Algoritmos e Estruturas de Dados

II

Prof. Ricardo J. G. B. Campello

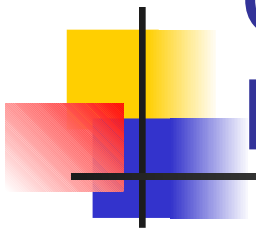
Armazenamento Secundário

Adaptado dos Originais de:

Leandro C. Cintra

Maria Cristina F. de Oliveira

Organização de Informação em Disco



- Disco:
  - conjunto de ‘pratos’ empilhados
    - Dados são gravados nas superfícies desses pratos
- Superfícies:
  - organizadas em **trilhas**
- Trilhas:
  - são organizadas em **setores**
- Cilindro:
  - conjunto de trilhas na mesma posição

Organização de Informação em Disco

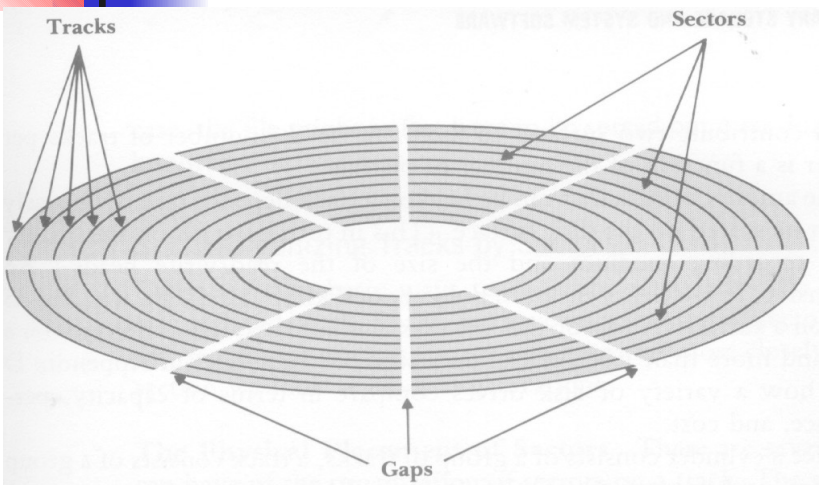


FIGURE 3.2 Surface of disk showing tracks and sectors.

