Memória Secundária

Helton Hideraldo Bíscaro

Apresentação

Introdução à estrutura de arquivos

Perspectivas do processamento de dados

- Armazenamento
- Organização
- Acesso
- Processamento

Estrutura de dados X estrutura de arquivos

Ambos envolvem:

- representação dos dados + operações para acesso aos dados

Diferenças:

- Estrutura de dados lida com dados na memória principal
- Estrutura de arquivos lida com dados na memória secundária

Memória Secundária

Arquitetura do Computador

Tempo para obter uma informação: Memória principal X memória secundária

- Memória principal: 60 nanosegundos $= 60 \times 10^{-9}$ segundos
- Memória secundária: 15 milisegundos $= 15 \times 10^{-3}$ segundos

Tipos de operações em qualquer tipo de memória

- leitura $\longrightarrow$ recuperação da informação armazenada
- escrita $\longrightarrow$ gravação (ou armazenamento) da informação na memória

Memórias

memória principal (RAM)

- volátil: informação é perdida quando desligado
- Pequena (alto custo)
- Acesso rápido

Memória secundária

- não volátil
- tem maior capacidade de armazenamento
- é mais barata
- Acesso lento (comparado ao da RAM)

Exemplos de memórias secundárias

- Discos rígidos e flexíveis
- Fitas Magnéticas
- Cd-rom (compact disk read only memory), DVD
- Zip disks, Pen drives.

Memórias

Objetivos da estrutura de arquivos

- Minimizar o número de acessos ao disco para obter a informação desejada
- Agrupar informações relacionadas que geralmente são solicitadas ao mesmo tempo

Memórias

Objetivos da estrutura de arquivos

- Minimizar o número de acessos ao disco para obter a informação desejada
- Agrupar informações relacionadas que geralmente são solicitadas ao mesmo tempo

Objetivos da estrutura de arquivos

O ideal é obter a informação com apenas 1 acesso a disco.
Se o ideal não pode ser atingido $\rightarrow$ menor no. de acessos possível
Exemplo:

- O método de busca binária permite que um registro pesquisado entre 50.000 seja encontrado em no máximo 16 comparações ($\log_2 50.000 = 15.610$)
- mas acessar o disco 16 vezes para buscar uma informação é tempo demais
- Precisamos de estruturas que permitam recuperar esse mesmo registro em dois ou três acessos!

Objetivos da estrutura de arquivos

O ideal é obter a informação com apenas 1 acesso a disco.

- O queremos estruturas que agrupem informações de modo a permitir que toda (ou quase toda) a informação necessária seja obtida em uma única operação de acesso a disco.
- É difícil obter uma estrutura de arquivos estável $\longrightarrow$ dados mudam aumentam e diminuem a medida em que informações são alteradas, adicionadas ou removidas.

Estrutura de arquivos

A evolução do armazenamento e acesso.

