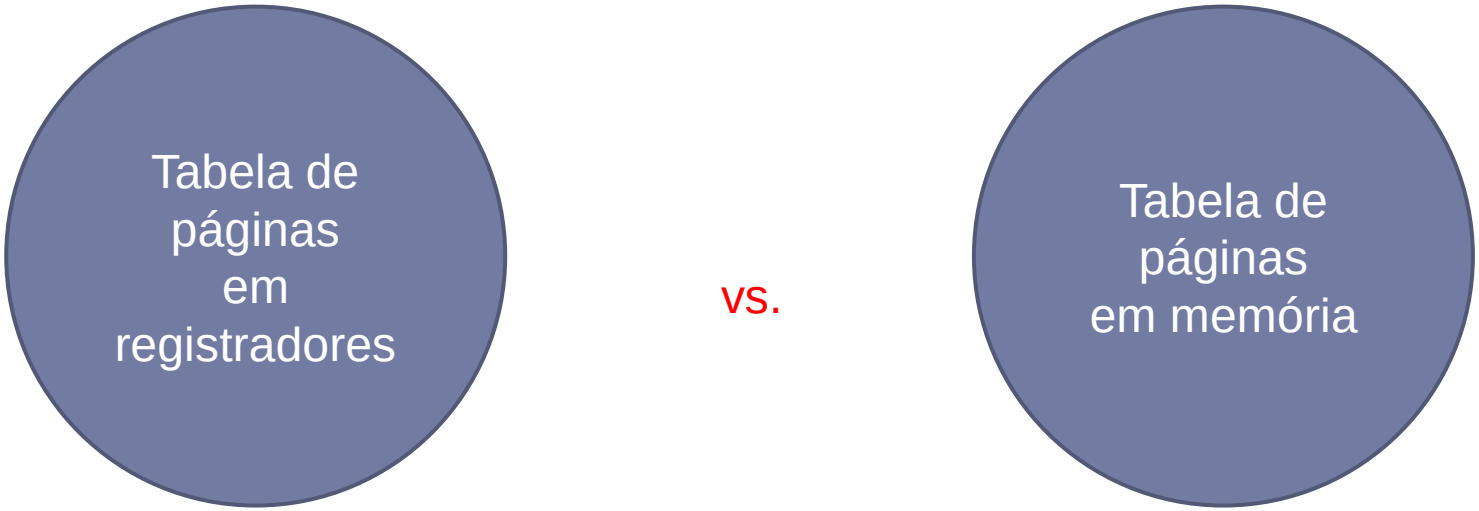


Figura 3.11 Entrada típica de uma tabela de páginas.

Acelerando a paginação

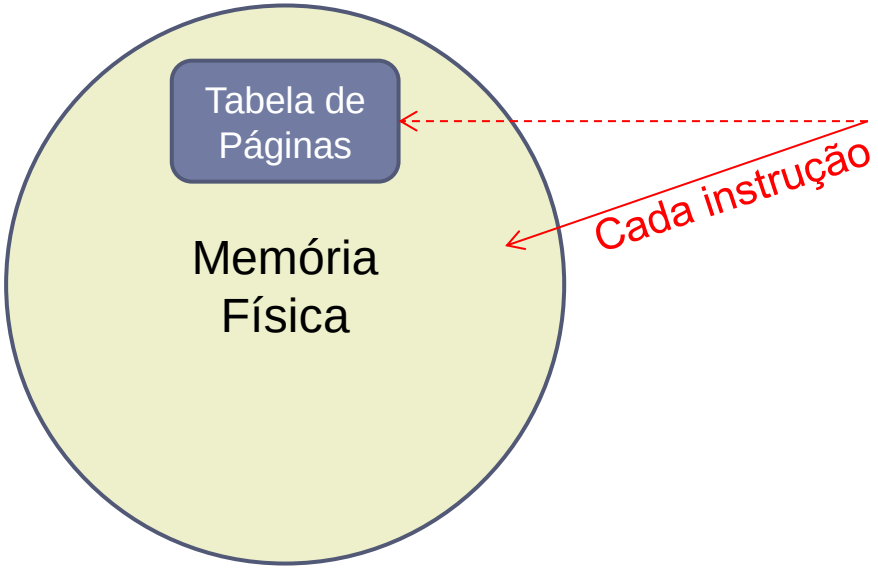
- Em qualquer sistema de paginação, dois problemas importantes devem ser enfrentados:
  - O mapeamento do endereço virtual para o endereço físico deve ser rápido.
  - Se o espaço virtual for grande, a tabela de páginas será grande.