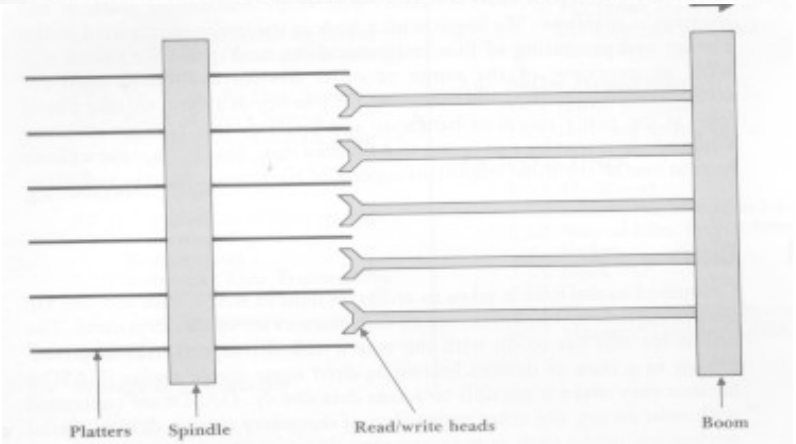
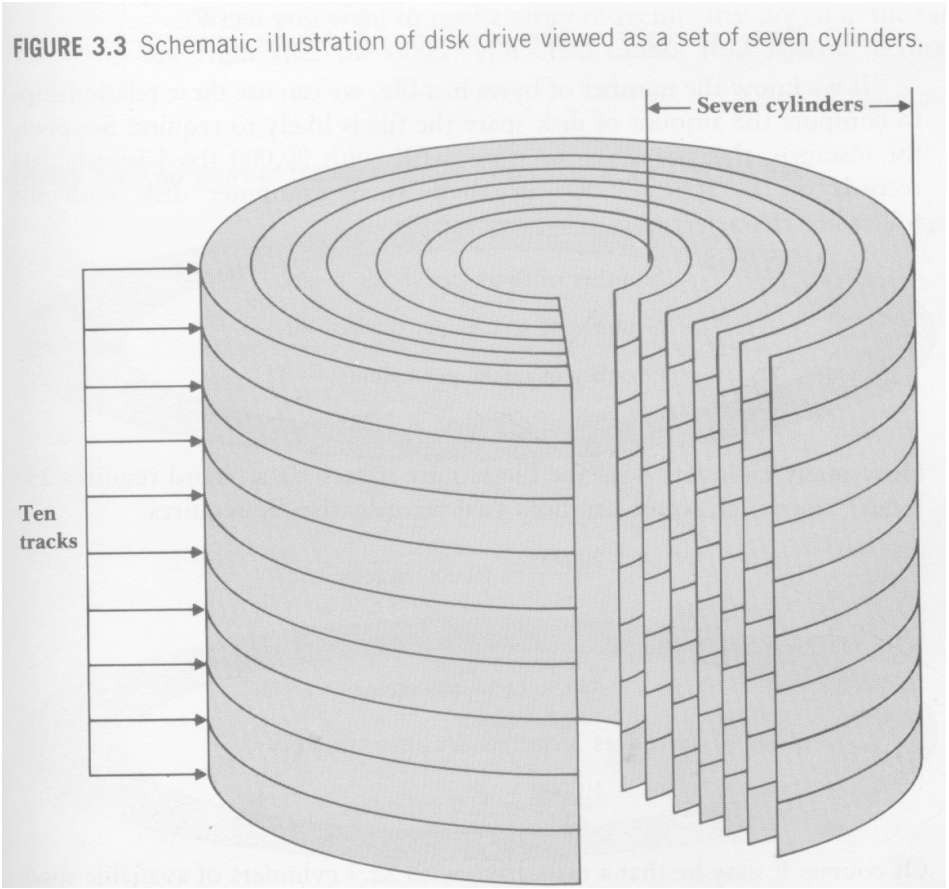
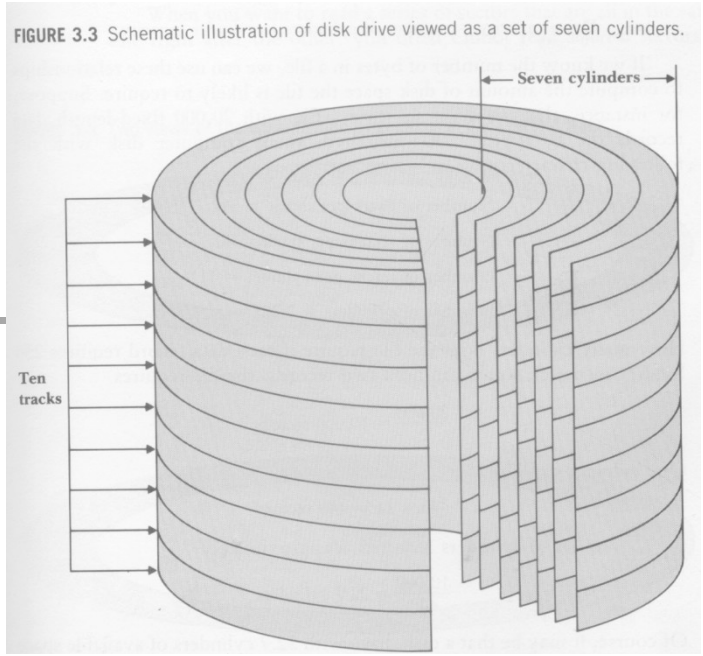


FIGURE 3.1 Schematic illustration of disk drive.

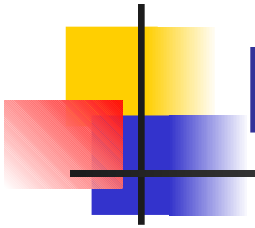
FIGURE 3.3 Schematic illustration of disk drive viewed as a set of seven cylinders.



