AULA Nº 11
SISTEMAS OPERACIONAIS

Técnicas de Memória Virtual

Contextualizando

Vimos

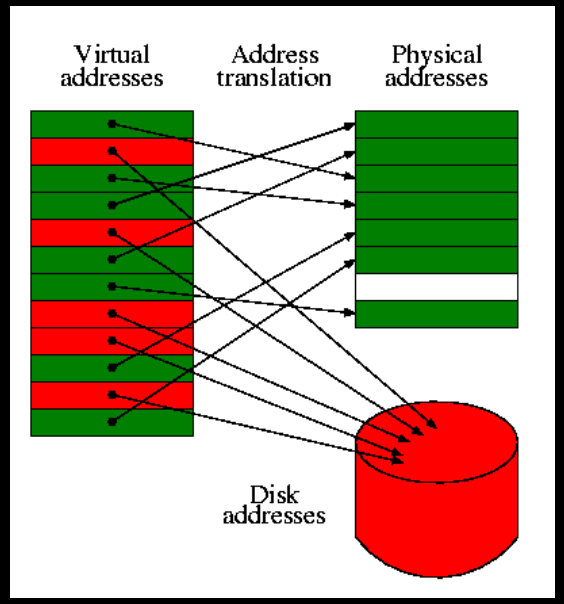
Introdução ao Gerenciamento de Memória

Agora

Técnicas de Memória Virtual

O que é Memória Virtual (MV)?

- **É uma técnica que usa a memória secundária como uma "cache" para partes do espaço dos processos.**
- Por que memória virtual?
  - O tamanho do software cada vez maior.
  - Maior grau de multiprogramação.
  - Executa programas maiores que a RAM.
- **Um processo usa endereços virtuais** e não físicos.
  - Utiliza o MMU para conversão.



Técnicas de MV

- **Paginação**
  - Blocos de tamanho fixo (e.g. 4KB).
  - O espaço de endereçamento virtual é dividido em páginas virtuais.
- **Segmentação**
  - Blocos de tamanho arbitrário chamado de segmentos.
  - Contém mesmo tipo de informações (e.g. dados, pilha)
- Mapeamento entre endereços reais e virtuais (MMU).
- Muitos SOs usam uma “mistura” das duas técnicas.

Paginação

- **Páginas** – unidades de tamanho fixo no dispositivo secundário.
- **Frames** – unidades correspondentes na memória física (RAM).
- **Page fault** – é o evento quando uma página que não está na RAM é referenciada.
  - Usa uma *trap* para carregar ou substituir uma página.
- **Tabela de Páginas** – estrutura para mapear uma página ao frame correspondente.
  - Cada processo tem um.

Exemplo de Paginação

- Um sistema que gera 64K de endereços virtuais (16 páginas e 8 frames).
- MMU faz o mapeamento.
- MOV REG,5
- Ela está mapeada à terceira frame, que começa em 8k = 8192.
- O endereço enviado ao barramento é 5 + 8192 = 8197.

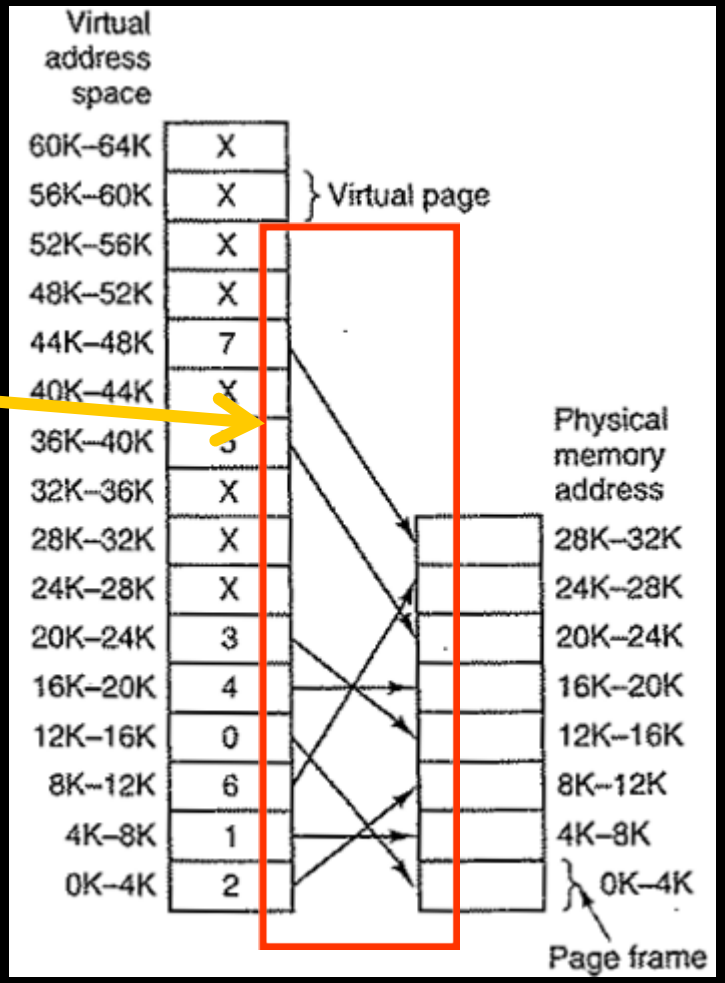
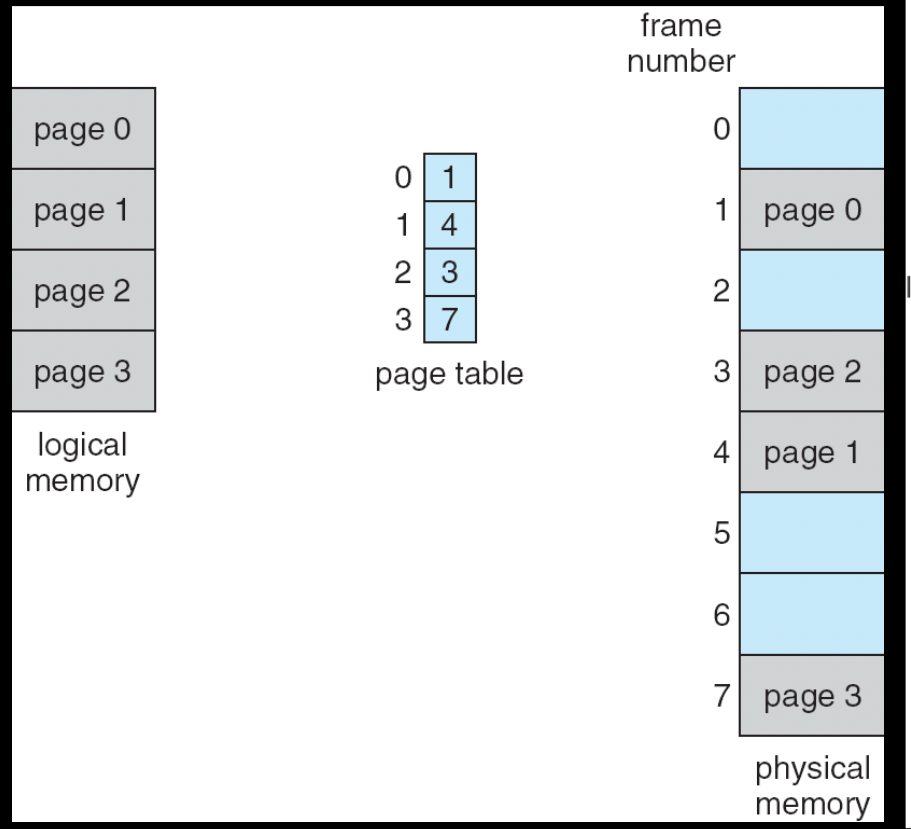
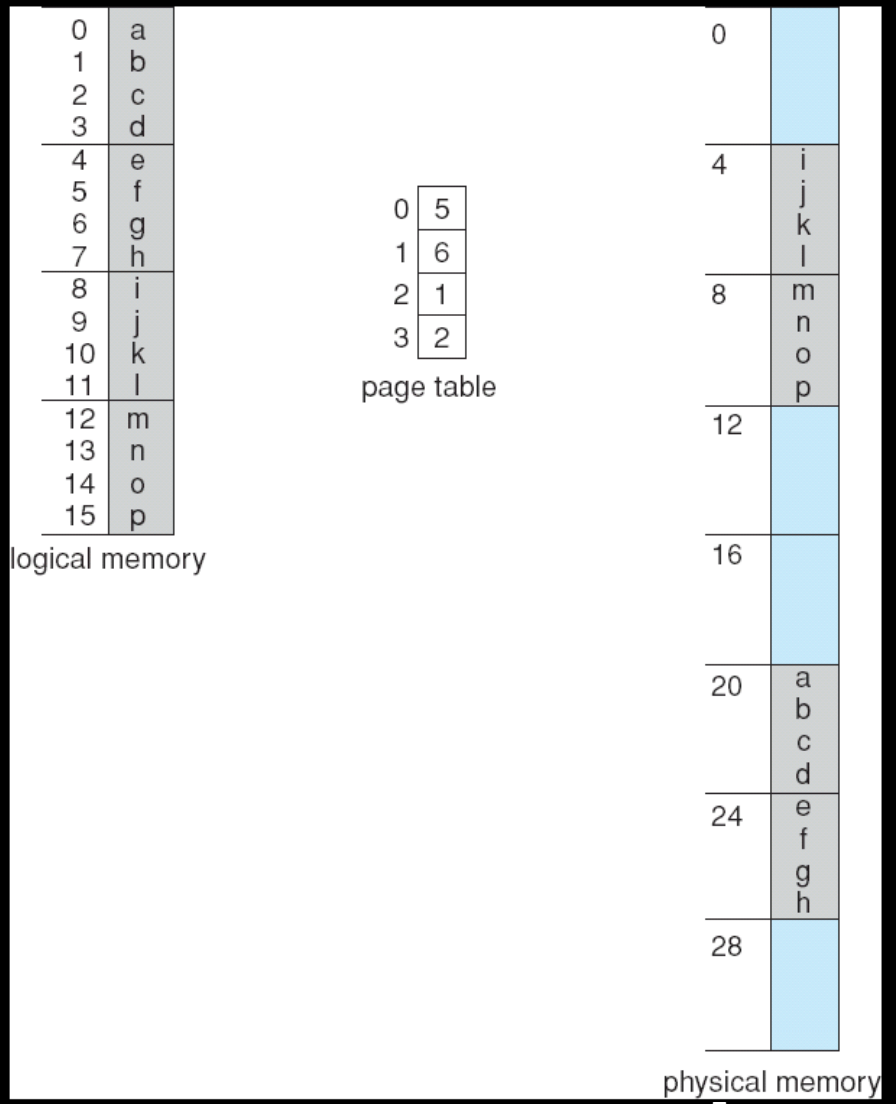


