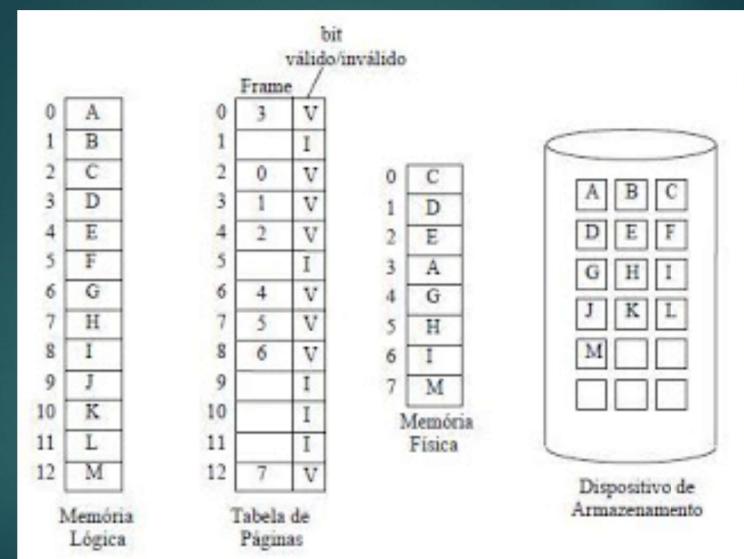


Memória Virtual: Paginação por demanda e Algoritmos de Substituição de páginas

- ▶ Na paginação por demanda apenas as páginas que o processo acessa são carregadas para a memória física. O bit de válido/inválido indica se a página já está presente na memória ou se ainda está no disco.

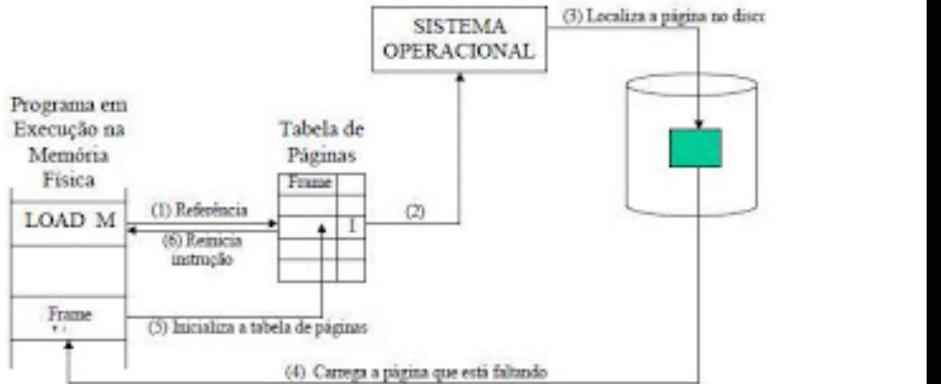
► Paginação Sob Demanda:

- ▶ É baseada na paginação simples:
- ▶ A memória lógica é dividida em páginas que podem ser colocadas em qualquer quadro da memória física;
- ▶ A TP é usada para conversão de endereços lógicos em físicos



- ▶ Na conversão de endereço lógico a MMU testa o bit de válido/inválido e se o bit for válido o endereço é convertido e o acesso é executado normalmente
- ▶ Caso contrário a MMU gera uma interrupção de proteção e aciona o SO
- ▶ O SO verifica se o acesso é a uma inválida (o processo deve ser abortado) ou a uma página que está no disco
- ▶ Se o acesso é a uma página que está no disco (falta de página ou page fault), o SO:
 - ▶ Coloca o processo numa fila de processos esperando por página
 - ▶ Localiza um quadro livre
 - ▶ Localiza a página no disco
 - ▶ Agenda uma operação de leitura da página no disco

- ▶ Enquanto um processo espera por uma página, a UCP é passada para outro processo
- ▶ Quando a leitura da página for concluída, o SO:
 - ▶ Atualiza a tabela de páginas do processo
 - ▶ Recoloca o processo na fila de prontos
 - ▶ Quando o processo reassumir a UCP ele deverá reiniciar a execução que gerou a falta de página
 - ▶ Na paginação por demanda pura, uma página só é carregada para a memória quando é referenciada
 - ▶ Aparte do SO que carrega as páginas do disco para a memória é chamada de pager



► Os algoritmos de substituição de páginas são políticas definidas para escolher qual(is) página(s) da memória dará lugar a página que foi solicitada e que precisa ser carregada. Isso é necessário quando não há espaço disponível para armazenar a nova página. Devemos ressaltar que se a página a ser removida sofreu alterações enquanto esteve na memória, a cópia virtual existente em disco deverá ser atualizada.