Capacidade do Disco



- Capacidade do setor
  - nº bytes (Ex. 512 bytes)
- Capacidade da trilha
  - nº de setores/trilha * capacidade do setor
- Capacidade do cilindro
  - nº de trilhas/cilindro * capacidade da trilha
- Capacidade do disco
  - nº de cilindros x capacidade do cilindro

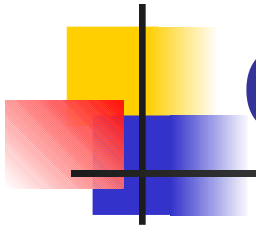


Endereços no Disco

- **Setor**: menor porção endereçável do disco
- Exemplo:
  - **fread**(pt_arq, &c, 1): lê 1 byte na posição corrente
    - Código executável faz chamada ao S.O.
      - S.O. determina em qual setor está esse byte
    - Se o setor necessário já está em um buffer de E/S:
      - acesso ao disco torna-se desnecessário
    - Caso contrário:
      - conteúdo do setor é carregado para o buffer
      - o byte desejado é lido do buffer para a RAM (endereço &c no exemplo)

Seeking Mecânico

- Movimento de posicionar a cabeça de L/E sobre a trilha/setor desejado
- O conteúdo de todo um cilindro pode ser lido com 1 único *seeking*
- É o movimento mais lento da operação leitura/escrita
  - **Gargalo:** Deve ser reduzido ao mínimo !



Cluster

- Conjunto de setores logicamente contíguos e com algum tipo de contigüidade física no disco
- Cluster pode ser lido com apenas 1 seeking

FAT – *File Allocation Table*

- Um arquivo pode ser visto pelo S.O. como um conjunto de clusters distribuídos no disco
  - Arquivos são alocados em um ou mais clusters
- Entradas na tabela determinam as localizações dos clusters de um certo arquivo lógico
  - Todos os setores de um cluster são lidos sem a necessidade de *seeking* adicional

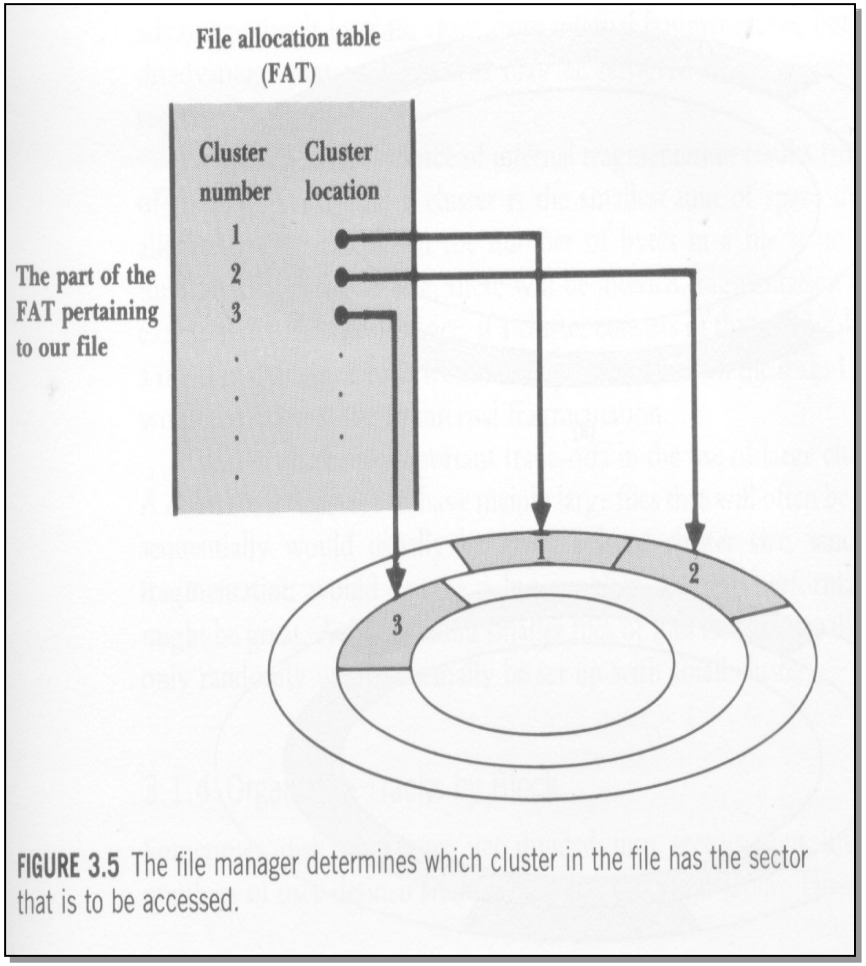
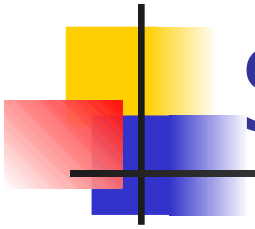
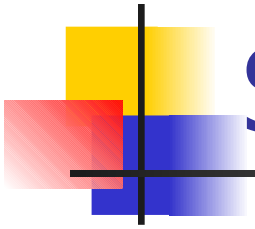


