Parte II: Gestão de Memória

e) MV por Paginação a pedido

(conclusão)

Memória Virtual por paginação a pedido

- A página só é carregada para memória quando é referenciada
  - Menos RAM utilizada
  - Tempo de resposta pior (sistema “mais lento”?!)
  - Mais programas em RAM
- Página referenciada por uma instrução
  - Referência válida ⇒ executar a instrução
  - Referência inválida ⇒ abortar a execução
  - ATU indica “Não-em-memória” ⇒ Necessário trazer página para RAM

Memória Virtual: bit de validade

- Cada página tem associado um bit de Validade (V) (V==1 ⇒ a página pertence ao Espaço de Endereçamento do processo, V==0 ⇒ a página não pertence ao EE do processo)
- Na transformação de endereços, se V na entrada da tabela de páginas está a 0 ⇒ segmentation fault (página inválida).
- Permite suportar eficientemente mapas esparsos (“com buracos”)

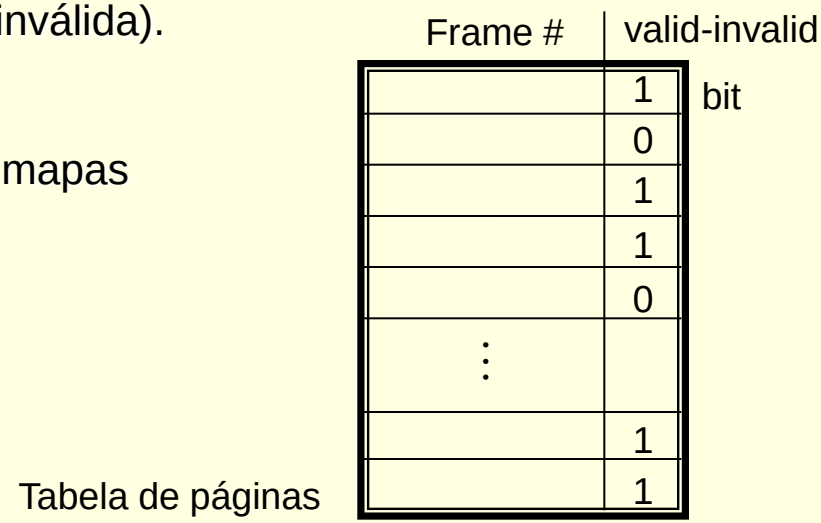


Tabela de páginas

Memória Virtual: bit de presença

- Cada página tem associado um bit de Presença (P) (P==1 ⇒ em RAM, P==0 ⇒ não está na RAM – está em disco ou terá de ser criada – e.g., nova página para um malloc())
- Inicialmente P==0 em todas as páginas.
- Na transformação de endereços, se P na entrada da tabela de páginas está a 0 ⇒ page fault (falta de página).