Buffers para Tradução de Endereços - TLB

- Velocidade da execução é limitada pela frequência com que a CPU pode acessar a memória...
  - Reduzindo desempenho pela metade

- Programas tendem a fazer um grande número de referências a um mesmo pequeno conjunto de páginas virtuais (*observação*)



Buffers para Tradução de Endereços - TLB

- TLB (memória associativa)
  - Apenas endereços mais usados
  - Hardware localizado dentro da MMU
  - Compara as entradas paralelamente
  - Ex. Loop
  - Ausência de página (*page miss*)
  - Presença de página (*page hit*)

Válida	Página virtual	Modificada	Proteção	Moldura da página
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Tabela 3.1 Uma TLB para acelerar a paginação.

Tabelas de Páginas Multiníveis

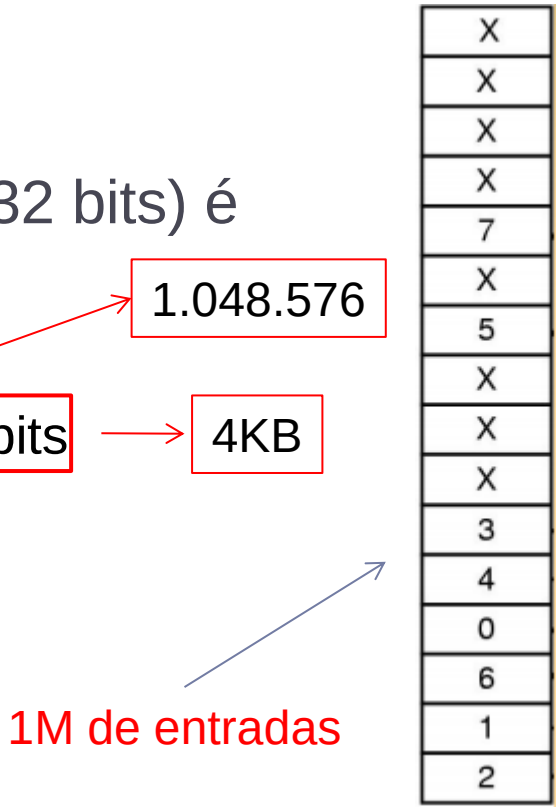
- Com TLB conseguimos resolver o problema do acesso rápido
- Mas e o problema do tamanho das tabelas de páginas? (Endereço virtual muito grande)

- **Paginação Multinível**
  - Quebra o espaço de endereço lógico em múltiplas tabelas de página

Tabelas de Páginas Multiníveis

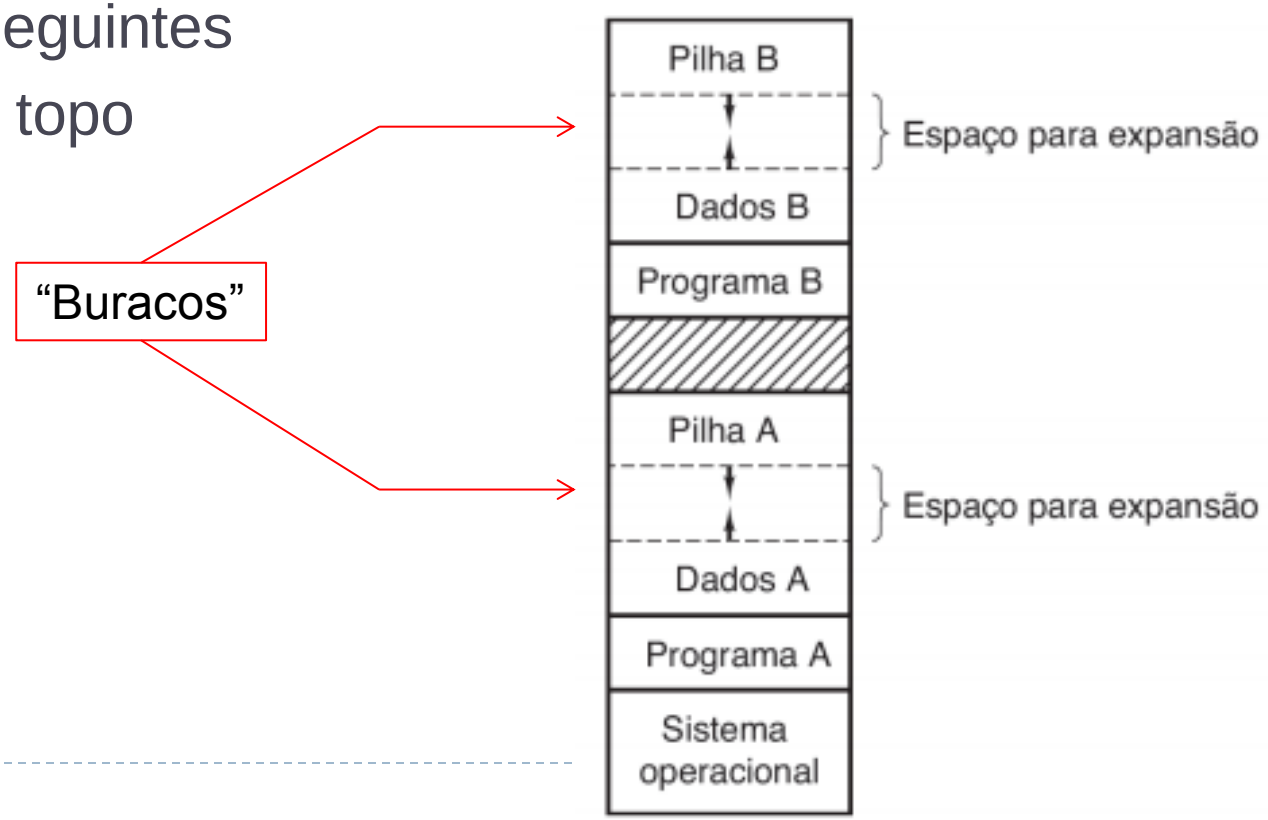
- A ideia é evitar manter na memória todas as tabelas de página o tempo todo