FIGURE 3.5 The file manager determines which cluster in the file has the sector that is to be accessed.



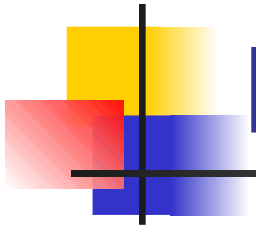
Sistema de Arquivos

- A organização do disco em setores, trilhas e cilindros é física:
  - Disco já vem pré-formatado de fábrica
  - Pré-formatação: envolve, p. ex., gerar os gaps entre setores e trilhas e armazenar, no início de cada setor, o endereço daquele setor e da trilha correspondente.
- É necessária uma formatação lógica, que 'instala' o sistema de arquivos no disco
  - Subdivide o disco em regiões endereçáveis



Sistema de Arquivos

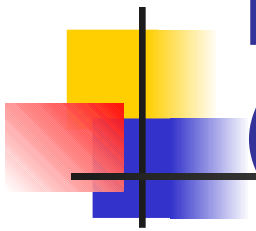
- O sistema de arquivos FAT (Windows*) endereça grupos de setores (clusters)
  - 1 *cluster* = 1 unidade de alocação
  - 1 *cluster* = n setores
- Se um programa precisa acessar um dado, cabe ao sistema de arquivos do S.O. determinar em qual cluster ele está
- Um arquivo ocupa, no mínimo, 1 *cluster*
  - Unidade mínima de alocação



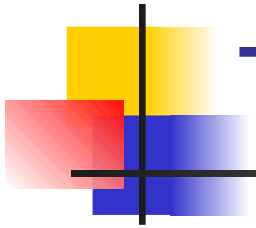
Fragmentação Interna

- Perda de espaço útil decorrente da organização em setores e clusters de tamanho fixo
- Pode ocorrer em nível de **setores** ou **clusters**

Fragmentação Interna (clusters)

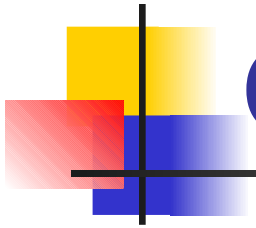


- Arquivos diferentes não compartilham clusters
- Cada arquivo ocupa no mínimo um cluster
  - Exemplo:
    - 1 cluster = 3 setores de 512 bytes
    - arquivo com 3 registros de 100 bytes cada
    - quanto espaço se perdeu?



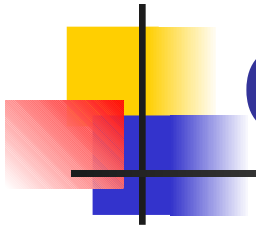
Tamanho do Cluster

- Normalmente é definido de forma automática pelo S.O.
  - quando disco é formatado
- Determinado pelo máximo que o sistema consegue manipular e pelo tamanho do disco. Exemplos:
  - FAT16 (Windows): pode endereçar 2^{16} = 65.536 clusters
  - FAT32 (Windows): pode endereçar 2^{32} clusters
- Trade-off (uso de espaço vs tempo acesso):
  - maiores clusters ⇒ maior fragmentação interna
  - maiores clusters ⇒ maior contigüidade dos arquivos