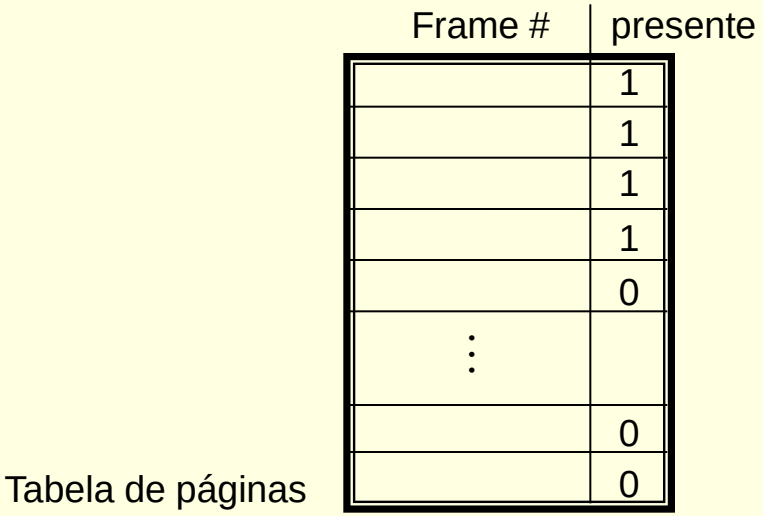
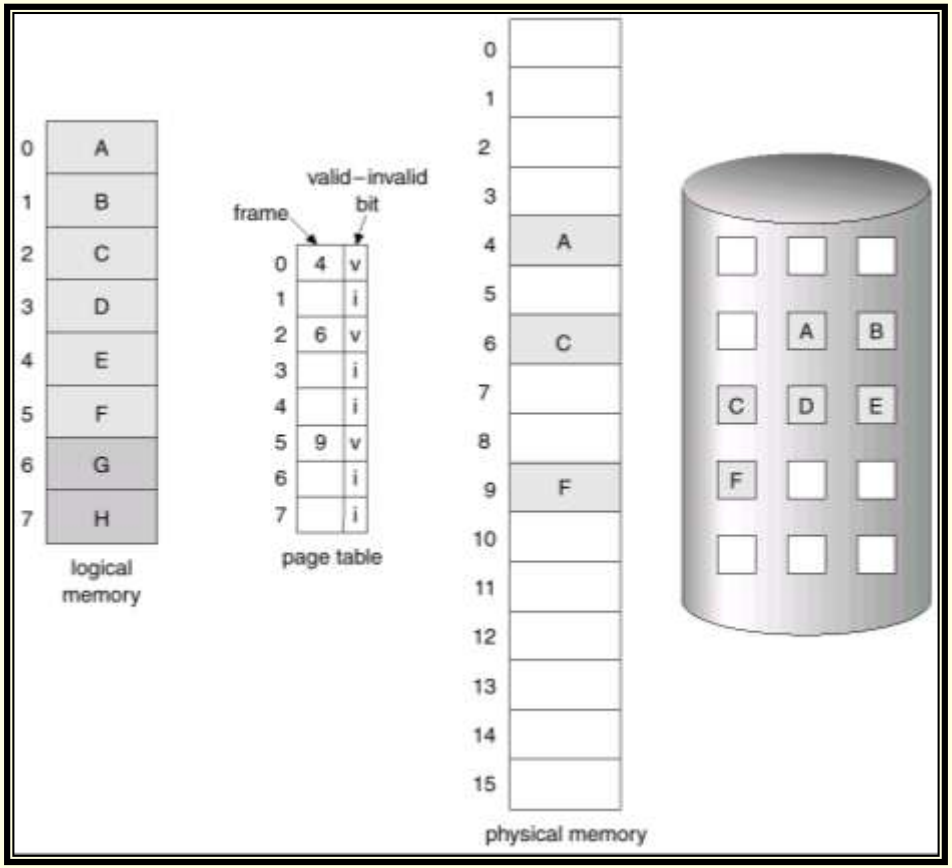


