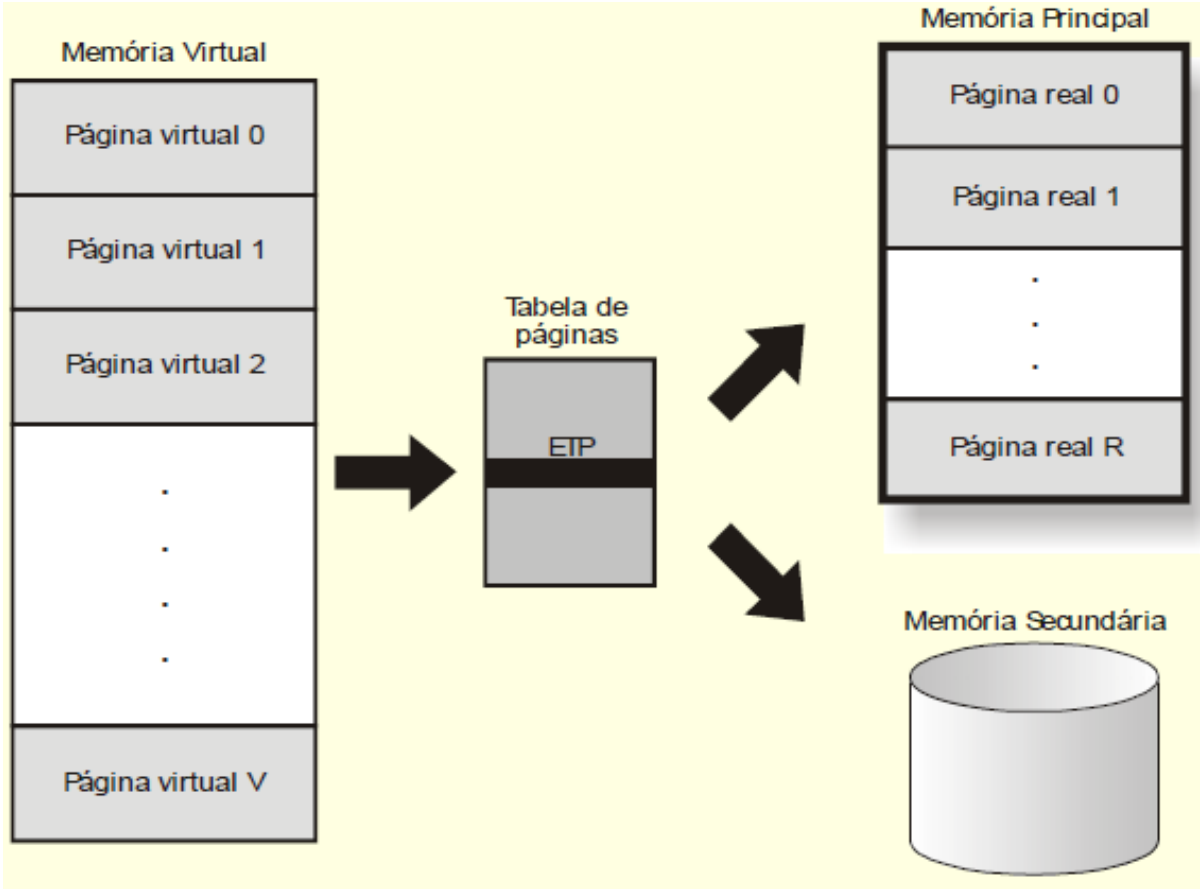
Memória virtual

- implementada pelo SO c/ auxílio da memória secundária
- implementação via **paginação** ou segmentação
- maior que a memória física



Page Fault

- 1ª referência a uma página → trap p/ SO
- SO examina tabela de páginas:
  - referência inválida: aborta
  - página fora de memória
    - obtém frame livre
    - traz página do disco p/ o frame
    - reseta a entrada na tabela (bit de validade = 1)
    - repete instrução que causou o page fault