Custo de Acesso a Disco

- Fisicamente, uma combinação de 3 fatores:
  - Tempo de Busca (seek): tempo p/ posicionar o braço de acesso no cilindro correto
  - Delay de Rotação: tempo p/ o disco rodar de forma que a cabeça de L/E esteja posicionada sobre o setor desejado
  - Tempo de Transferência: tempo p/ transferir os bytes
    - (nº de bytes transferidos / nº de bytes por trilha) * tempo de rotação



Custo de Acesso a Disco

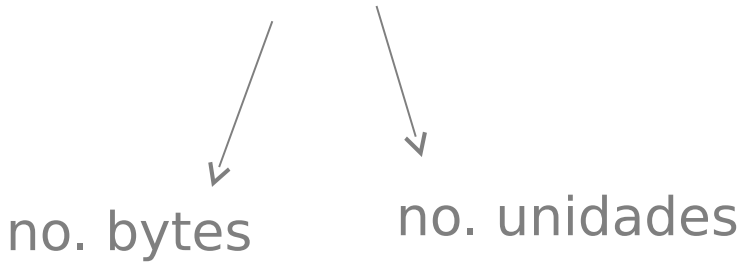
- Os tempos de acesso não são afetados apenas pelas características físicas do disco:
  - Também pela distribuição do arquivo no disco
  - Também pelo modo de acesso
    - aleatório x seqüencial

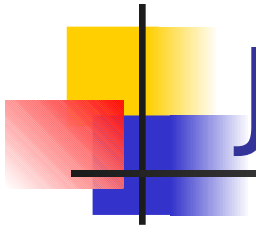
Jornada de um Byte

- O que acontece quando 1 programa escreve um byte p/ um arquivo em disco?

  pt_arq = **fopen**(“meu_arq.dat”, “a”)

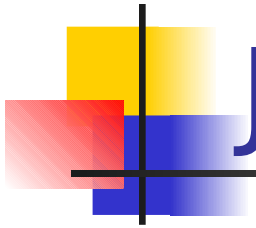
  fwrite(&c, 1, 1, pt_arq)





Jornada de um Byte

- Operações em memória:
  - O comando ativa o S.O., que supervisiona a operação
    - S. O. ativa o seu gerenciador de arquivos (**file manager**)
    - File Manager:
      - Verifica se o arquivo existe, se tem permissão de escrita, etc
      - Obtém a localização do arquivo físico correspondente ao arquivo lógico
      - Determina em que setor o byte deve ser escrito
      - Verifica se esse setor já está no **buffer** de E/S
      - Se não estiver no buffer, solicita que o setor seja carregado para o buffer
      - Escreve ou reescreve o byte correspondente no buffer



Jornada de um Byte

- Operações fora da memória
  - Processador de E/S
    - aguarda a disponibilidade do controlador de disco p/ poder efetivamente disparar a escrita
  - Controlador de disco
    - move a cabeça de L/E para trilha correta
    - localiza o setor correto sob rotação do disco
    - reescreve o setor (e o novo byte)
      - informação proveniente do buffer