Tabela de Páginas

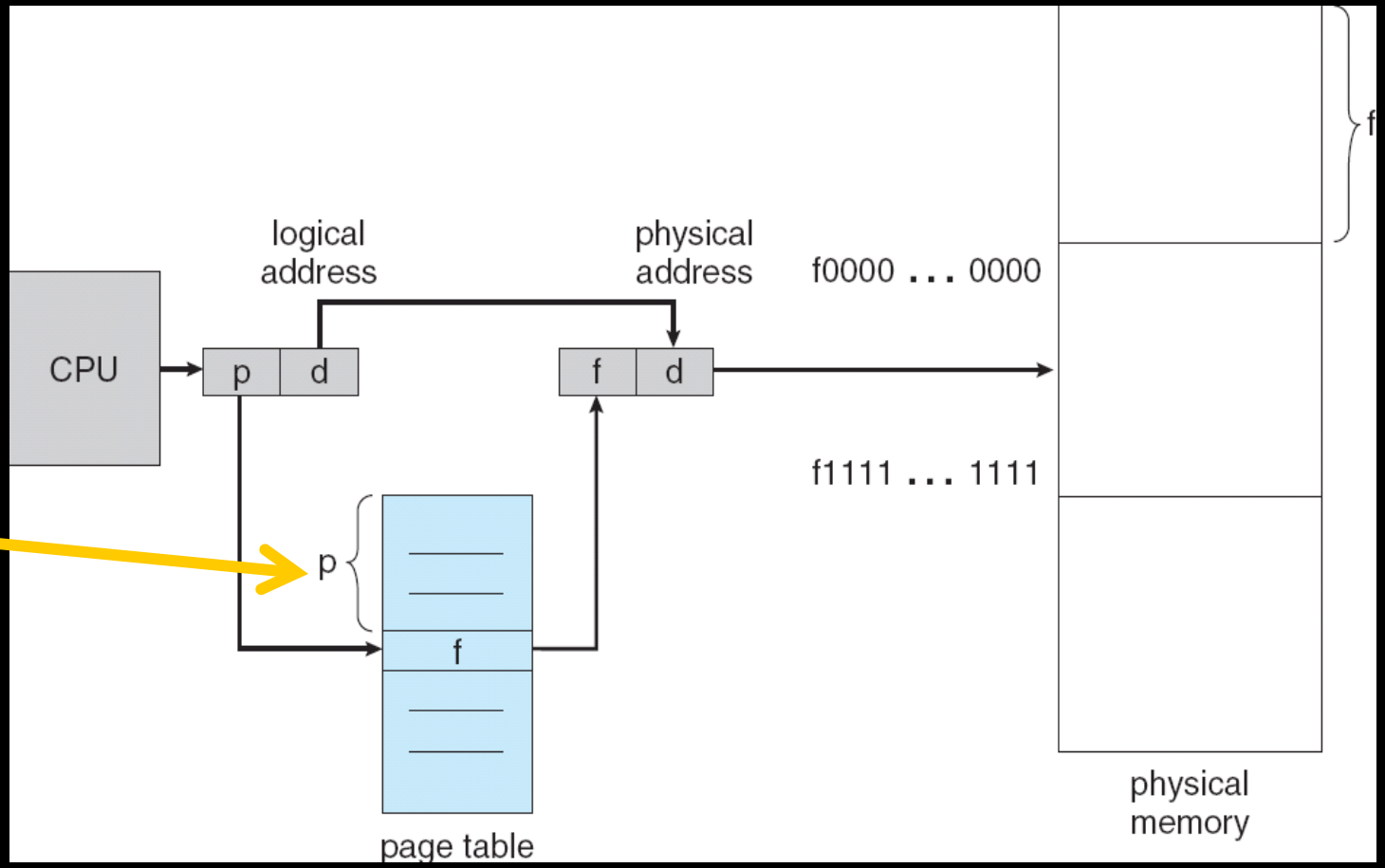
- **Argumento de entrada** -> # página virtual
- **Argumento de saída** -> # página real