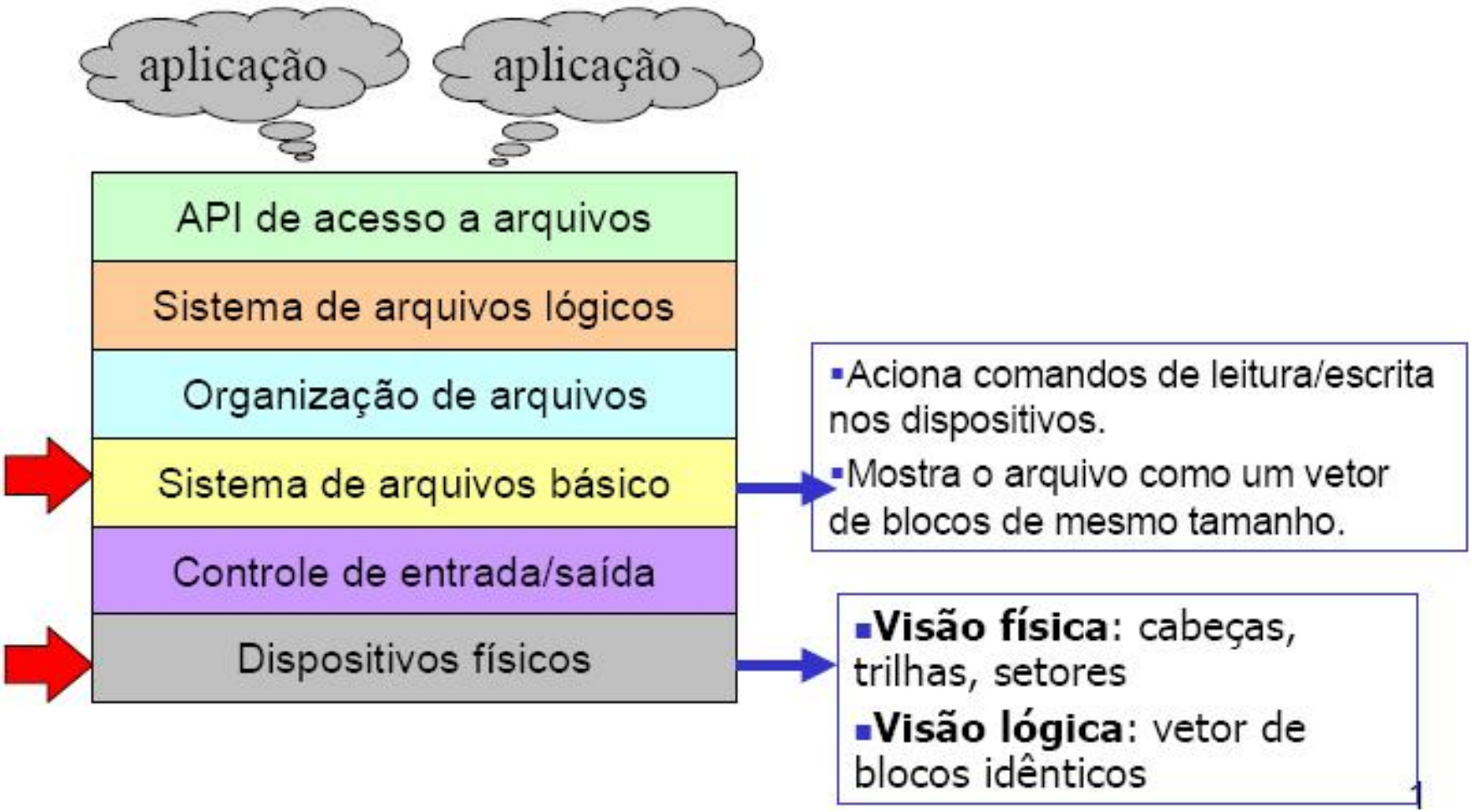
- Inicialmente: dados em fitas $\longrightarrow$ o acesso era seqüencial, e o custo do acesso era diretamente proporcional ao tamanho do arquivo.
- rapidamente os arquivos cresceram de modo inaceitável para o processamento através de acesso seqüencial
- Dispositivos como disco e tambores apareceram, o que permitiu a associação de índices aos arquivos.
- Índices
  - permitem guardar uma lista de chaves/ponteiros em um arquivo maior, que pode ser pesquisado mais rapidamente.
  - Dada a chave associada ao registro procurado, a busca no índice retorna um ponteiro para o registro no arquivo principal, e a informação pode, então, ser obtida diretamente

Estrutura de arquivos

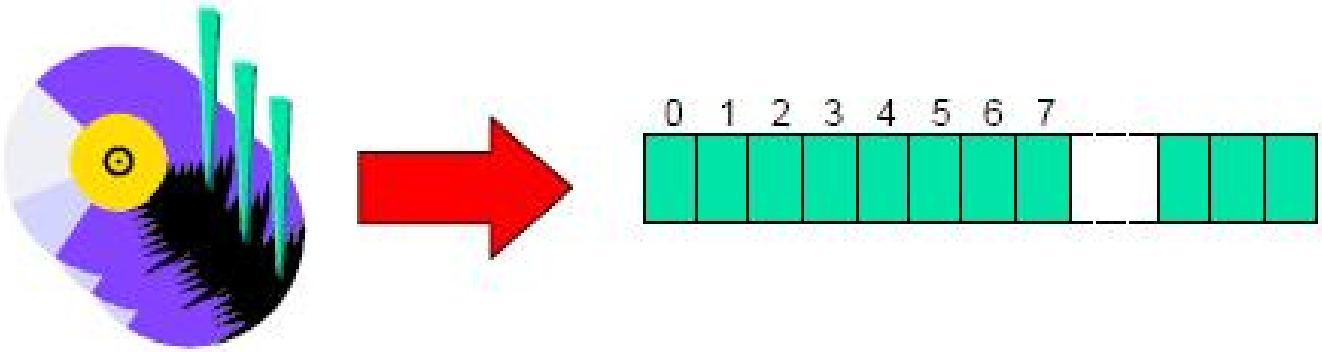
Estruturação dos índices.