Jornada de um Byte

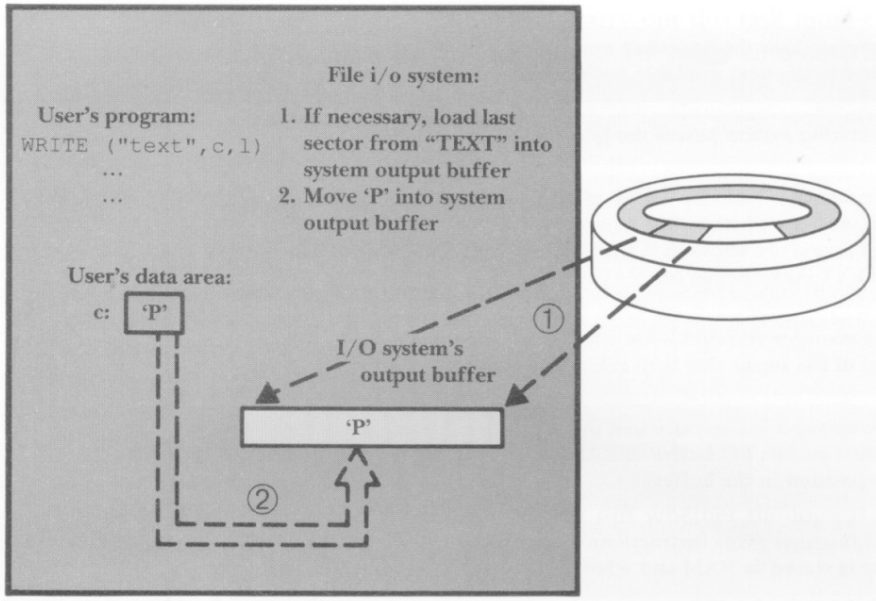


FIGURE 3.15 The file manager moves *P* from the program's data area to a system output buffer, where it may join other bytes headed for the same place on the disk. If necessary, the file manager may have to load the corresponding sector from the disk into the system output buffer.

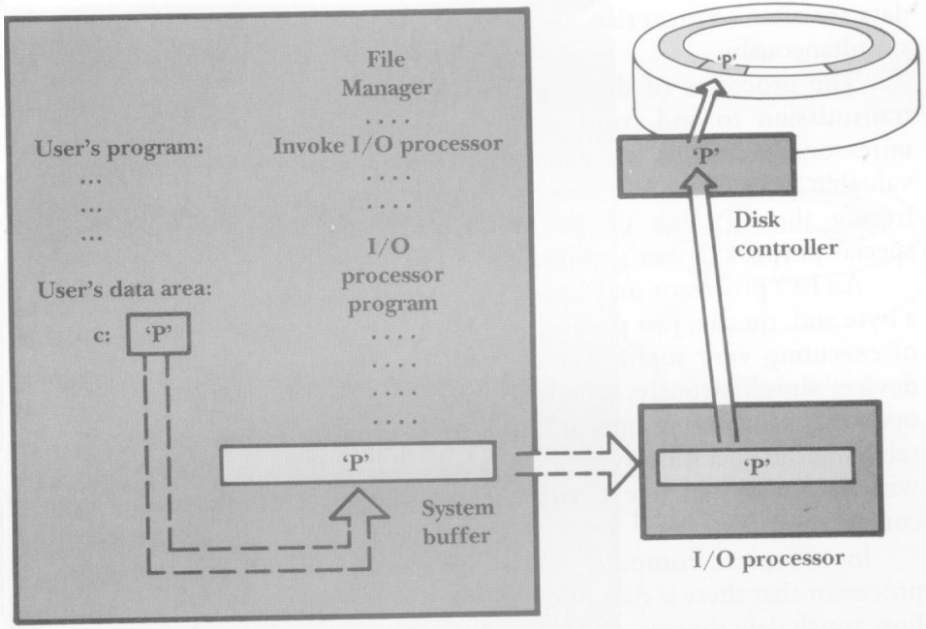
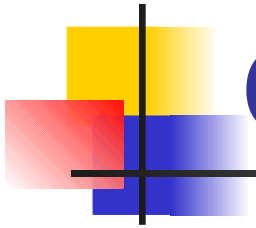
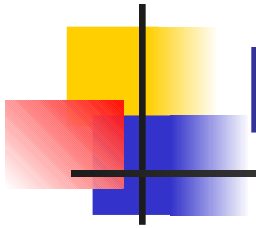


FIGURE 3.16 The file manager sends the I/O processor instructions in the form of an I/O processor program. The I/O processor gets the data from the system buffer, prepares it for storing on the disk, and then sends it to the disk controller, which deposits it on the surface of the disk.



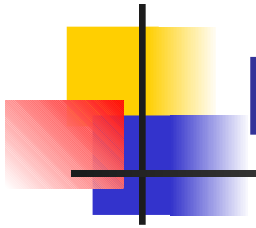
Gerenciamento de Buffer

- *Buffering:*
  - permite usar memória RAM intermediária para processar informação sendo transferida (E/S)
  - reduz nº de acessos ao dispositivo secundário



Mais Informações

- Disponíveis via alguns links...
  - http://www.pcguide.com/ref/hdd/
  - http://www.clubedohardware.com.br/tipo/3
  - http://www.gdhpress.com.br/hmc/
  - ...



Bibliografia

- **M. J. Folk and B. Zoellick, *File Structures: A Conceptual Toolkit*, Addison Wesley, 1987.**