Busca de um Endereço

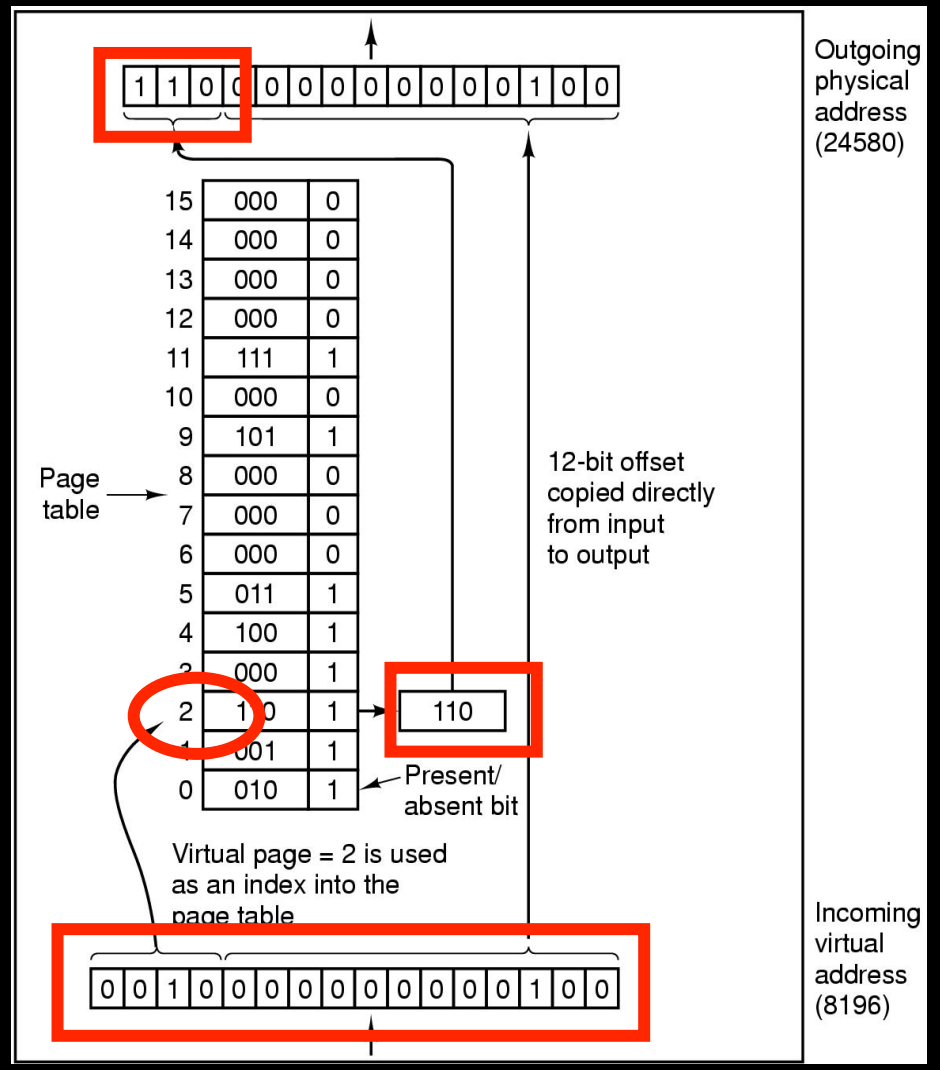
- Busca seqüencial? Binária?
- Qualquer que seja a alternativa, é lenta.
- Ideal – seria # página servir como índice na tabela.



Exemplo de uma Busca

- Endereço 0010000000000100

- MMU com 16 páginas de 4KB.
- Endereço virtual de 16 bits.
- A tabela tem 16 entradas (0000 a 1111).
- Hardware com 8 frames.
- Endereço virtual de 15 bits.



Componentes do Endereço

- **Número de página (p)** – usado como um índice para uma tabela de página .
- **Deslocamento de página (d)** – **combinado com endereço de base** para definir o endereço de memória físico que é enviado à unidade de memória.

- **Páginas maiores**: leitura mais eficiente, tabela menor, mas fragmentação interna.
- **Páginas menores**: leitura menos eficiente, tabela maior, mas menor fragmentação.

Componentes da Tabela

- **Page frame number** – identifica (número) a página real.
- **Bit de Residência (presente/ausente)** – se 1, então página correspondente é válida e está na RAM; **page fault**?
- **Bits de proteção** – 0 (leitura/escrita) 1 (leitura) 2 (execução).
- **Bit de modificação** – 1 (página alterada) 0 (não alterada).
- **Bit de referência** – 1 (foi referenciada "recentemente").
- **Bit de cache** – permite desabilitar o *caching* da página.

Onde Armazenar as Tabelas?

- **Array de Registradores**, se a memória for pequena
  - Mantidos no hardware.
- **Na própria memória RAM**
  - A MMU gerencia utilizando um ou dois registradores.
- **Em uma memória cache na MMU** chamada Memória Associativa.
  - Usada para melhorar o desempenho da tabela na RAM.

Tabela na RAM

- Usa dois registradores:
  - **Registrador de base da tabela de página** (PTBR)
    - Aponta para o início da tabela, indicando o endereço físico de memória onde a tabela está alocada.
  - **Registrador de tamanho da tabela de página** (PTLR)
    - Indica tamanho da tabela de página (número de entradas da tabela → número de páginas).
- Problema: **Dois acessos para instrução/dados na RAM**:
  - Um para a tabela e outro para o dado/instrução em si.
- Solução: cache chamado TLB.

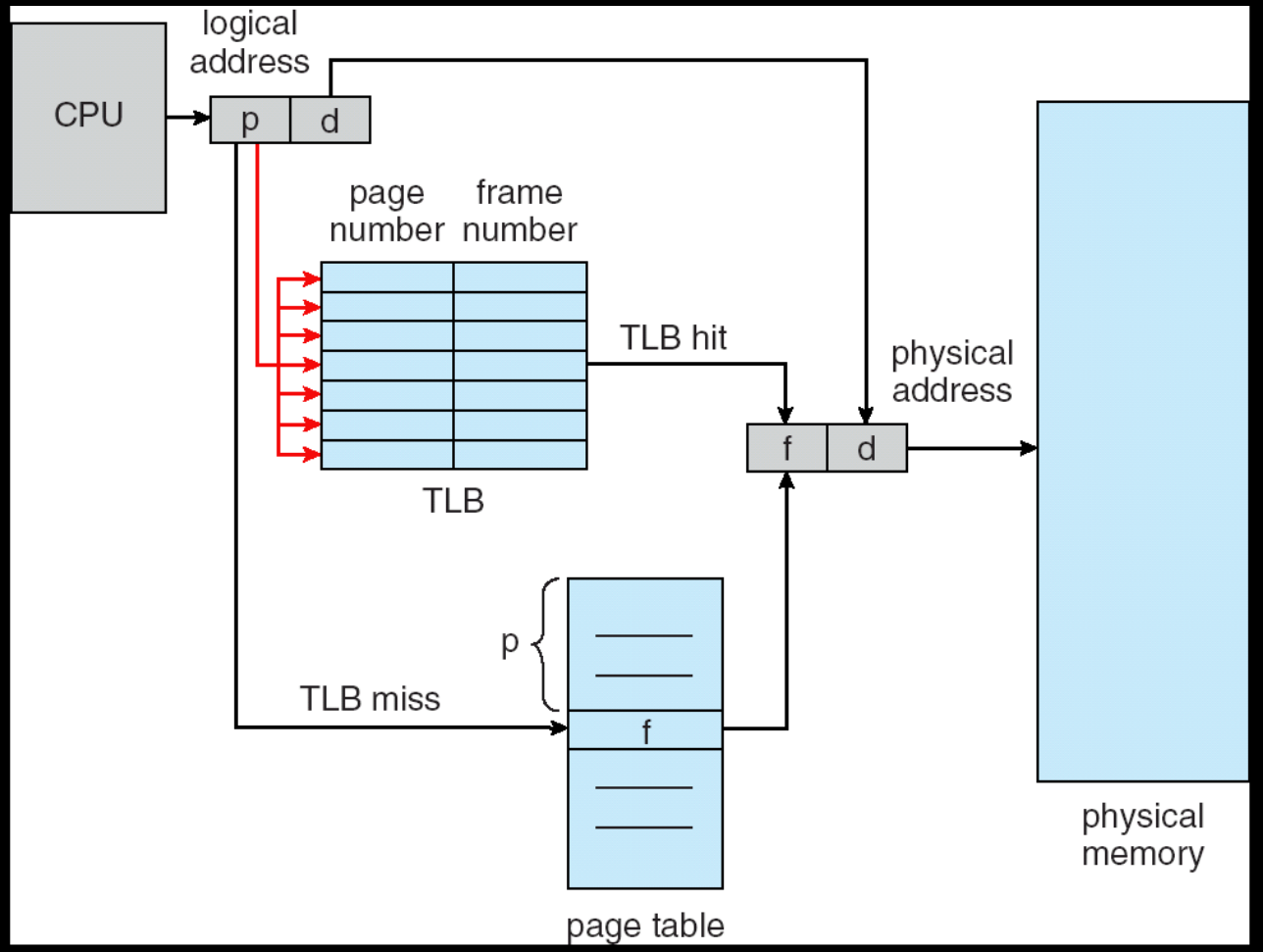
Translation Lookaside Buffer (TLB)