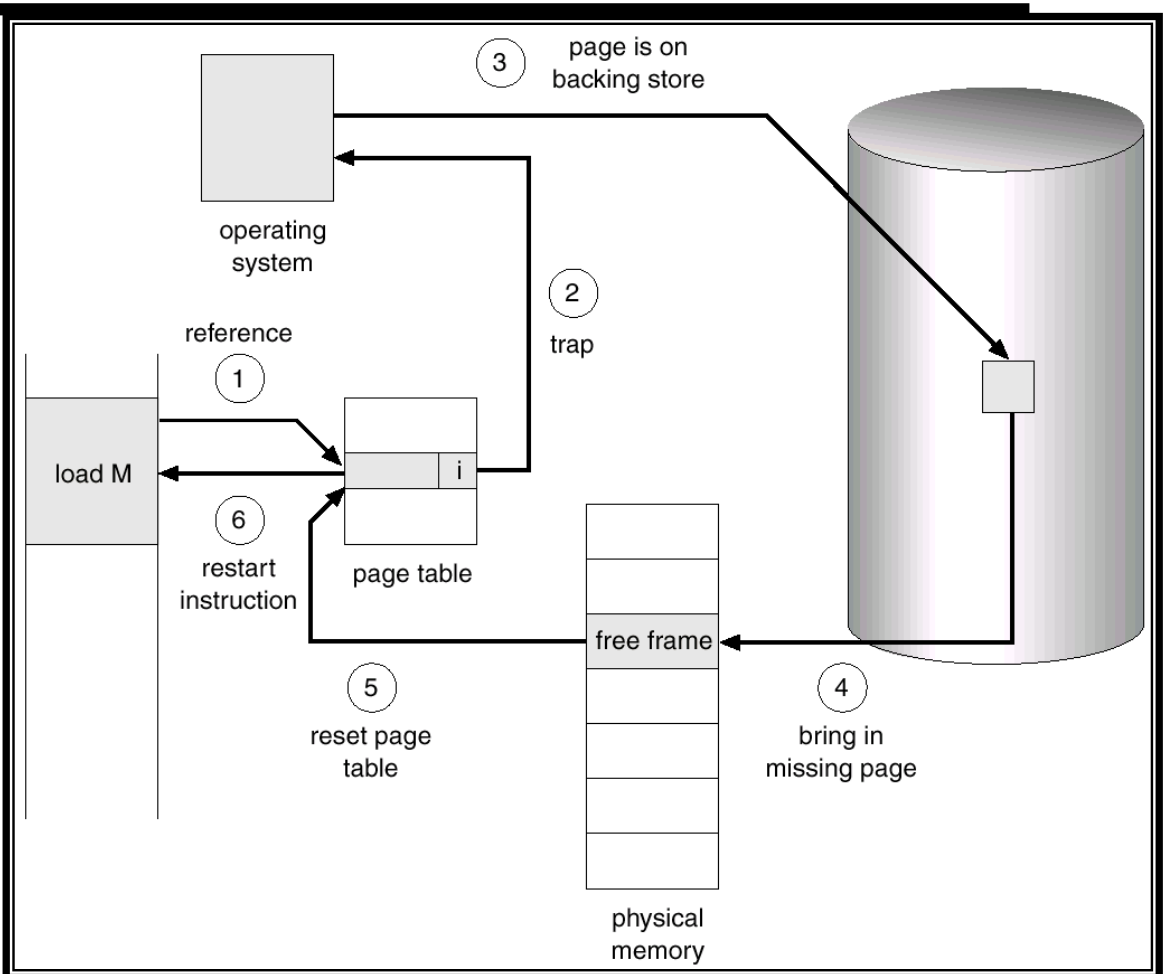