- Exemplo
  - Um endereço lógico (em máquinas de 32 bits) é dividido em:
    - um número de página contendo 20 bits → 1.048.576
    - um deslocamento de página contendo 12 bits → 4KB



Tabelas de Páginas Multiníveis

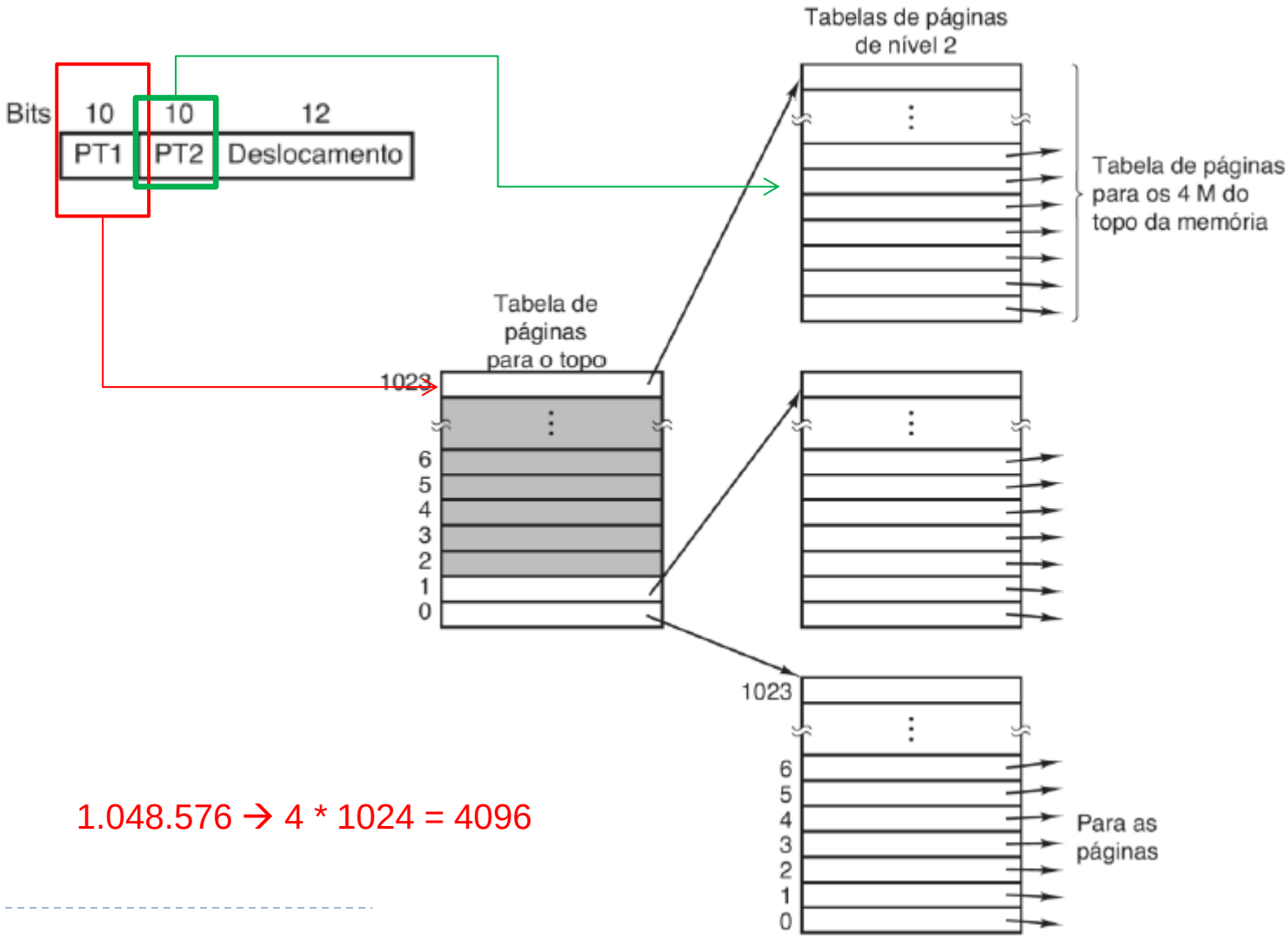
- Exemplo: 1 programa precisa de 12 megabytes
- Programa:
  - Código: 4MB da base da memória
  - Dados: 4MB seguintes
  - Pilha: 4MB do topo



Tabelas de Páginas

1M de entradas

X
X
X
X
7
X
5
X
X
X
3
4
0
6
1
2



1.048.576 → 4 * 1024 = 4096

Exercício

1. O que é falta de página?

2.Qual a função da TLB?