- usar **árvores** como solução para estruturar os índices. O problema é que as árvores tendem a crescer de maneira desigual a medida que as chaves são inseridas e removidas (ficam desbalanceadas).
- usar as **árvores AVL**. Mesmo árvores perfeitamente balanceadas exigem muitos acessos para localizar um dado registro.
- usar **árvores-B**. (B-trees) permitem um excelente desempenho na busca por registros do arquivo, ao custo de não mais se poder acessar seqüencialmente o arquivo de modo eficiente, mas garantem o acesso a um registro mantido em um arquivo com milhões de entradas com apenas três ou quatro acessos ao disco
- usar **árvore-B+**. Árvore-B + uma lista encadeada no nível inferior da ávore-B
- Usar **Hashing** (espalhamento). É uma boa maneira de se fazer apenas 1 acesso, se estivermos trabalhando com arquivos que não mudam de tamanho (acesso eficiente a arquivos estáticos).
- Usar **Hashing dinâmico**, que pode ser aplicado a arquivos dinâmicos eficientemente.

Arquitetura da gerência de arquivos



Arquitetura da gerência de arquivos



Memória secundária

Discos

- A informação é gravada em uma ou mais superfícies, e armazenada em trilhas na superfície.
- Cada trilha é normalmente dividida em setores.
- Um setor é a menor porção endereçável do disco. Quando um READ precisa de um byte em particular, o S.O busca a superfície, trilha e setores corretos, e encontra o byte necessário.

Cilindro

- Disco é formado por diversos pratos.
- Vantagem: toda a informação contida num cilindro pode ser obtida sem movimentação da cabeça de R/W.
- Essa movimentação, chamada seeking, é normalmente a parte mais lenta da operação de acesso.

Memória secundária

Memória secundária

Estimativa de capacidade/espaço necessários

- Discos variam em tamanho entre 2 e 14 polegadas, e em capacidade entre 400KB a bilhões de bytes.
- Normalmente as 2 superfícies de um prato são utilizadas, exceto no caso dos pratos superior e inferior.
- A quantidade de dados armazenados em uma trilha depende de quão densamente os bits podem ser armazenados [qualidade do meio e tamanho da cabeça R/W].
  - capacidade da trilha = nro. setores/trilha x nro bytes/setor
  - capacidade do cilindro = nro. trilhas/cilindro x capacidade da trilha
  - capacidade do drive = nro. cilindros x capacidade do cilindro

Memória secundária

Exemplo: Queremos armazenar 20.000 registros de tamanho fixo num disco de 300 Mbytes que contém:

- 512 bytes/setor
- 40 setores/trilha
- 11 trilhas/cilindro
- 1331 cilindros

Quantos cilindros são necessários, se cada registro tem 256 bytes ?

- São 2 registros/setor $\rightarrow$ serão necessários 10.000 setores.
- Em cada cilindro há 40 x 11 = 440 setores.
- O número de cilindros é aproximadamente 10.000/440 = 22.7 cilindros.

É provável que um drive tenha essa quantidade de cilindros disponível, mas não de forma contígua, e o arquivo pode ser espalhando em dezenas, ou centenas, de cilindros.