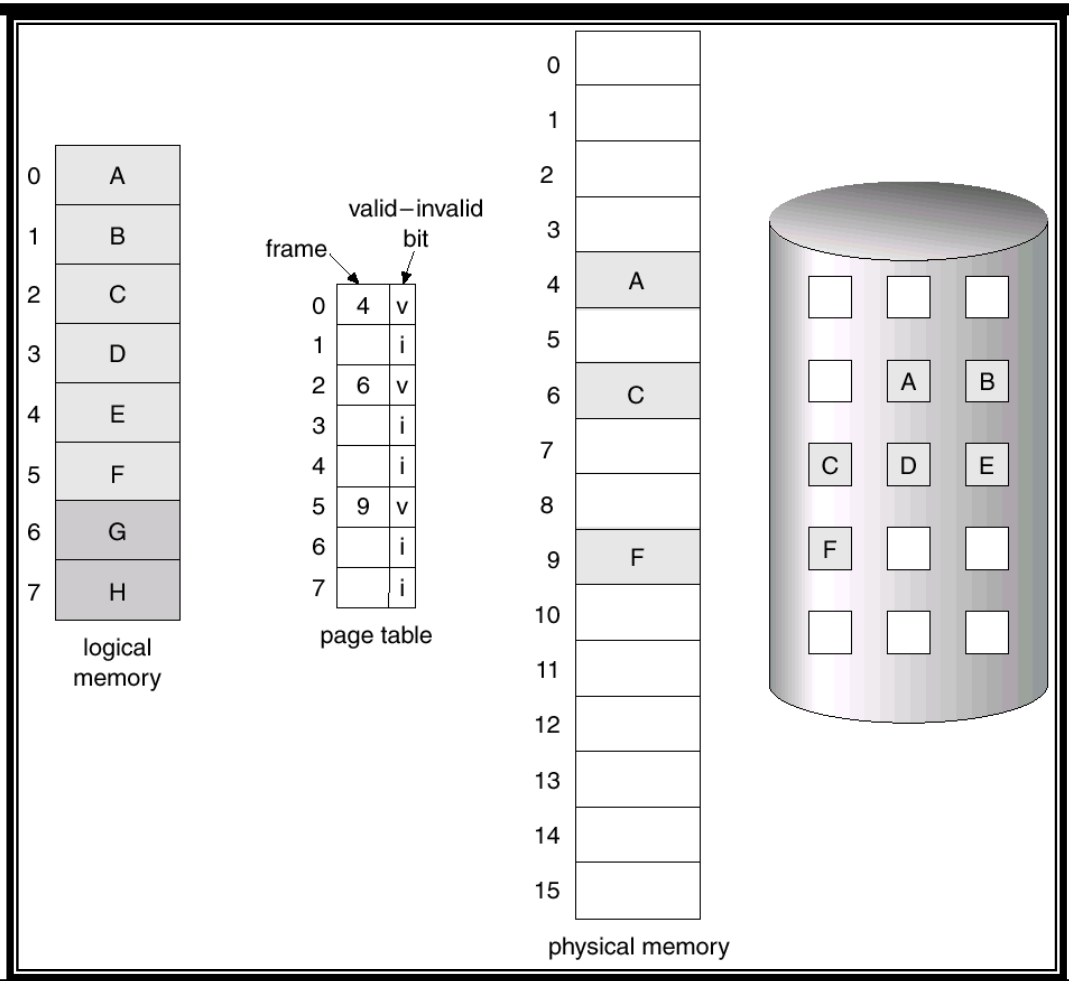