- Cache da tabela das páginas mais usadas (hardware).

Valid	Virtual page	Modified	Protection	Page frame
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

Funcionamento da TLB

- Pode ser implementada em **hardware** ou **software.**
- Em hardware é mais **rápido**, mas **ocupa um espaço** que poderia ser usado para outras funções, como **cache**.



Dois Tipos de Falha de Página

- **Soft miss**
  - Quando a página referenciada não está na TLB, mas na RAM.
  - Basta atualizar a TLB.
- **Hard miss**
  - A página não está na memória física (e nem na TLB).
  - Trazer do disco à RAM (e então à TLB).
  - Muito lento.

Paginação

Contextualizando

Vimos

Técnicas de Memória Virtual

Agora

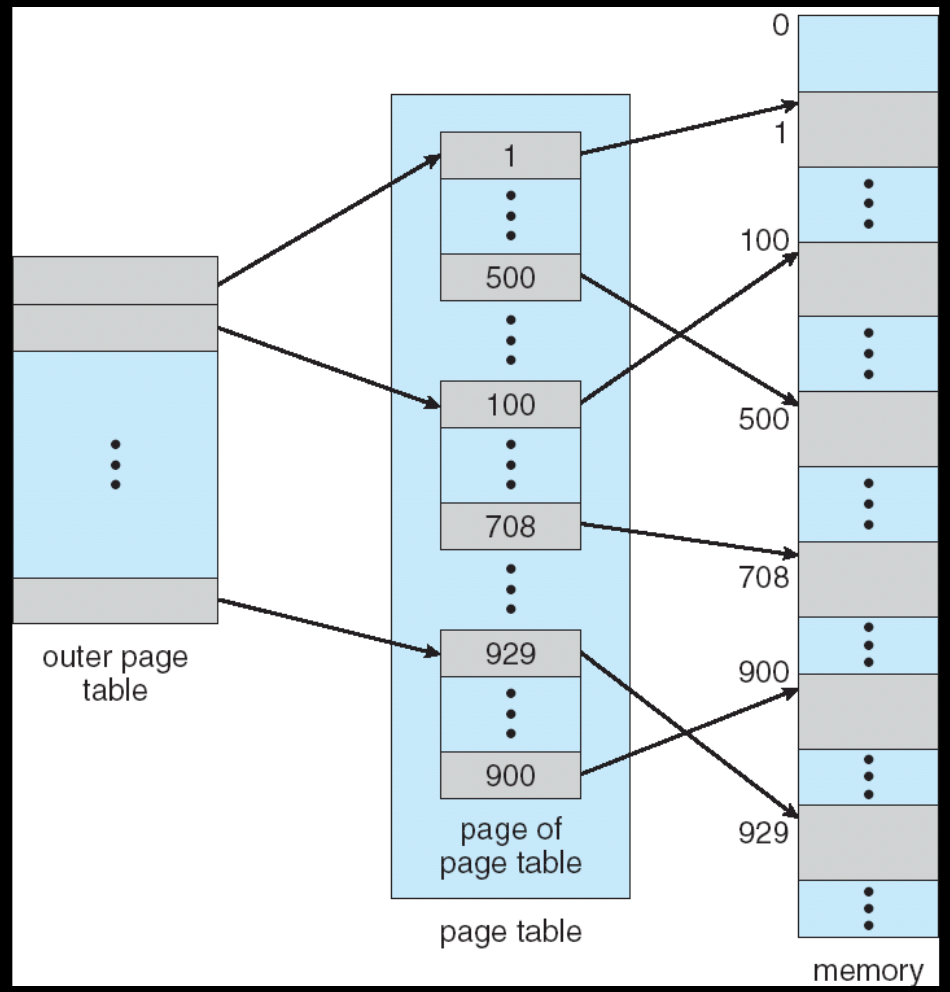
Paginação

Como Organizar a Tabela de Páginas?

- Problema com Tabelas de Páginas grandes.
- Resultado de RAMs atuais que são de grande capacidade.
- Estruturas da Tabela de Páginas:
  - Paginação Hierárquica (Multi-nível)
  - Tabelas de Página com Hash
  - Tabelas de Páginas Invertidas

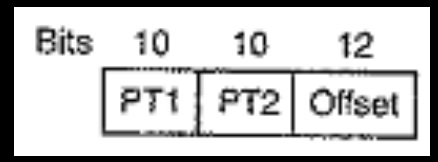
Paginação Hierárquica

- Quebre o espaço de endereço lógico em múltiplas tabelas de páginas.
- Uma técnica simples é uma tabela de página em dois níveis.
- **Manter apenas a parte da tabela necessária.**
- Ocupa menos espaço para o próprio SO.



Endereçamento em Dois Níveis

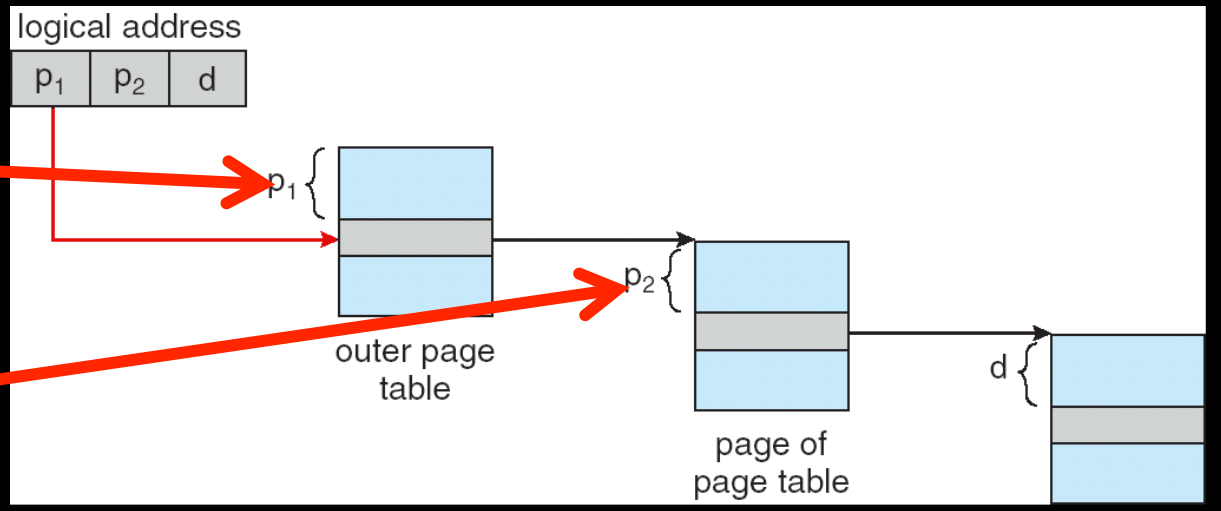
- Um endereço lógico (em máquinas de 32 bits) é dividido em:
  - um número de página contendo 20 bits.
  - um deslocamento de página contendo 12 bits.