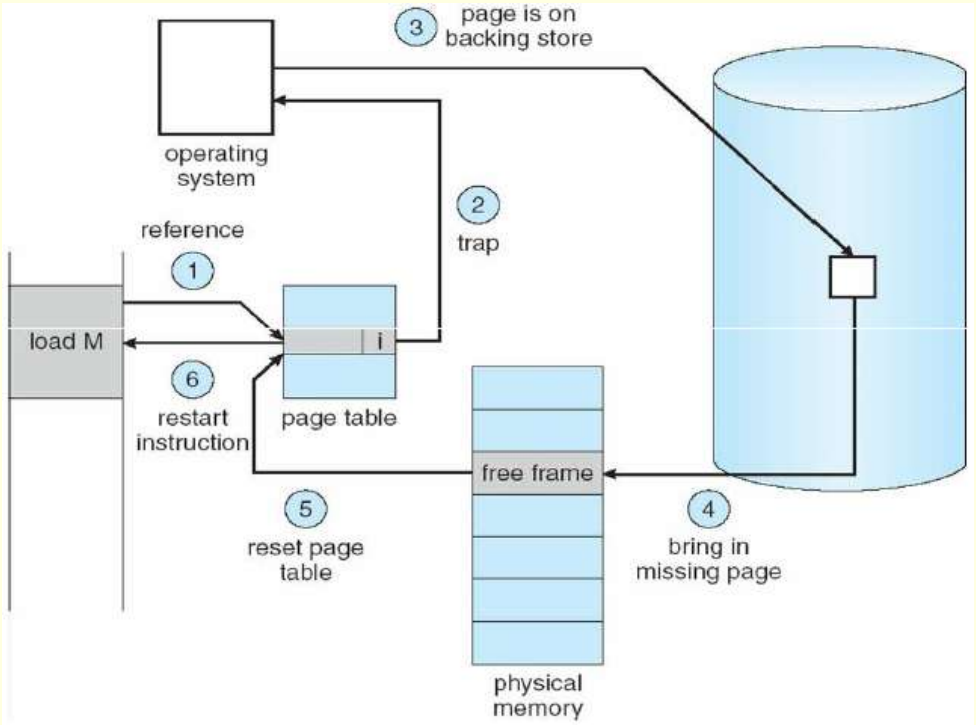
Tabela de páginas

Memória Virtual: quando parte das páginas não está em RAM



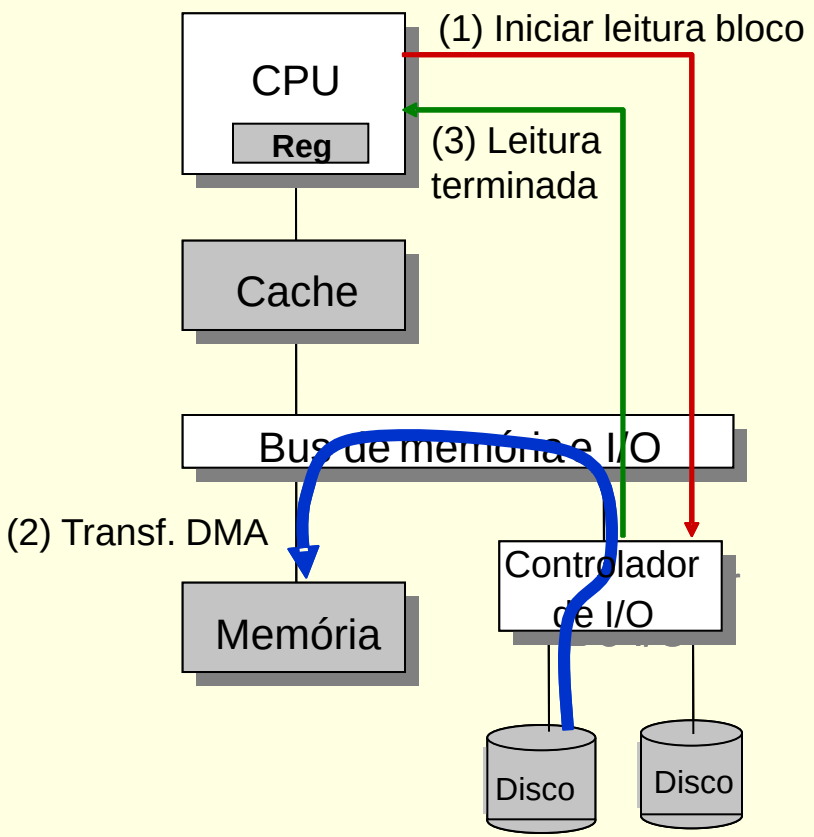
Memória Virtual: falta de página

- Tabela de páginas indica que o conteúdo referenciado pelo endereço virtual não está em RAM
- É invocado um procedimento de excepção para trazer os dados para memória:
  - o processo corrente é suspenso, outros podem continuar
  - o SO tem controlo completo sobre a localização das páginas



Memória Virtual: atendimento da falta de página

- CPU (driver do disco) pede ao controlador:
  - ler bloco de dim. P a partir do end. X e escrever na RAM começando no end. Y (driver adormece)
  - Ocorre a leitura
    - Direct Memory Access (DMA)
    - sob controlo do I/O controller
  - I / O Controller assinala o fim
    - Interrupção, driver acorda
    - SO passa o processo a READY



Memória Virtual: bit de "página modificada"

- Cada página tem associado um bit de Modificada (M), ou *dirty* (suja) (M==1 ⇒ foi modificada desde que veio do disco, M==0 ⇒ versão da página em RAM igual à cópia em disco)
- Inicialmente M==0 em todas as páginas que são carregadas; durante a execução, algumas poderão ser modificadas (variáveis que mudam de valor, stack/heap,...)
- Se fôr preciso libertar uma frame em que M==1, a página tem de ser copiada para disco (para a área de paginação do SO)

