

## 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

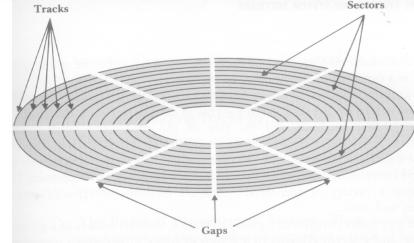


FIGURE 3.2 Surface of disk showing tracks and sectors.

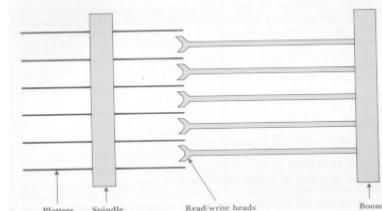


FIGURE 3.1 Schematic illustration of disk drive.

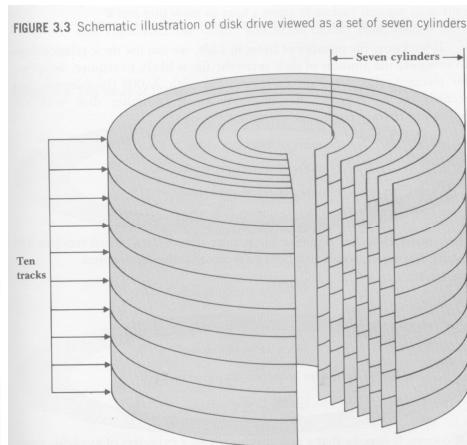
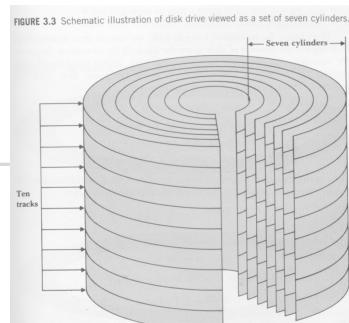


FIGURE 3.3 Schematic illustration of disk drive viewed as a set of seven cylinders.

3

## 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



4



## 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 qual superfície, trilha e setor em que 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)

5



## 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 !

6

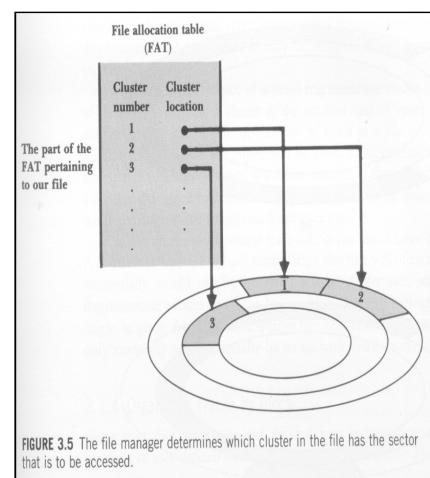
## Cluster

- Conjunto de setores logicamente contíguos e com algum tipo de contigüidade no disco:
  - Contigüidade pode não ser física
    - Tempo é requerido entre leituras de dois setores
    - Rotação do disco pode inviabilizar contigüidade física
    - Contigüidade pode ser intercalada (sincronizada)
  - Cluster pode ser lido com apenas 1 seeking

7

## 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
- Cada entrada na tabela dá a localização física de um cluster de um certo arquivo lógico
  - Todos os setores do cluster são lidos sem a necessidade de *seeking* adicional



8



## 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

9



## 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 (FAT)
- Um arquivo ocupa, no mínimo, 1 *cluster*
  - Unidade mínima de alocação