- Como a tabela de página é paginada, o número de página é dividido ainda em:
  - Número de página PT1 (10 bits) – índice da tabela mais externa.
  - Número de página PT2 (10 bits) – deslocamento da tabela mais externa.

Endereçamento em Dois Níveis

- Como funciona o endereçamento?
  - Número de página PT1 (10 bits) – **índice da tabela mais externa**.
  - Número de página PT2 (10 bits) – **deslocamento da página dentro da tabela mais externa**.



Tabelas de Página em Hash

- **O número de página virtual é usado para função Hash.**
- Cada entrada na tabela contém uma lista ligada de elementos, consistindo de:
- O # da página virtual.
- O # da moldura (frame).
- Um ponteiro para o próximo.

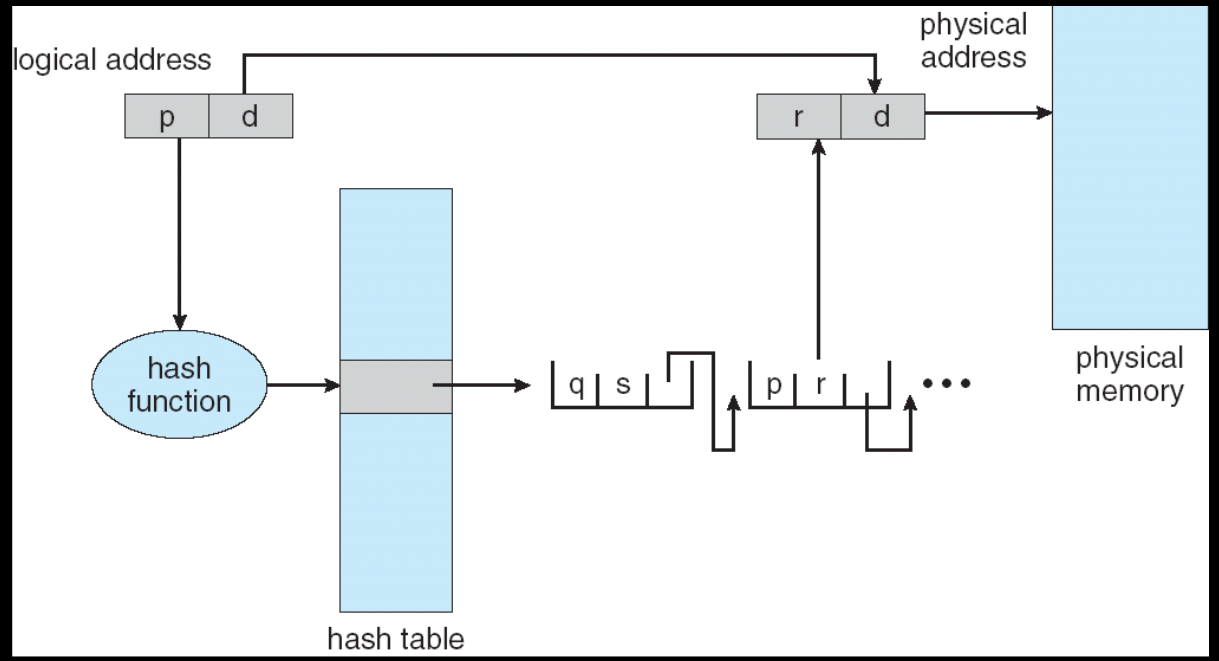
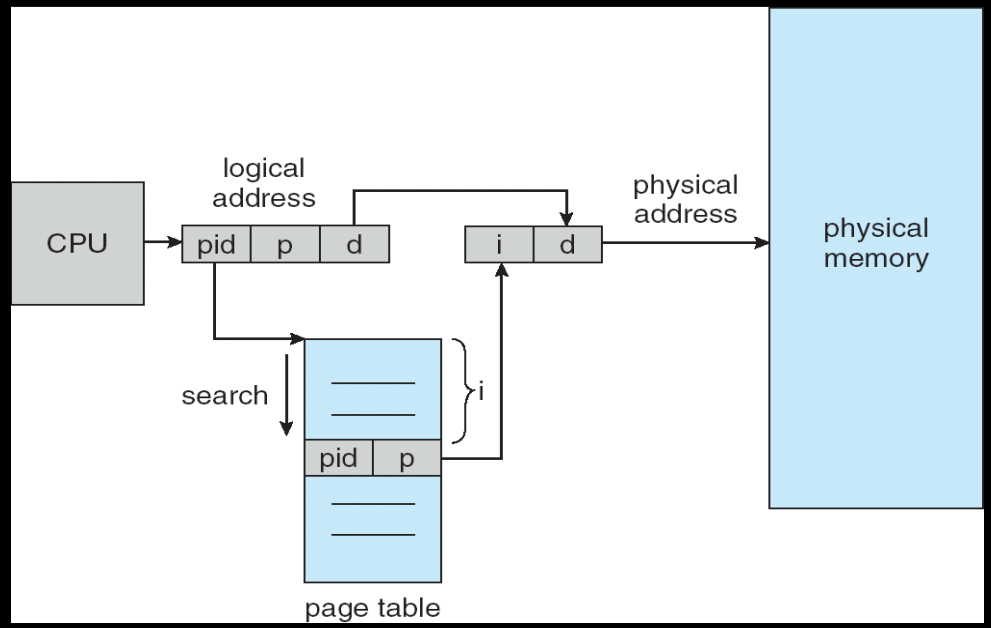


Tabela de Página Invertida

- Possui uma entrada por moldura na memória física.
  - Em vez de uma entrada por página no espaço virtual.
  - **A entrada inclui o processo e a página virtual.**
  - **O deslocamento na leitura é o índice do frame.**
- Poupam muito espaço quando o **espaço virtual é muito maior que o físico**



Problemas da Tabela Invertida

- Fazer leitura de toda a tabela a cada acesso na memória.
- Aumenta o tempo para ler a memória.
- O que fazer?
- Usar a TLB para guardar as mais acessadas.
- Caso a página buscada não esteja na TLB, devemos procurar em toda a tabela invertida.