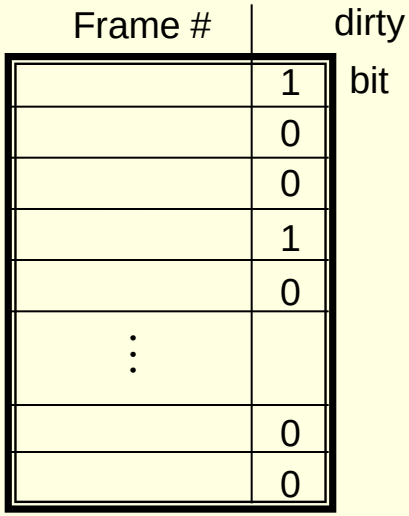


Tabela de páginas

Memória Virtual: tabela "completa"

- Se o SO tiver de "sacrificar" uma página (diz-se vítima) presente em RAM para libertar a frame (porque é necessária para esse mesmo ou para um outro processo), não é muito interessante sacrificar uma que foi modificada, porque esta terá de ser escrita em disco (operação "lenta")
- Bits de protecção
  - Associados a cada página, realizam o controle de acessos e são diferentes para as páginas de:
    - Constantes: r--
    - Variáveis: rw-
    - Heap: rw-
    - Stack: rwx
    - Código: r-x

Tabela de páginas

Frame #	Prm	P	V	M
	r--	1	1	0
	rw-	1	1	0
	---	0	0	0
	rwx	0	1	1
	---	0	0	0
	⋮			
	---	0	0	0
	r-x	1	1	0

Memória Virtual: RAM "cheia" (1)

- Que fazer quando não há nenhuma frame livre?
  - Gestão de Memória: libertar uma frame "à força"
    - Em primeiro lugar, tentar libertar uma frame ocupada por uma página não modificada ➔ não é preciso copiá-la para disco;
    - Se não for possível, escolher uma "vítima", de acordo com uma dada política,
      - Copiá-la para disco, para a área de paging (por razões históricas alguns sistemas, e.g., Unix, designam-na de swapping).
    - Marcar a frame como livre na estrutura de dados de gestão das frames,
    - Marcar a página como "não presente" no mapa de páginas do processo, e associá-la ao seu endereço em disco.

Memória Virtual: RAM "cheia" (2)

- Que política para libertar frames ocupadas?
  - Objectivo: Minimizar o número de faltas de página.
  - Mas como?
    - Com um bom algoritmo de substituição de páginas…
    - FIFO: a vítima é a página mais antiga em memória
      - Mas a página mais antiga pode ser muito acedida! ☹
    - Um algoritmo que escolha a página **menos** acedida!
      - Mas uma página pouco acedida no passado, pode vir a ser muito acedida no futuro... mas não podemos saber o futuro ☺
      - E como saber se uma página é a menos acedida?
        - Associar um timestamp a cada página,
        - Procurar a página de timestamp mais antigo…

Memória Virtual: RAM "cheia" (3)