Memória secundária

Visões da organização de trilhas em setores

- visão lógica: Os setores são adjacentes.
- Fisicamente: Mão é uma boa forma de organização.
  - após ler os dados de um setor, pode ser necessário processar os dados.
  - Se dois setores logicamente adjacentes também estão fisicamente adjacentes, pode-se perder a leitura do próximo setor (pois o disco continua em movimento de rotação), e teria que esperar uma outra rotação inteira do disco para acessar o setor perdido.

Técnica: intercalação(interleaving) vários setores físicos são colocados entre setores logicamente consecutivos.

Memória secundária

Clustres

- Um cluster consiste de vários setores (logicamente) contíguos (a contigüidade física depende do fator de intercalação).
- Quando um programa acessa um arquivo, o gerenciador de arquivos deve associar o arquivo lógico às suas posições físicas.
- Para isso, o arquivo é visto como uma série de clusters de setores.
- Considerando que um cluster associado a um arquivo foi encontrado, todos os seu setores podem ser acessados sem necessidade de um seeking adicional.
    - Essa organização por setores/clusters é gerenciada através de uma File Alocation Table (FAT).
    - Nessa tabela, cada entrada dá a localização física do cluster associado a um certo arquivo lógico.
    - Em muitos sistemas,o administrador pode decidir o número de setores/cluster.
    - A vantagem de se ter clusters com um maior número de setores é a possibilidade de ler mais setores sem precisar de operações de seeking adicionais.

Memória secundária

Extents

- Um extend é um grupo de clusters consecutivos
- Se um arquivo é armazenado em um único extend pode ser buscado com um único seeking
- Quando novos clusters são adicionados, o gerenciador de arquivos tenta manter o arquivo ocupando um único extent
- Se não for possível, novos extents são adicionados, e o arquivo deixa de ser formado por um só extent
- Quanto maior o número de extents, mais espalhado está o arquivo pelo disco, e maior é a quantidade de seekings necessários para processá-lo no caso de acesso seqüencial ao mesmo.

Memória secundária

Fragmentação

- Em geral, todos os setores de um drive possuem um mesmo tamanho, isto é, podem armazenar o mesmo número de bytes.
- O que ocorre se o setor tem 512 bytes e os registros de um registro têm 300 bytes? Existem 2 alternativas possíveis:
  1. armazenar um registro por setor. Neste caso, a recuperação de um registro exige a busca de apenas um setor, mas uma área não utilizada é mantida em cada setor. Essa perda de espaço é denominada fragmentação interna.
  2. permitir que os registros sejam quebrados em diversos setores. Neste caso, não se perde espaço com fragmentação interna, mas podem existir registros cuja recuperação exige a busca de 2 setores.

Clusters e fragmentação interna

- Um cluster é a menor unidade de espaço que pode ser alocada a um arquivo. Quando ocorre fragmentação nos clusters ?

Exemplo: Considere clusters com 3 setores de 512 bytes cada, e um arquivo com 1 byte. Quanto de espaço, em bytes, é perdido?