Alocação de Páginas: Fixa

- 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**; (ii) número muito grande de páginas reais **causa desperdício de memória principal.**

Alocação de Páginas: Dinâmica

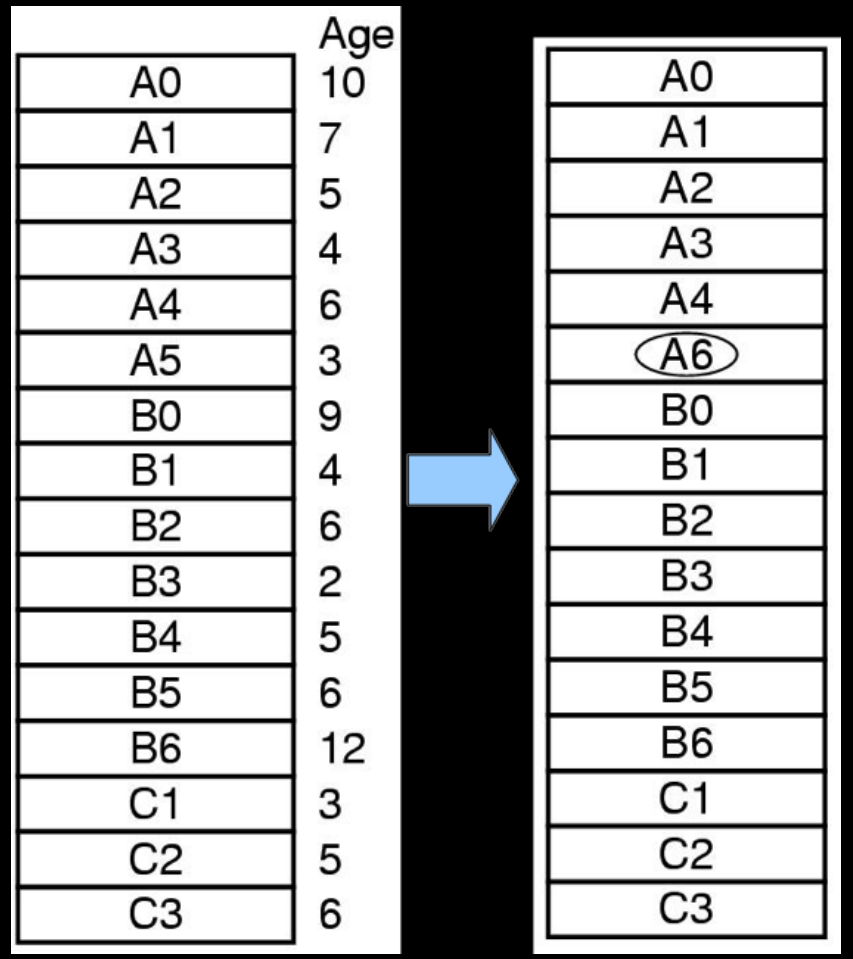
- **Número máximo de páginas reais alocadas ao processo varia durante sua execução.**
- **Vantagem: (i) processos com elevada taxa de paginação podem ter seu limite de páginas reais ampliado; (ii) processos com baixa taxa de paginação podem ter seu limite de páginas reais reduzido.**
- **Desvantagem: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

Busca de Páginas

- **Paginação simples**:
  - Todas as páginas virtuais do processo são carregadas para a memória principal.
- **Paginação por demanda** (Demand Paging):
  - Processos começam com nenhuma página na memória.
  - Assim que a CPU tenta executar a primeira instrução, gera um *page fault*.
  - O SO traz a página que falta à memória.

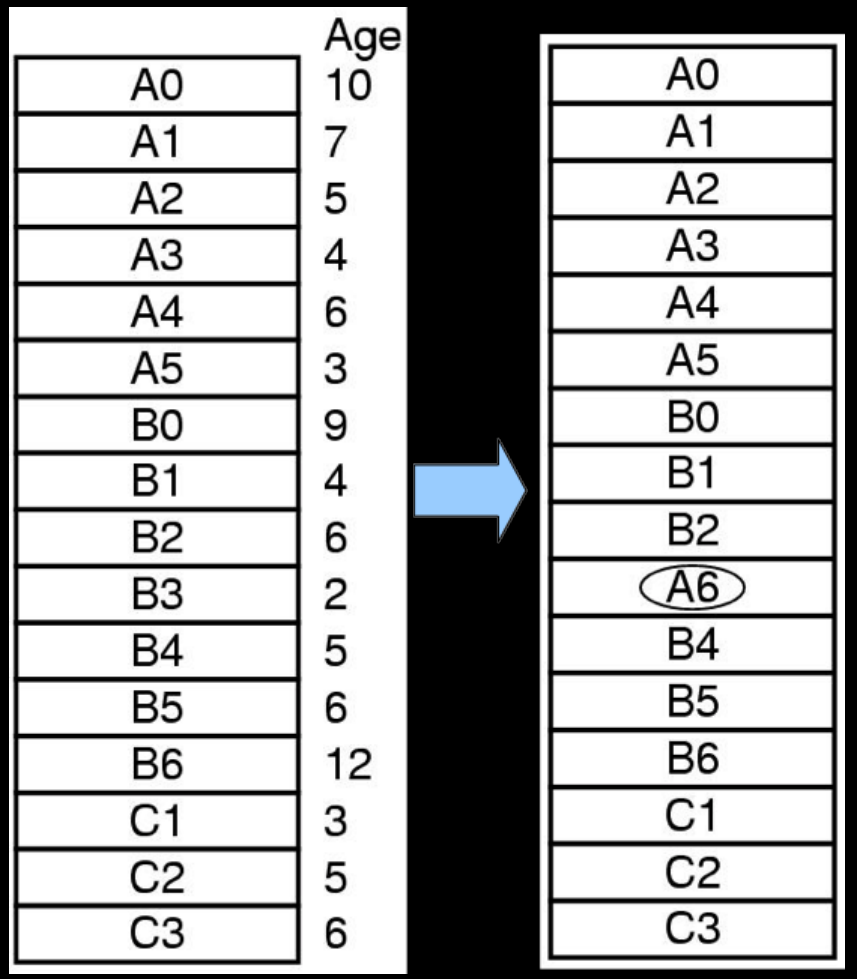
Política de Substituição

- **Local**
  - Considera apenas o processo em questão.



Política de Substituição

- **Global**
  - Leva em conta os processos executáveis.



Algoritmos de Substituição de Página

- Ótimo
- NRU
- FIFO
- Segunda Chance
- Relógio
- LRU
- Working set
- WSClock

Algoritmos de Substituição de Páginas

Contextualizando

Vimos

Paginação

Agora

Algoritmos de Substituição de Páginas

Algoritmos de Substituição de Páginas

- Ótimo
- NRU
- FIFO
- Segunda Chance
- Relógio
- LRU
- Working set
- WSClock

Algoritmo Ótimo

- **Cada página é marcada com o número de instruções que serão executadas antes que a página seja referenciada.**
- **Retira da memória a página que tem menos chance de ser referenciada.**
- **Praticamente impossível de se saber.**
- **Usado em simulações para comparação com outros algoritmos.**

Algoritmo Not Recently Used Page Replacement (NRU)

- Para o SO coletar estatísticas de página de uso:
- Dois bits à página → R(eferenciada) e M(odificada)
  - Classe 0 (00) → não referenciada, não modificada;
  - Classe 1 (01) → não referenciada, modificada;
  - Classe 2 (10) → referenciada, não modificada;
  - Classe 3 (11) → referenciada, modificada;
- Referenciada → lida ou escrita
- Modificada → escrita

Algoritmo Not Recently Used Page Replacement (NRU)

- R e M são atualizados a cada referência à memória.
- Armazenados em cada entrada da tabela de página.
- No início, ambos R e M são 0 para todas suas páginas.
- Periodicamente, o bit R é limpo
  - Marcar as que não foram referenciadas recentemente.
- A cada interrupção de relógio, por exemplo.
- O bit M não é limpo, pois o S.O. precisa saber se deve escrever a página no disco.

Algoritmo do FIFO

- SO mantém uma fila das páginas correntes na memória.
- A página no início da fila é a mais antiga e a página no final é a mais nova.
- Quando ocorre um *page fault*, página do início é removida.
- A nova é inserida ao final da fila.
- Simples, mas pode ser ineficiente, pois uma página que está em uso constante pode ser retirada.
- Pouco utilizado.