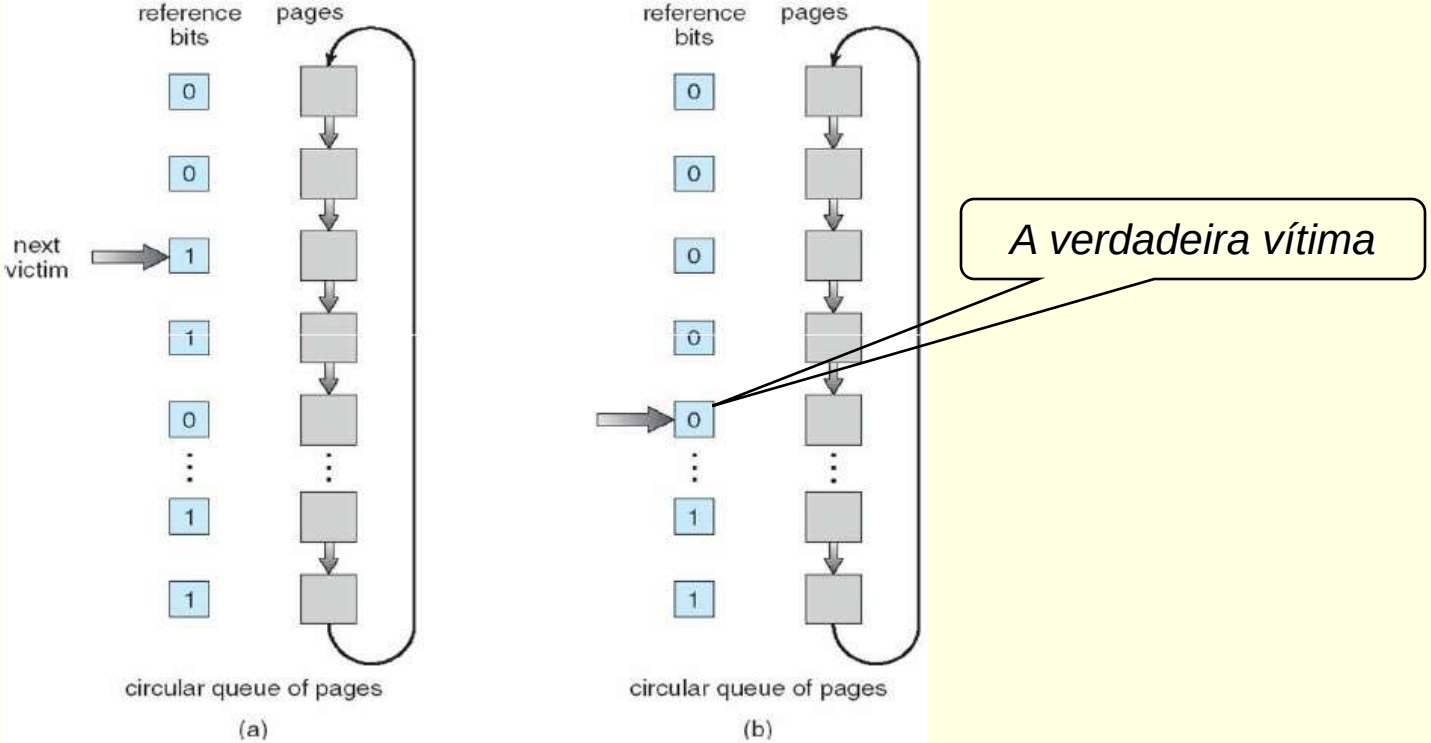
- Podemos realisticamente procurar a página mais antiga?
  - Percorrer 1000000 de timestamps num sistema com 4GB de RAM e páginas de 4KB não é viável, demora muiiiiito!
  - Solução?
- Aproximação: página recentemente menos usada (LRU)
  - Apenas 1 bit, inicializado a zero!
    - Página acedida, bit a 1
    - Escolher uma vítima com o bit a zero
      - E se não houver?
      - É preciso arranjar maneira de pôr os bits de novo a zero, senão não dá ☺

Memória Virtual: RAM "cheia" (4)

- Aproximação LRU: como pôr os bits a zero, "de vez em quando"?
  - Força bruta: de vez em quando, "zerar" tudo
  - Algoritmo da "2ª oportunidade" (second chance)
    - As páginas são tratadas como se pertencessem a uma lista circular (ver próxima pag.)
      - Num dado momento o apontador aponta uma vítima potencial
      - Se a vítima tem o bit a um, ele é posto a zero, deixa-se a página em memória, e considera-se a vítima seguinte…
      - Quando aparecer uma vítima com o bit a zero, é escolhida!

Memória Virtual: RAM "cheia" (5)

- Aproximação LRU "2ª oportunidade": um "apontador" regista a próxima vítima potencial…



Memória Virtual: outras questões…

Unix Windows NT Netware MacOS DOS/VS Vax/VMS Linux Solaris HP/UX AIX Mach Chorus

- Substituições globais vs. locais
  - Locais: vítimas, só do próprio processo
  - Globais: a vítima pode ser doutro processo
- Garantir que um processo tem sempre direito a um número mínimo de páginas em memória
- Numa falta de página, carrega-se sempre a página e umas “quantas” vizinhas – um cluster de páginas (usado no Windows)
- … e muito mais … ☺