3.Como funciona a tabela de páginas multinível?

Algoritmo de Substituição de Páginas

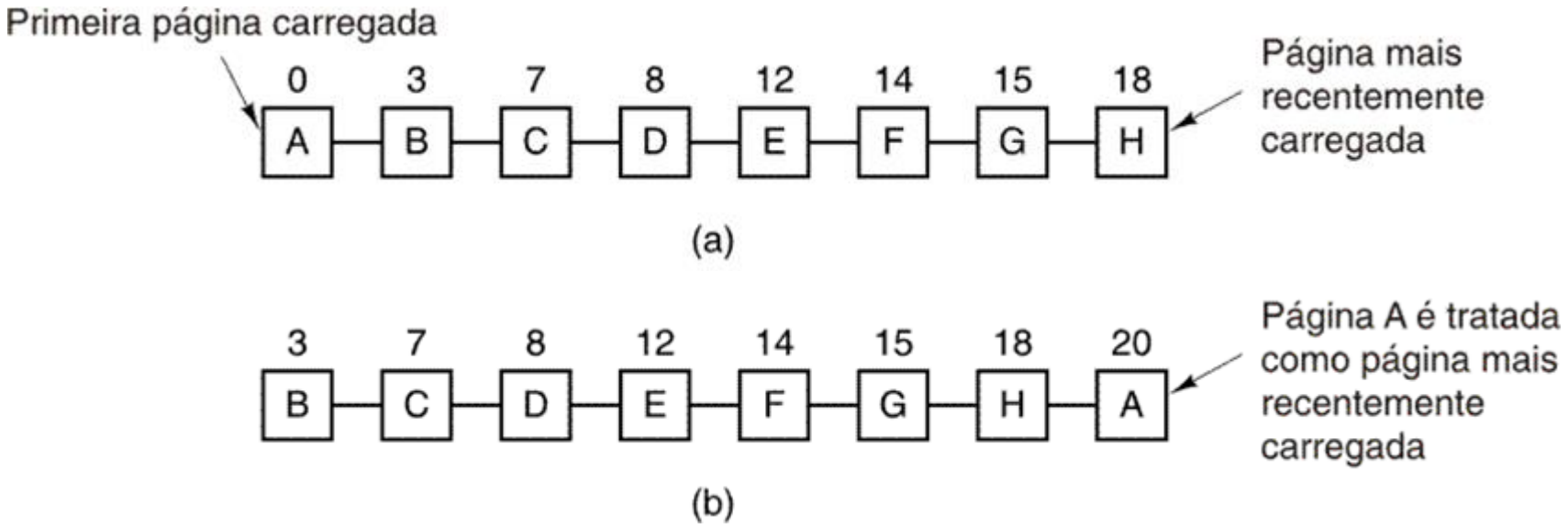
Substituição de Páginas

- Falta de página (*page-fault*) na memória:
  - qual página deve ser removida?
  - alocação de espaço para a página a ser trazida para a memória
- A página modificada deve primeiro ser salva
  - se não tiver sido modificada é apenas sobreposta/descartada
- Melhor não escolher uma página que está sendo muito usada
  - provavelmente precisará ser trazida de volta logo

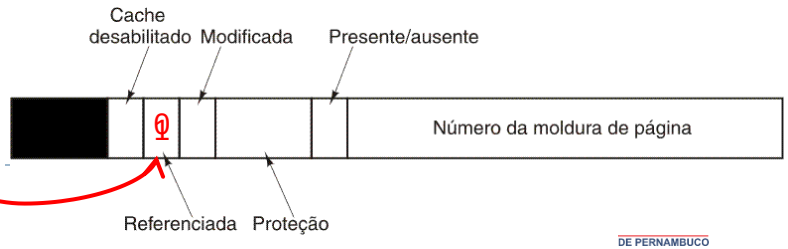
Primeira a Entrar, Primeira a Sair (FIFO)

- Mantém uma lista encadeada de todas as páginas
  - página mais antiga na cabeça da lista
  - página que chegou por último na memória no final da lista
- Na ocorrência de falta de página
  - página na cabeça da lista é removida
  - nova página adicionada no final da lista
- Desvantagem
  - página há mais tempo na memória pode ser usada com muita freqüência