Memória secundária

Organização de discos em blocos definidos pelo usuário

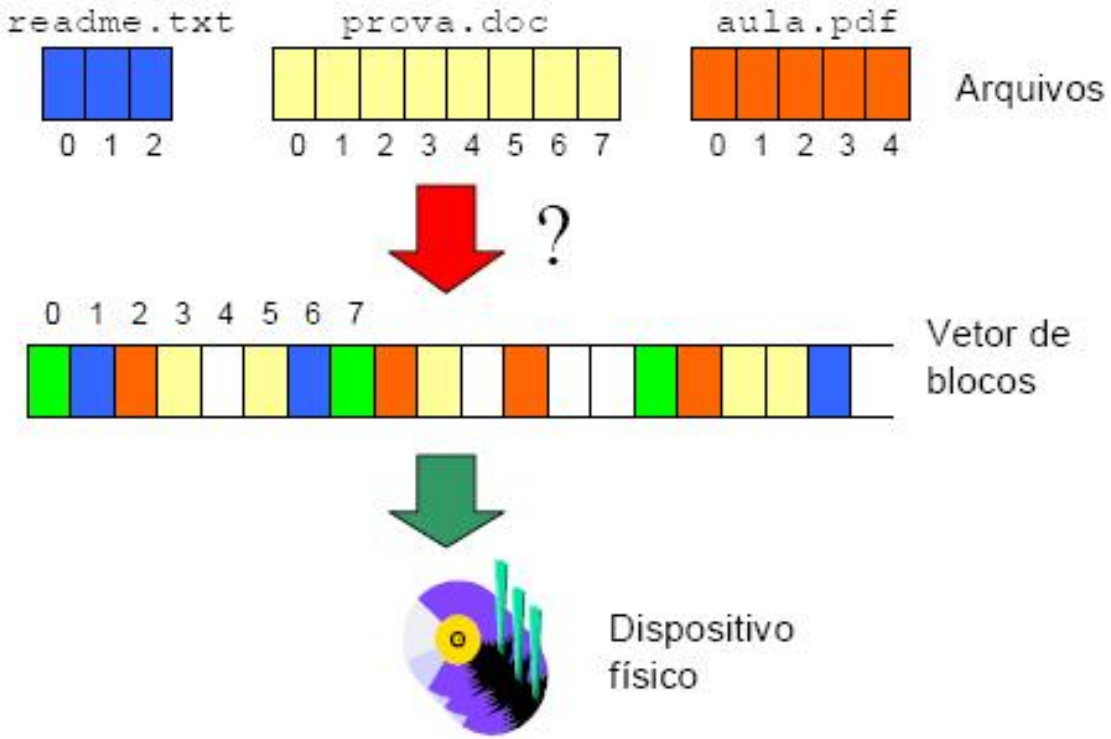
- Em alguns sistemas, pode-se organizar as trilhas do disco não por setores, mas por blocos de tamanho variável, definido pelo usuário.
- Neste caso, a quantidade de dados transferidos em uma operação de leitura/escrita é dada pelo tamanho do bloco.

Fator de bloco: número de registros por bloco em um arquivo.

- se tivermos registros de 300 bytes, podemos definir blocos de 300 bytes (ou um valor múltiplo de 300). Assim:
- não existe mais o problema de fragmentação interna.
- não teremos mais "quebra"de registros entre setores (uso de mais de um setor por registro).

Memória secundária

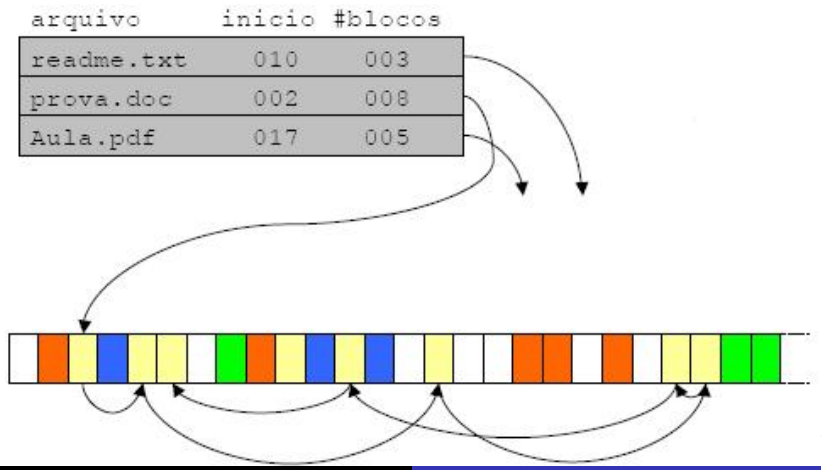
Organização de discos em blocos



Memória secundária

Organização de discos em blocos: Alocação Encadeada

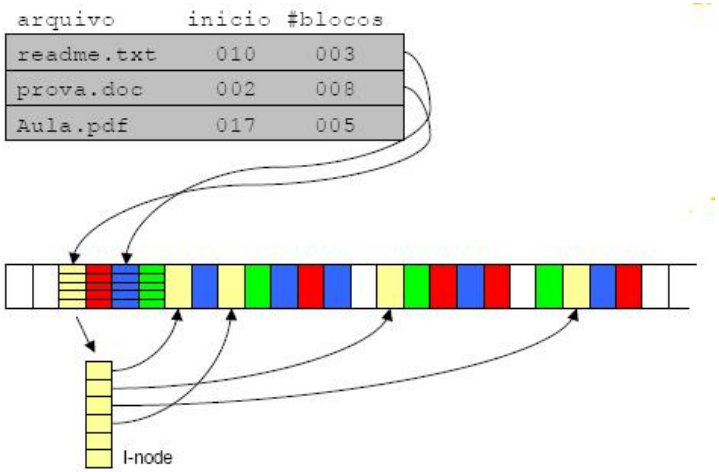
- Os arquivos são armazenados como listas de blocos
  - cada bloco aponta para o próximo
  - diretório aponta para o bloco inicial
  - os blocos podem estar espalhados
  - Base de funcionamento da FAT
- Sistema de arquivos Windows