► Por outro lado, se a página não foi modificada significa que sua cópia está atualizada e, portanto, não é necessário reescrevê-la. Políticas de substituição de páginas devem ser utilizadas em sistemas que fazem uso de memória virtual paginada com o objetivo de melhorar o desempenho do sistema computacional.

► Os algoritmos de substituição de páginas podem ser classificados, basicamente, em: algoritmos com espaço fixo e algoritmos com espaço variável. A diferença entre estes dois tipos de algoritmos é que o de espaço fixo trabalha sobre uma área de memória sempre constante, enquanto que os de espaço variável podem modificar o tamanho da memória alocada dinamicamente.

Algoritmo de substituição de páginas FIFO

- FIFO (First-in, First-out) é um algoritmo de substituição de páginas de baixo custo e de fácil implementação que consiste em substituir a página que foi carregada há mais tempo na memória (a primeira página a entrar é a primeira a sair). Esta escolha não leva em consideração se a página está sendo muito utilizada ou não, o que não é muito adequado pois pode prejudicar o desempenho do sistema. Por este motivo, o FIFO apresenta uma deficiência denominada anomalia de Belady: a quantidade de falta de páginas pode aumentar quando o tamanho da memória também aumenta.

Algoritmo de substituição de páginas FIFO

- Por estas razões, o algoritmo FIFO puro é muito pouco utilizado. Contudo, sua principal vantagem é a facilidade de implementação: uma lista de páginas ordenada pela “idade”. Dessa forma, na ocorrência de uma falta de página a primeira página da lista será substituída e a nova será acrescentada ao final da lista.

Algoritmo de substituição de páginas LRU

- O LRU (Least Recently Used) é um algoritmo de substituição de página que apresenta um bom desempenho substituindo a página menos recentemente usada. Esta política foi definida baseada na seguinte observação: se a página está sendo intensamente referenciada pelas instruções é muito provável que ela seja novamente referenciada pelas instruções seguintes e, de modo oposto, aquelas que não foram acessadas nas últimas instruções também é provável que não sejam acessadas nas próximas.

Algoritmo de substituição de páginas LRU

- Apesar de o LRU apresentar um bom desempenho ele também possui algumas deficiências quando o padrão de acesso é sequencial (em estruturas de dados do tipo vetor, lista, árvore), dentro de loops, etc. Diante dessas deficiências foram propostas algumas variações do LRU, dentre eles destacamos o LRU-K. Este algoritmo não substitui aquela que foi referenciada há mais tempo e sim quando ocorreu seu k-último acesso. Por exemplo, LRU-2 substituirá a página que teve seu penúltimo acesso feito há mais tempo e LRU-3 observará o antepenúltimo e assim por diante.

Algoritmo de substituição de páginas LRU

- A implementação do LRU também pode ser feita através de uma lista, mantendo as páginas mais referenciadas no início (cabeça) e a menos referenciadas no final da lista. Portanto, ao substituir retira-se a página que está no final da lista. O maior problema com esta organização é que a lista deve ser atualizada a cada nova referência efetuada sobre as páginas, o que torna alto o custo dessa manutenção.

Algoritmo de substituição de páginas Ótimo

- O algoritmo ótimo, proposto por Belady em 1966, é o que apresenta o melhor desempenho computacional e o que minimiza o número de faltas de páginas. No entanto, sua implementação é praticamente impossível. A idéia do algoritmo é retirar da memória a página que vai demorar mais tempo para ser referenciada novamente.

Algoritmo de substituição de páginas Ótimo

- Para isso, o algoritmo precisaria saber, antecipadamente, todos os acessos à memória realizados pela aplicação, o que é impossível em um caso real. Por estes motivos, o algoritmo ótimo só é utilizado em simulações para se estabelecer o valor ótimo e analisar a eficiência de outras propostas elaboradas.