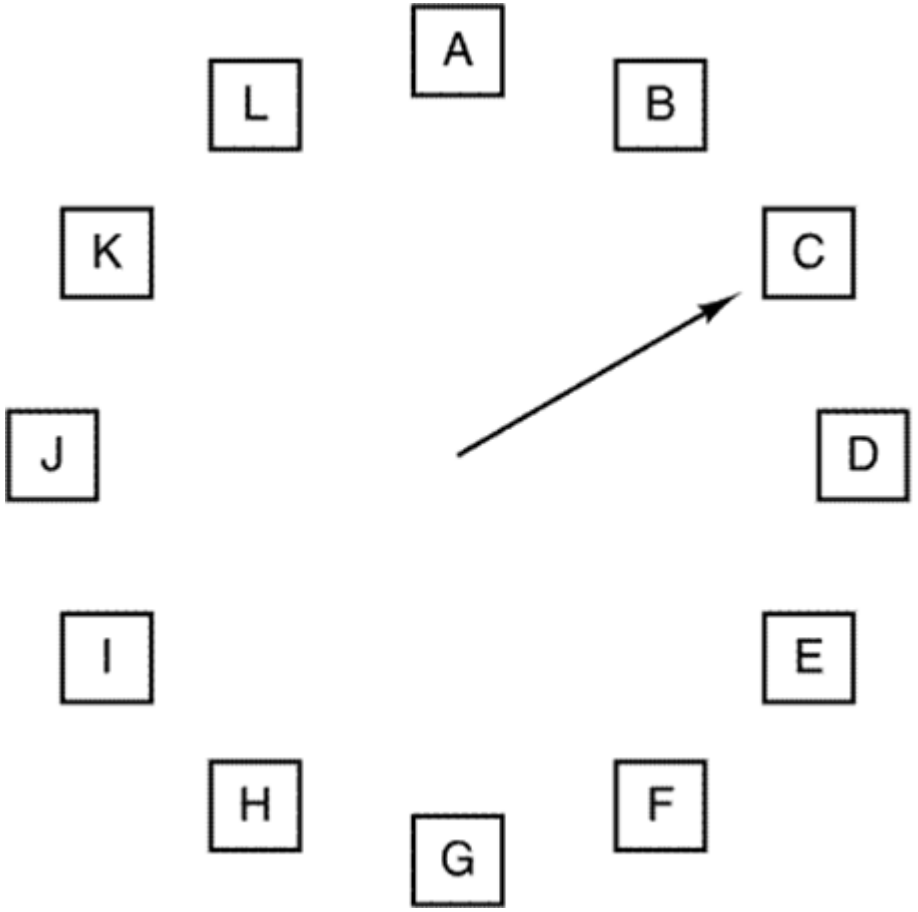
Segunda Chance (SC)



- lista de páginas em ordem FIFO
  - números representam instantes de carregamento das páginas na memória
- Estado da lista em situação de falta de página no instante 20, com o bit R da página A em **1**



Relógio

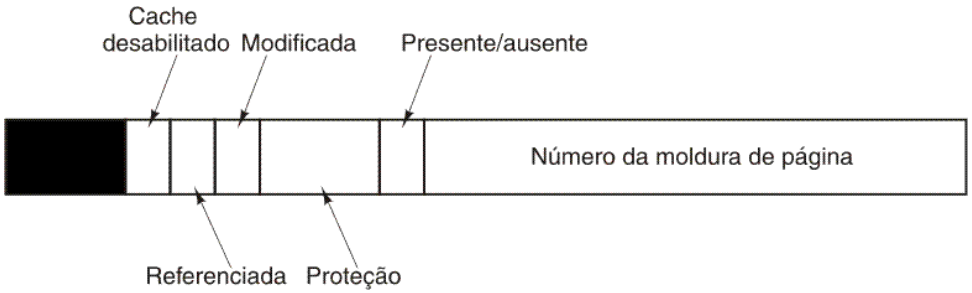


Lista Circular

Quando ocorre uma falta de página,
a página apontada é examinada.
A atitude a ser tomada depende do bit R:
R = 0: Retira a página,
R = 1: Faz R = 0 e avança o ponteiro.