Memória secundária

Organização de discos em blocos: Alocação Indexada

- Baseada em tabelas de blocos
  - um bloco especial guarda a tabela de blocos do arquivo: index-node (i-node)
  - diretório aponta para os i-nodes
  - blocos podem estar espalhados
- Sistema de arquivos Linux



Memória secundária

Organização de discos - Overhead (espaço sem dados)

- O processo de formatação provoca a inclusão de informações extras.
- discos organizados por setor:
  - marcas de início/fim de setor, marcas de sincronização, se o setor é válido ou danificado, entre outras, são inseridas no processo de formatação. Esse overhead é invisível para o usuário.
- discos organizados por bloco:
  - também contém overhead (de outro tipo), alguns dos quais o programador precisa conhecer: marcas de sub-blocos e inter-blocos devem ser adicionadas em cada bloco. Deste modo, um número maior de informação que não é dado pode ser colocada no disco, quando comparado ao esquema de organização por setores.
  - vantagem: permite ao programador escolher a organização dos dados, de forma otimizar o acesso.

Memória secundária

Organização de discos Exemplo:

- Seja um disco organizado em blocos com 20.000 bytes/trilha no qual o overhead/bloco é 300 bytes. Queremos armazenar um arquivo contendo registros de 100 bytes. **Quantos registros podem ser armazenados, se o fator de bloqueamento é 10 ?**

  Se 10 registros de 100 bytes são armazenados por bloco, cada bloco terá 1.000 bytes de dados, mas usará 1.300 bytes no total, devido ao overhead. Então, o número de blocos que podem ser colocados em uma trilha é dado por: $20.000/1.300 = 15.38 = 15$, portanto 15 blocos, ou 150 registros.
  Quantos registros podem ser armazenados, se o fator de bloqueamento é 60 ?

Memória secundária

Organização de discos Exemplo:

- Blocos maiores, em geral, levam a um uso mais eficiente do espaço, pois há um número menor de bytes de overhead, em comparação com o número de bytes ocupados por dados.
- Mas nem sempre os blocos maiores garantem ganhos: a fragmentação interna dentro da trilha ocorre, pois as trilhas têm tamanho fixo e só podemos por um número inteiro de blocos dentro de uma trilha, e sempre sobra espaço no final.
- O que ocorreria com fator de bloqueamento 97 ?
- O que ocorreria com fator de bloqueamento 98 ?

Memória secundária

Organização de discos em blocos

- A escolha do tamanho dos blocos é importante para a eficiência do sistema.
- Blocos pequenos:
  - menor perda por fragmentação interna
  - mais blocos por arquivo: maior custo de gerência
- Blocos grandes:
  - maior perda por fragmentação interna
  - menos blocos por arquivo: menor custo de gerência

Memória secundária

Bibliografia

- CLAYBROOK, Billy. Técnicas de gerenciamento de arquivos.
- TANENBAUM, Andrews S. Organização Estruturada de Computadores. P. 21-42.
- VELLOSO, Fernando de Castro. Informática: conceitos básicos. Rio de Janeiro: Editora Campus, 1994, p. 15-27.
- TOLEDO, N. Introdução a Organização de Computadores. P. 37-56.
- SITES NA INTERNET
- http://www.cit.ac.nz/smac/hf100/hf100m4.htm
- http://www.well.com/user/memory/memtypes.htmUKM
- http://www.inf.ufsm.br/ bonella/m.html
- http://www.kingston.com/king/mg3.htm