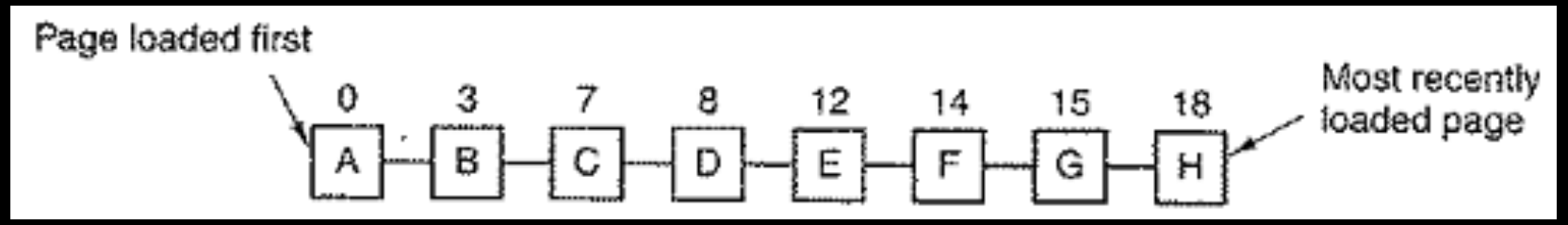
Algoritmo da Segunda Chance

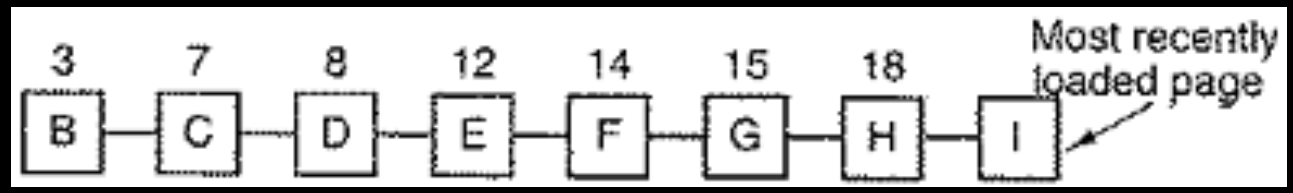
- FIFO + bit R.
- Inspeciona o bit R da página mais velha.
- Se for 0 → é velha e não usada recentemente → é trocada.
- Se for 1, o bit é feito 0.
- A página é colocada no final da fila.
- Seu tempo de carga é modificado, fazendo parecer que recém chegou na memória (recebe uma segunda chance).
- A busca continua.

Algoritmo da Segunda Chance

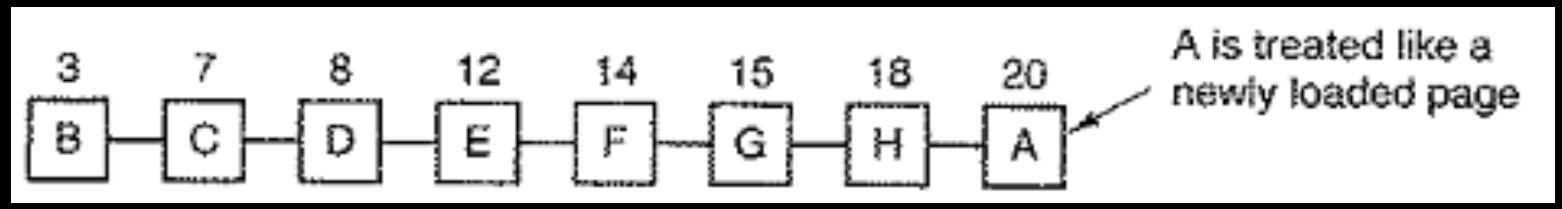
Ocorre page fault no tempo 20 e RA = 0.

A é removido, e o novo elemento é inserido ao final.



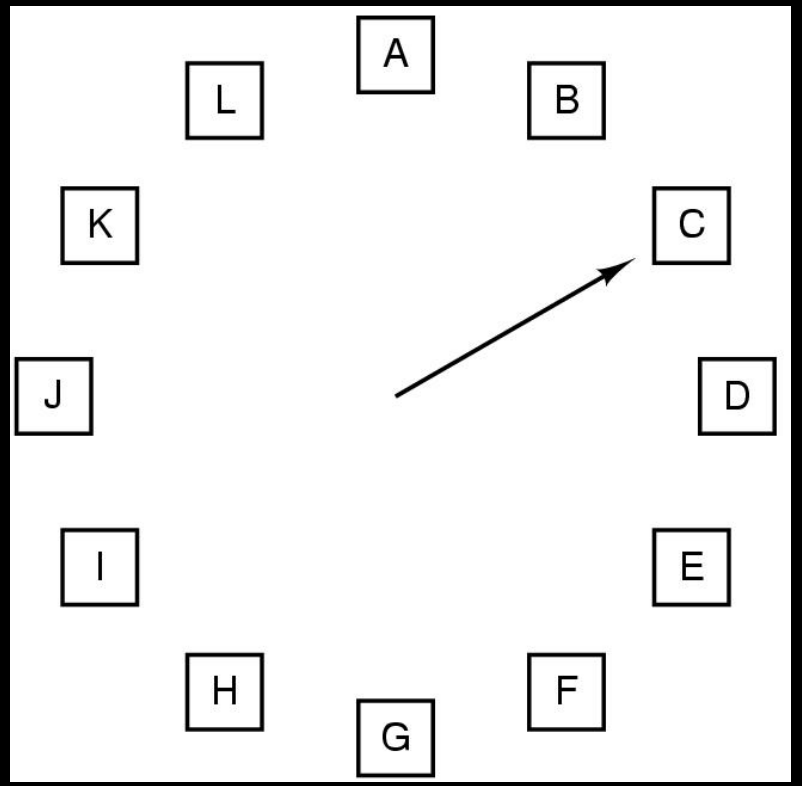


Se RA =1, página é colocada no final



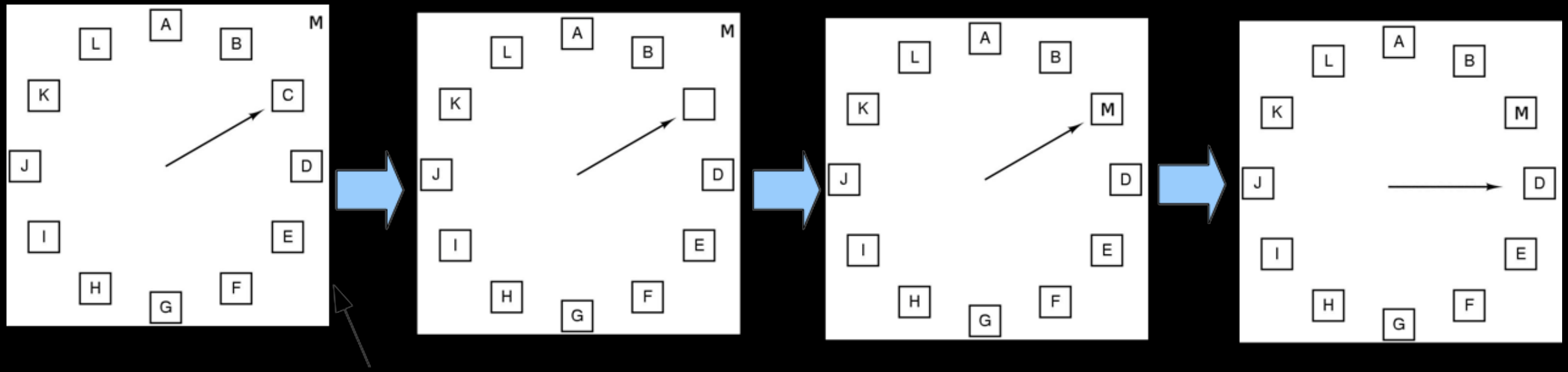
Algoritmo do Relógio

- Melhoria ao Segunda Chance.
  - 2a. Chance move páginas na lista.
- Lista circular com ponteiro apontando para a página mais antiga, na forma de um relógio.
- A cabeça aponta para a página mais antiga.
- Se Rc=0, substitui a página.
- Senão avança para o próximo.