Vantagens em relação a segunda chance por não precisar ficar reinserindo páginas no final da lista

Não Usada Recentemente (NUR/NRU)



- Cada página tem bits Referenciada (R) e Modificada (M)
  - Bits são colocados em 1 quando a página é referenciada e modificada

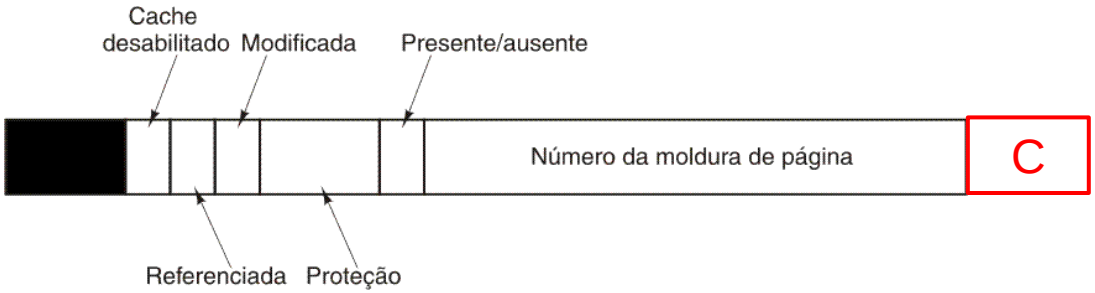
Periodicamente colocado R para 0 (ex. cada tique de *clock*)

- As páginas são classificadas
  - Classe 0: não referenciada (0), não modificada (0)
  - Classe 1: não referenciada (0), modificada (1)
  - Classe 2: referenciada (1), não modificada (0)
  - Classe 3: referenciada (1), modificada (1)

Remove aleatoriamente dentro da classe "mais baixa"

- NUR remove página aleatoriamente
  - da classe de ordem mais baixa que não esteja vazia

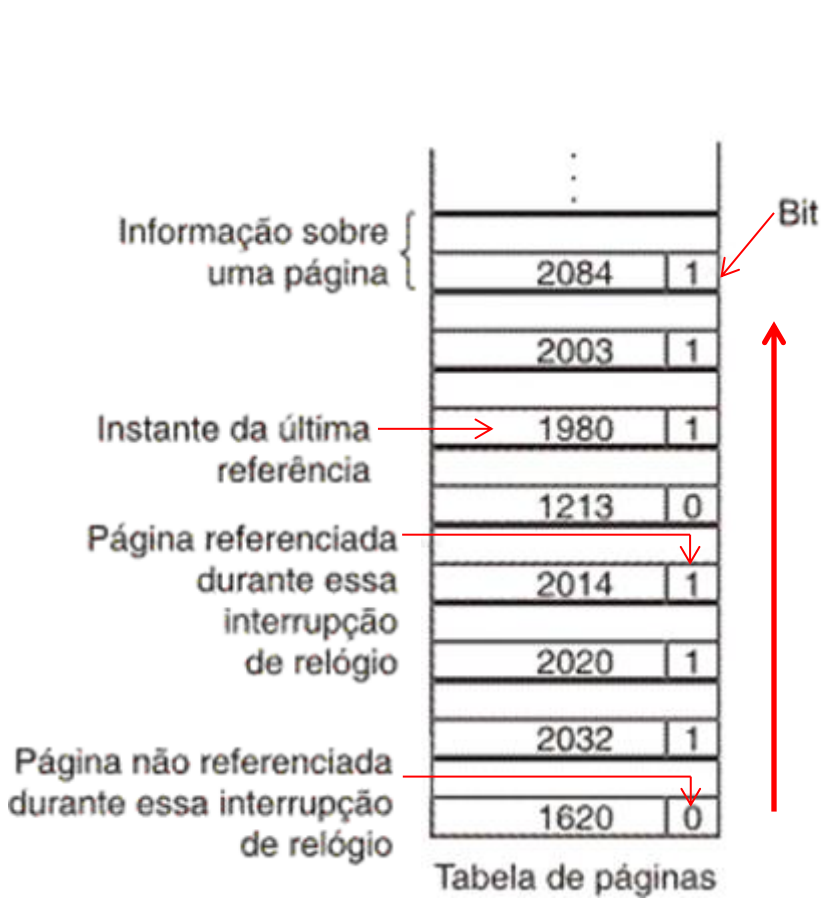
Menos Recentemente Usada (MRU/LRU)



- Assume que páginas usadas recentemente logo serão usadas novamente
  - Premissa baseada em observação/experimentos
  - retira da memória a página que há mais tempo não é usada
- Uma lista encadeada de páginas deve ser mantida
  - página mais recentemente usada no início da lista, menos usada no final da lista
  - atualização da lista à cada referência à memória → custo alto….
- Alternativa: manter contador em cada entrada da tabela de página
  - escolhe página com contador de menor valor
  - zera o contador periodicamente.
    - Porque fazer isso?