Tabela de páginas com algumas páginas fora da MP

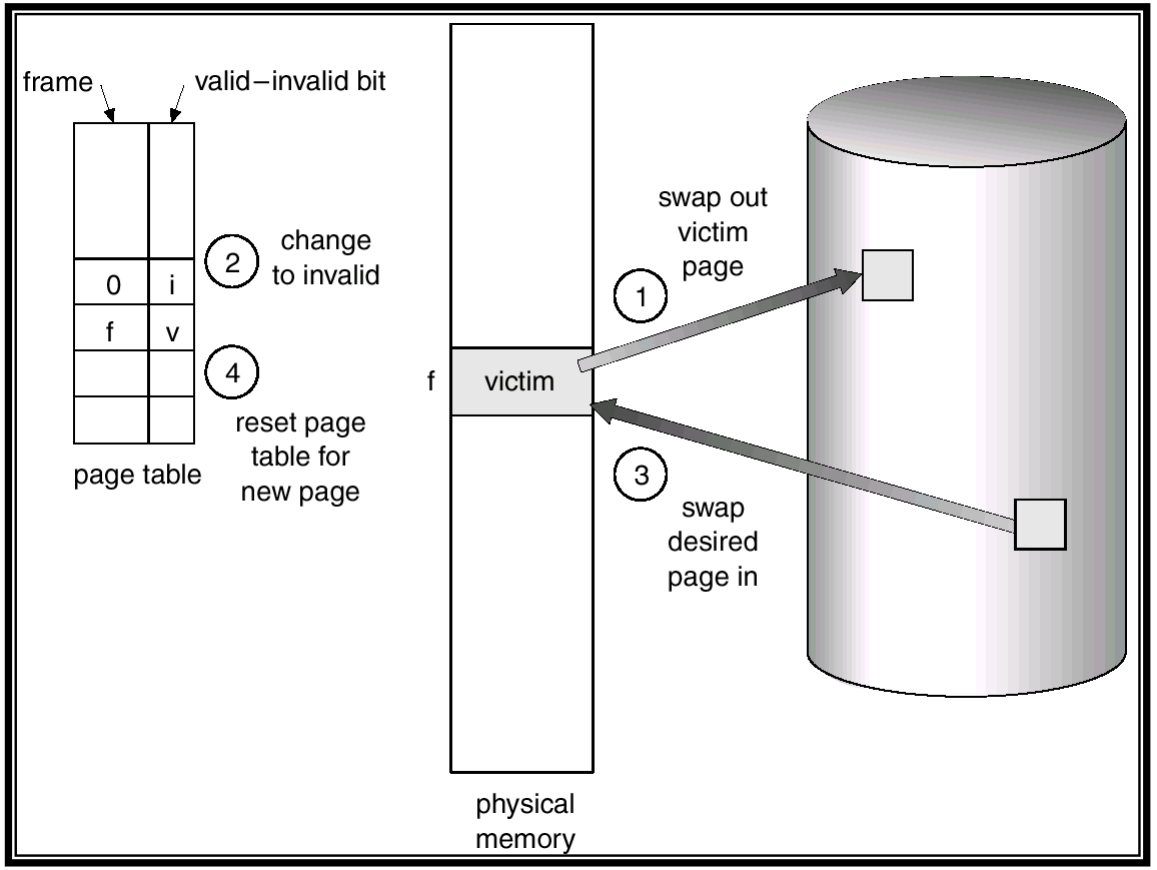


Políticas de busca de páginas

- Paginação por demanda
  - páginas são transferidas da MS p/ MP apenas quando referenciadas
- Paginação antecipada
  - carrega, além da página referenciada, outras páginas que podem ser necessárias ao longo da execução do processo

Substituição de página (1)

- localiza página no disco
- frame livre ?
  - sim → usa
  - não → vítima
- lê a página p/ frame livre, atualiza tabela de páginas e de frames
- reinicia processo



Substituição de página (2)

- *bit modificado* (*dirty bit*)
  - só páginas modificadas são gravadas no disco
  - reduz sobrecarga de transferências de páginas

- vítima
  - Algoritmo de substituição de páginas
  - objetivo: menor taxa de page fault

Algoritmo FIFO (1)

- vítima: página que está a mais tempo na memória
- 3 frames (3 páginas podem estar na memória física ao mesmo tempo)

Páginas	1	2	3	4	1	2	5	1	2	3	4	5
Frame 1	1	1	1	4	4	4	5	5	5	5	5	5
Frame 2		2	2	2	1	1	1	1	1	3	3	3
Frame 3			3	3	3	2	2	2	2	2	4	4
PF ?	x	x	x	x	x	x	x			x	x	

9 PF

Algoritmo FIFO (2)

- supondo 4 frames ...

Páginas	1	2	3	4	1	2	5	1	2	3	4	5
Frame 1	1	1	1	1	1	1	5	5	5	5	4	4
Frame 2		2	2	2	2	2	2	1	1	1	1	5
Frame 3			3	3	3	3	3	3	2	2	2	2
Frame 4				4	4	4	4	4	4	3	3	3
PF ?	x	x	x	x			x	x	x	x	x	x

10 PF

- anomalia de Belady: > frames $\neq$ < PF

Algoritmo Ótimo

- substitui a página que não será usada pelo maior período de tempo

Páginas	1	2	3	4	1	2	5	1	2	3	4	5
Frame 1	1	1	1	1	1	1	1	1	1	1	4	4
Frame 2		2	2	2	2	2	2	2	2	2	2	2
Frame 3			3	3	3	3	3	3	3	3	3	3
Frame 4				4	4	4	5	5	5	5	5	5
PF ?	x	x	x	x			x				x	

6 PF

- implementação ??? → comparações

Algoritmo LRU - Least Recently Used (1)

- substitui a página menos recentemente usada, ou seja, que não foi usada pelo maior período de tempo

Página	1	2	3	4	1	2	5	1	2	3	4	5
Frame 1	1	1	1	1	1	1	1	1	1	1	1	5
Frame 2		2	2	2	2	2	2	2	2	2	2	2
Frame 3			3	3	3	3	5	5	5	5	4	4
Frame 4				4	4	4	4	4	4	3	3	3
PF?	x	x	x	x			x			x	x	x

8 PF

Algoritmo LRU - Least Recently Used: implementações (2)

- com contadores
  - cada entrada da tabela possui um contador
  - a cada referência à página, o valor do clock é copiado p/ o contador
  - quando uma página precisa ser substituída, verifica-se os contadores p/ determinar qual trocar