10

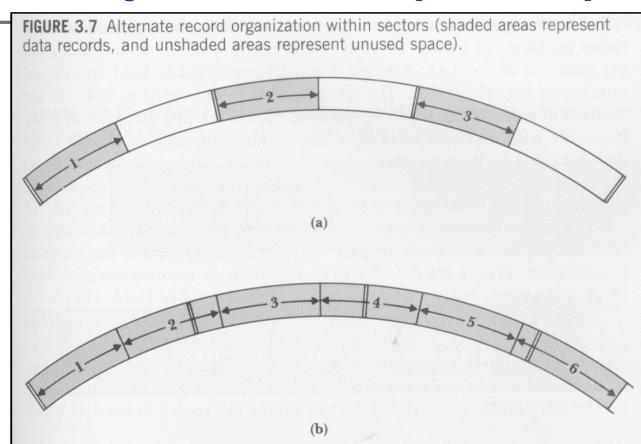
## Fragmentação Interna

- Perda de espaço útil decorrente da organização em setores e clusters de tamanho fixo:
  - Pode ocorrer em nível de **setores** ou **clusters**
  - Em nível de setores ocorre se, por simplicidade de acesso, deseja-se associar cada registro a um setor:
    - Registros não atravessam ou compartilham setores
      - Cada registro é associado a um endereço físico único
      - Possível quando tamanho registros < tamanho setor
    - Exemplo...

11

## Fragmentação Interna (Setores)

### • Exemplo:



- Em geral, evita-se a fragmentação interna de setores alocando múltiplos registros aos setores (vide figura)

12



## Fragmentação Interna (clusters)

- Fragmentação também ocorre em nível de clusters
  - Razão 1: se registros não atravessam clusters
  - Razão 2: 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?

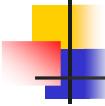
13



## Tamanho do Cluster

- Normalmente é definido de forma automática pelo S.O.
  - quando disco é formatado
- Determinado pelo máximo que a FAT 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  $\Rightarrow$  maior fragmentação interna
  - maiores clusters  $\Rightarrow$  maior contigüidade dos arquivos

14



## 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
    - $(n^o \text{ de bytes transferidos} / n^o \text{ de bytes por trilha}) * \text{tempo de rotação}$