Conjunto de Trabalho (WS)

Conjunto de páginas que um processo está usando atualmente é denominado conjunto de trabalho



Algoritmo

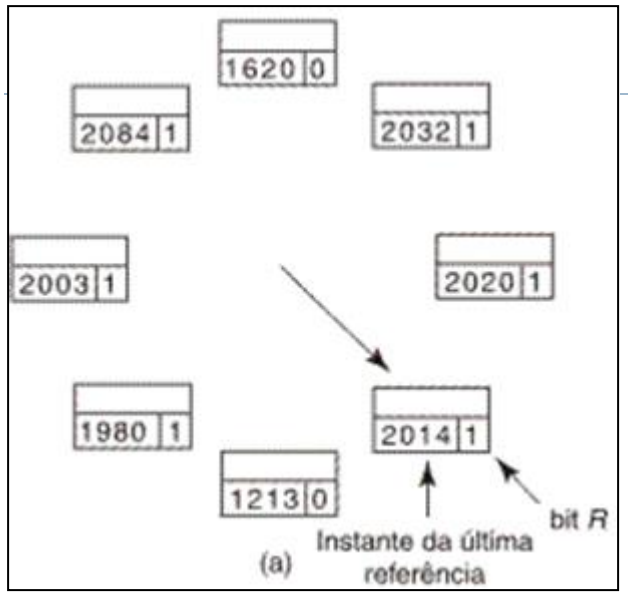
Varra todas as páginas examinando o bit R:

se (R == 1)
faça Instante da última referência igual ao tempo virtual atual

se (R == 0 e idade > t)
remova esta página

Se (R == 0 e idade ≤ τ)
lembre-se do menor tempo → (maior idade)

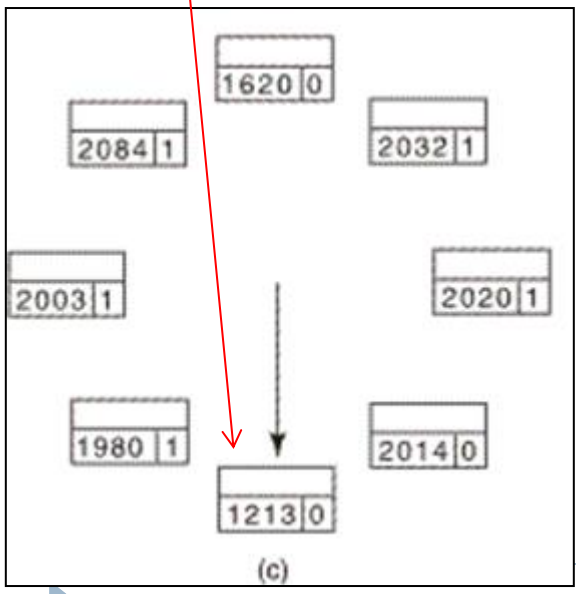
WSClock

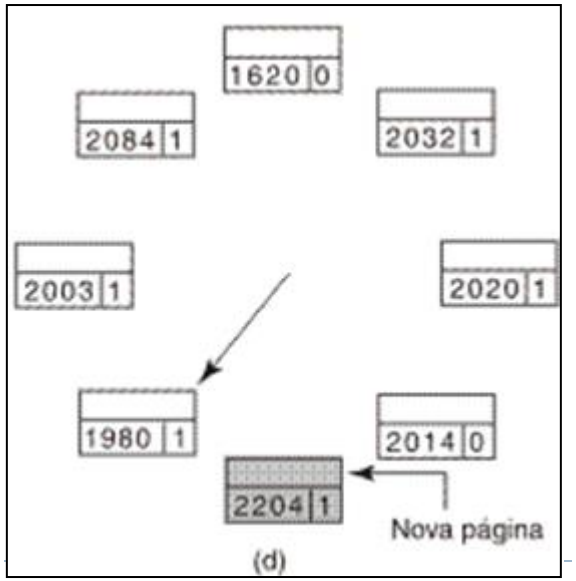


Assim como no **WS**:

Se (R==0 e idade > t) (com M = 0)
remova esta página
// pois ela está fora do conjunto de trabalho e há uma cópia válida em disco