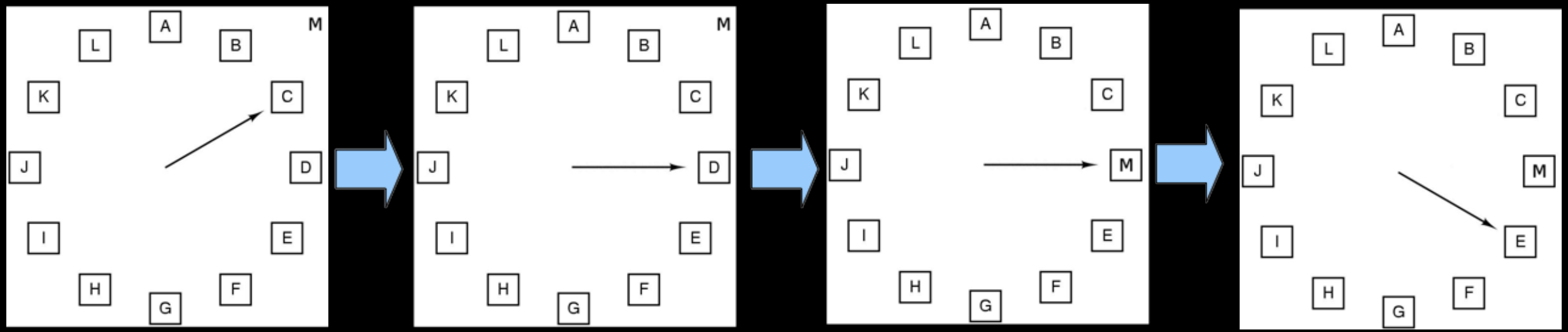
Algoritmo do Relógio

- Se Rc=0, substitui a página.
- Senão avança para o próximo.



Algoritmo do Relógio

- Se Rc=1, avança para o próximo



Algoritmo Least Recently Used Page Replacement (LRU)

- Ideia:
  - Páginas muito usadas ultimamente provavelmente usadas novamente nas próximas.
  - Troca a página que permaneceu em desuso pelo maior tempo.
- Alto custo:
  - Deve-se manter lista encadeada com todas as páginas que estão na memória, com as mais recentemente.
- Implementada em Hardware ou Software

LRU em Hardware

- **MMU** deve suportar a implementação LRU.
- Contador em hardware (64 bits), incrementado automaticamente após cada acesso
- **Tabela de páginas** armazena o valor desse contador – C – em cada entrada
- **Em um page fault, o SO examina todas as entradas na tabela, para encontrar a com menor C**
- Usado no Linux.

LRU em Software – NFU (Not Frequently Used)

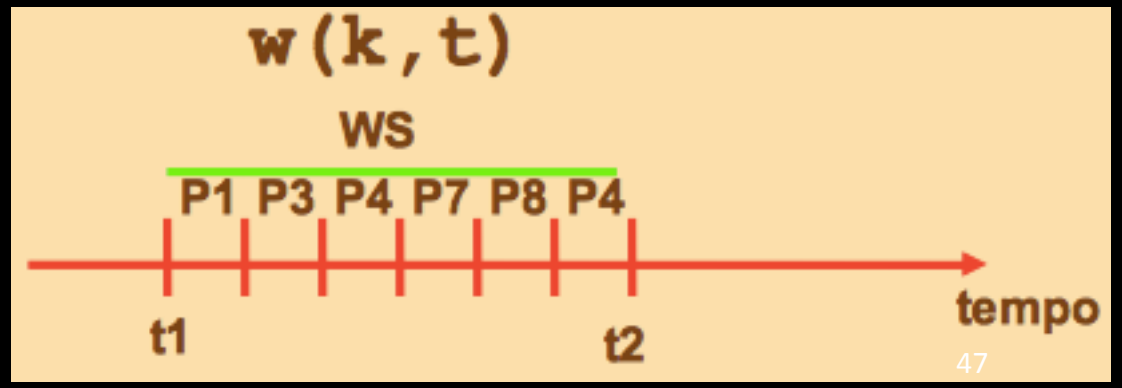
- **Para cada página existe um contador implementado em software**, iniciado com zero.
- Em um page fault, escolhe a página com o menor contador
- Problema: esse algoritmo não se esquece de nada.

- **Páginas frequentemente acessadas muito no início não serão candidatas**.

Algoritmo do *Working Set*

- Conjunto de páginas que um processo está efetivamente utilizando em um determinado tempo t.
- Um processo só é executado quando todas as páginas necessárias no tempo t estão na memória.
  - e.g. quando sofreu interrupção.
- A idéia é determinar o *working set* de cada processo e ter na memória antes de rodar o processo.

- w(k,t) -> conjunto de páginas resultante das últimas k referências à memória no período t.



Algoritmo do Working Set

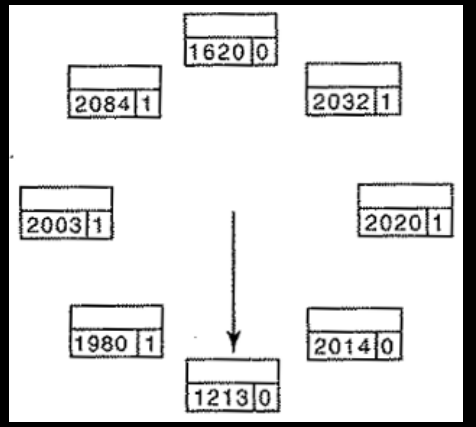
- Podemos estimar o número de páginas necessárias quando o programa é trazido do disco com base em seu *working set* de **quando foi interrompido**.
- **Pré-paginação** consiste em carregar essas páginas antes de rodar novamente o processo.
- O *working set* pode ser visto como o conjunto de páginas que o processo referenciou durante os últimos t segundos de sua execução.
- Utiliza o bit R.
- Evita o thrashing (ultrapaginação).

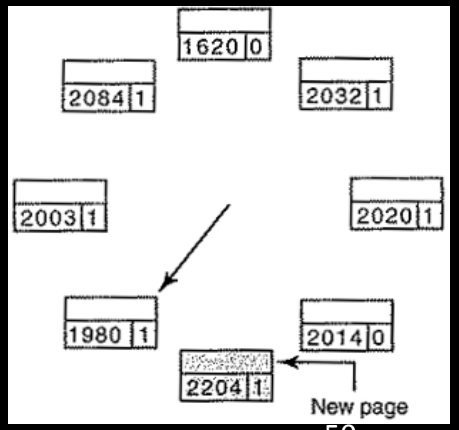
Algoritmo WSClock

- Clock + Working Set.
- Amplamente usado, devido à sua simplicidade e performance.
- Utiliza lista circular de páginas com o tempo do último acesso.
- À medida que mais páginas são carregadas, entram na lista, com as páginas do *working set.*
- Cada entrada contém o tempo de último uso, além dos bits R e M.

Algoritmo WSClock

- A cada page fault, a página da cabeça é examinada primeiro.
- Se R=1, a página foi usada durante o ciclo de clock corrente → não é candidata a remoção.
- Faz R = 0 e avança a cabeça à próxima página, repetindo o algoritmo para esta página.





Concluindo

Foram abordados nesta aula:

- **Memória Virtual, Paginação e Algoritmos de Substituição de Página**

Bibliografia baseada

- **Sistemas Operacionais Modernos; Tanenbaum, A. S. 4ª Edição**

Na próxima aula

- **Introdução à E/S**