- lista encadeada
  - páginas mais referenciadas $\rightarrow$ início da lista
  - páginas menos referenciadas $\rightarrow$ final da lista

Algoritmo Bits de referência

- Variação do LRU
- associa um bit a cada entrada na tabela de páginas
  - início: todos os bits em 0
  - a cada acesso: bit é setado (1)
- troca página que tem 0 (se existir)
- sabe-se que páginas foram acessadas, mas não a ordem

Algoritmo LFU (Least Frequently Used)

- considera a frequência de consulta
- troca a página menos referenciada
- implementação: contador (vítima = menor contador)
- páginas bastante acessadas são mantidas na memória
- processos mais recentes são prejudicados

Páginas	1	2	3	1	3	1	3	1	3	4
Frame 1	1	1	1	1	1	1	1	1	1	1
Frame 2		2	2	2	2	2	2	2	2	**4**
Frame 3			3	3	3	3	3	3	3	3
PF ?	x	x	x							x

Contadores
1 – 4
2 – 1
3 - 4

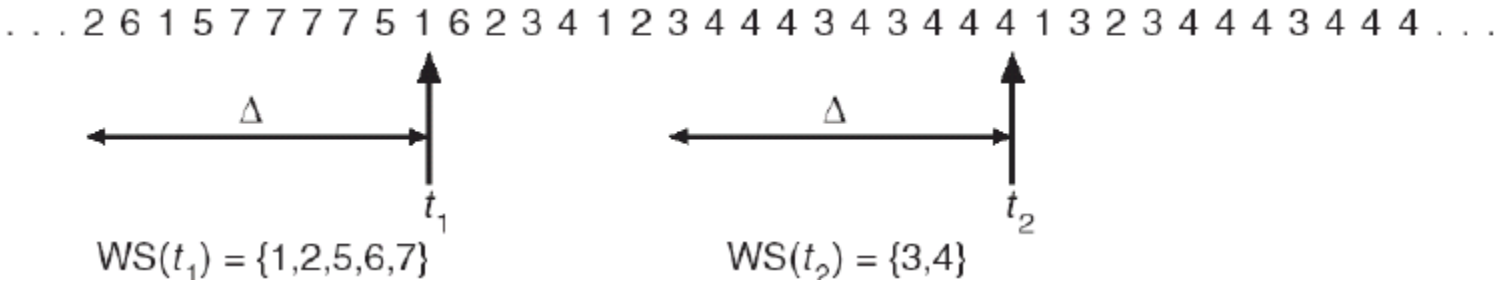
FIFO – substituiria a página 1

Algoritmo NRU - Not Recently Used

- substitui a página não usada recentemente
  - 2 bits: 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
  - periodicamente, R é limpo p/ diferenciar as páginas que não foram referenciadas recentemente
    - a cada *tick*/interrupção do relógio
    - classe 3 → classe 1

Conjunto de Trabalho (Working Set)

- processo executa: migra de localidade em localidade
  - localidade: conjunto de páginas usadas simultaneamente
  - programa = {localidades} que podem se sobrepor
- Conjunto de trabalho (WS) = conjunto de páginas recém referenciadas
  - Base: modelo de localidade
  - $\Delta \rightarrow$ define a janela do conjunto de trabalho
    - Página em uso - está no WS



Políticas de alocação de páginas

- Alocação fixa
  - SO estabelece nº máximo de páginas a serem utilizadas
- Alocação variável
  - SO permite que o nº máximo de páginas de um processo varie (função da taxa de paginação e ocupação na MP)

- Substituição global
  - escolhe páginas de qualquer processo carregado na MP para substituir
- Substituição local
  - escolhe páginas do próprio processo

Exercícios

1) Considere um sistema de memória virtual que implemente paginação, onde o limite de frames por processo é igual a três. Descreva para os itens abaixo, onde é apresentada uma sequência de referências à páginas pelo processo, o número total de page faults para as estratégias de realocação de páginas FIFO e LRU.

   a) 1 / 2 / 3 / 1 / 4 / 2 / 5 / 3 / 4 / 3

   b) 1 / 2 / 3 / 1 / 4 / 1 / 3 / 2 / 3 / 3

2) Considere um processo com limite de páginas reais igual a quatro e um sistema que implemente a política de substituição FIFO. Quantos page faults ocorrerão considerando que as páginas virtuais são referenciadas na seguinte ordem: 0172327103. Repita o problema utilizando a política LRU.