idade = tempo virtual atual - instante da última referência





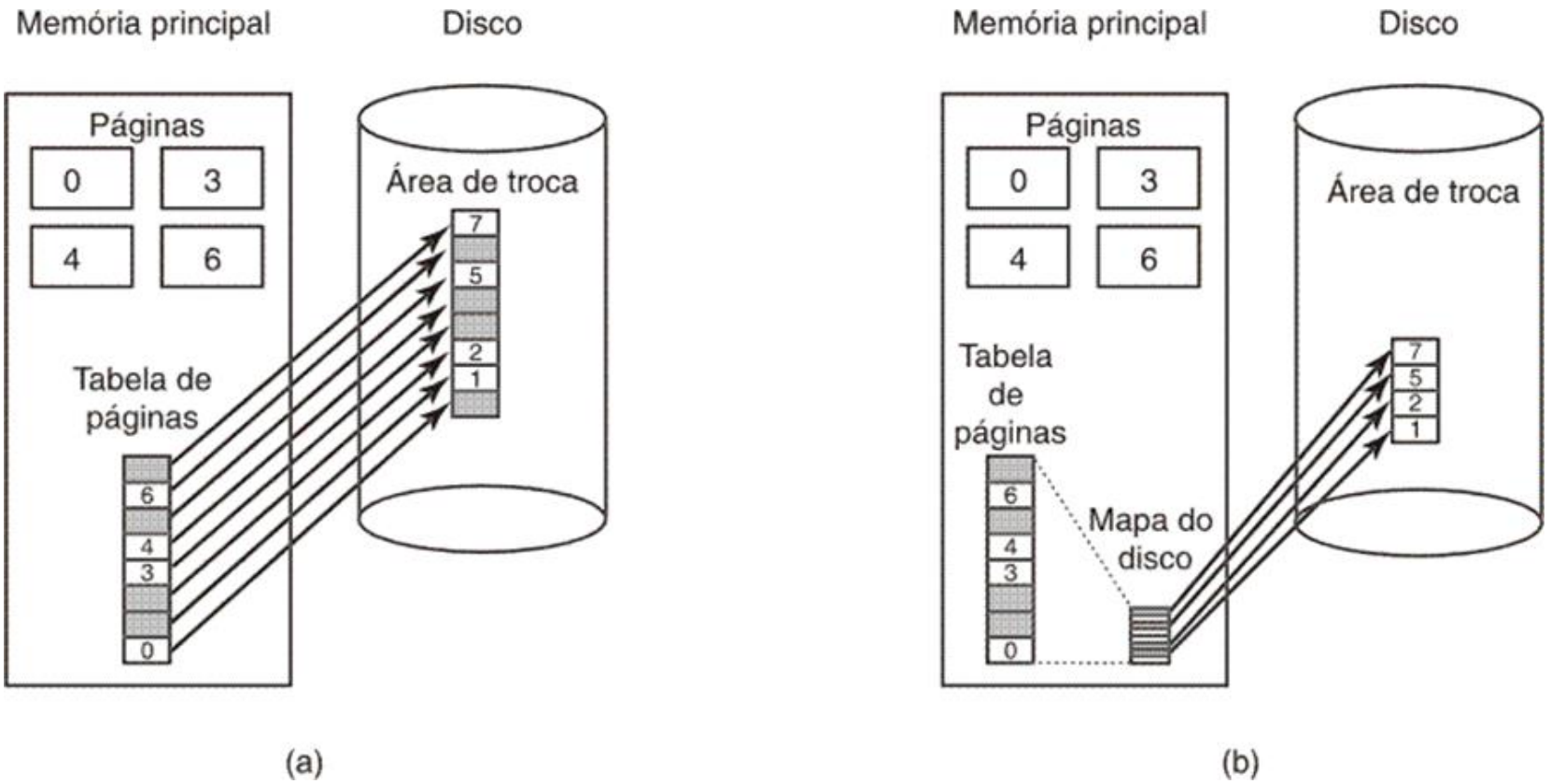
E se a página estiver suja?
É preciso salvá-la em disco...
E o tempo de espera?

O Algoritmo Ótimo!

- O algoritmo FIFO sempre seleciona a **página mais antiga** para ser trocada – First-In, First-Out
- O algoritmo LRU sempre seleciona a **página que não vem sendo usada há mais tempo** – Least Recently Used (Menos Recentemente Usada - MRU)
- O algoritmo ótimo sempre seleciona a **página que não será usada por mais tempo**...
  - Mas como o SO pode determinar quando cada uma das páginas será referenciada? Daqui a 10 instruções, 100 instruções, 1000 instruções...
  - **IMPOSSÍVEL!!!**

Memória Secundária

- Área de troca (*Swap Area*)



(a) Paginação para uma área de troca estática
(b) Páginas alocadas dinamicamente em disco

Exercício

1.Diferencie o algoritmo de substituição de página não usado recentemente e o algoritmo de substituição de página menos usado recentemente.

2.Qual o principal problema encontrado no algoritmo de substituição de página primeiro a entrar, primeiro a sair?

3.Como funciona o algoritmo de substituição de página segunda chance?

4.Explique o funcionamento do algoritmo de substituição de página de conjunto de trabalho.

Exercício

- Um computador com um endereçamento de 32 bits usa tabela de paginas de dois níveis. Os endereços são quebrados em um campos de 9 bits para a tabela de paginas de nível 1, um campo de 11 bits para a tabela de paginas de nivel 2 e um deslocamento. Qual o tamanho das paginas e quantas existem no espaço de endereçamento citado?

Exercício

- Se o algoritmo de substituição **FIFO** é usado com **quatro** molduras de página e **oito** páginas virtuais, quantas faltas de página ocorrerão com a cadeira de referências **0172327103** se os quatros quadros estão inicialmente vazios?

- E se o algoritmo for **LRU**?

Sistemas Operacionais
Gerenciamento de Memória

Carlos Ferraz (cagf@cin.ufpe.br)
Jorge Cavalcanti Fonsêca (jcbf@cin.ufpe.br)