17



## 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

18

## 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)
```

no. bytes                    no. unidades

19

## 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

20

# 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

21

# Jornada de um Byte

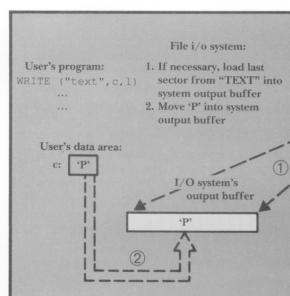


FIGURE 3.15 The file manager moves *P* from the program's data area to a system output buffer, where it may join other bytes headed for the same place on the disk. If necessary, the file manager may have to load the corresponding sector from the disk into the system output buffer.

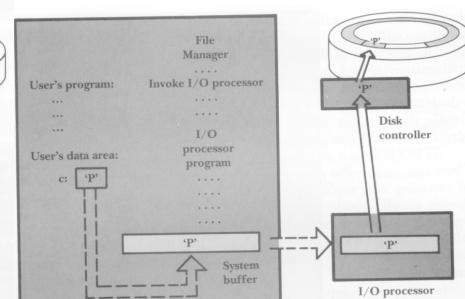


FIGURE 3.16 The file manager sends the I/O processor instructions in the form of an I/O processor program. The I/O processor gets the data from the system buffer, prepares it for storing on the disk, and then sends it to the disk controller, which deposits it on the surface of the disk.

22



## Gerenciamento de Buffer

- *Buffering:*

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

23

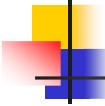


## Gerenciamento de Buffer

- **Nota:** buffer não precisa se limitar a 1 setor

- Corresponde à quantidade de info. que pode ser lida e armazenada provisoriamente em um único acesso
  - Pode-se denominar essa quantidade de **bloco**
- Por simplicidade, é conveniente raciocinar como:
  - 1 bloco = 1 cluster = n setores
  - 1 acesso = 1 bloco = um dado no. de registros
    - no. exato p/ registros de tamanho fixo
    - no. médio p/ registros de tamanho variável

24



## Buffer como Gargalo

- Buffer único:
  - programa que realiza, intercaladamente, operações de leitura/escrita, tem desempenho muito ruim
    - Exemplo crítico: ordenação externa
  - Logo, um sistema requer, no mínimo, 2 buffers:
    - 1 p/ entrada e 1 p/ saída

25



## Buffer como Gargalo

- Mesmo com 2 buffers, mover dados de e para o disco é muito lento
  - programas podem ficar 'I/O bounded'
- Para reduzir o problema:
  - Multiple buffering
    - Double buffering
    - Buffer pooling

26



## Fitas Magnéticas

- **Fitas:**

- permitem acesso seqüencial rápido
  - não permitem acesso direto
- Razoavelmente robustas e baratas
- Grande capacidade de armazenamento
- Usadas como memória **terciária**
  - back-up
  - arquivo morto

27



## Organização dos Dados na Fita

- Posição de um registro é dada por um deslocamento em bytes (*offset*) relativo ao início do arquivo
- **Posição lógica** de um byte no arquivo corresponde diretamente à sua **posição física** relativa ao início do arquivo

28

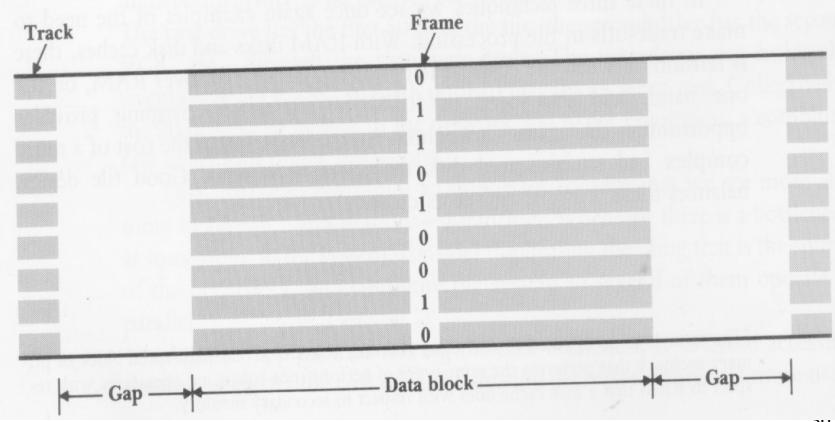
## Superfície da Fita

- A superfície pode ser vista como um conjunto de trilhas paralelas, cada qual sendo uma seqüência bits
- 9 trilhas paralelas (1 **frame**):
  - 1 **frame** = 1 byte + 1 bit de paridade
    - convenção da paridade para invalidar frames todas nulas
    - frame nula é então usada para preencher os **gaps** da fita

29

## Superfície da Fita

FIGURE 3.11 Nine-track tape.





## Superfície da Fita

- Frames são agrupadas em blocos de dados
  - blocos são separados por intervalos
  - intervalos são chamados **interblock gaps** :
    - **gaps** não possuem informação válida
    - são necessários para viabilizar parada/reinício
      - inércia mecânica

31



## Medidas de Comparação

- **Densidade:** bpi - *bytes per inch*
  - Ex: 6.250 bpi
- **Velocidade:** ips - *inches per second*
  - Ex: 200 ips
- **Tamanho do 'interblock gap':** *inches*
  - Ex: 0.3 *inches*
- 1 *inch* (polegada) ~ 2,5 cm.

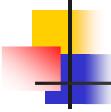
32



## Estimativa do Tamanho de Fita Necessário

- EX: armazenar em fita 1.000.000 de registros com 100 bytes cada. Suponha fita com 6.250 bpi, com intervalo entre blocos de 0.3 polegadas. Quanto de fita é necessário? Sejam:
  - $b$  = comprimento físico do bloco de dados (pol.)
  - $g$  = comprimento físico do intervalo (pol.)
  - $n$  = número de blocos de dados
  - $S$  = comprimento de fita necessário (espaço físico) é dado por:  $S=n*(b+g)$

33



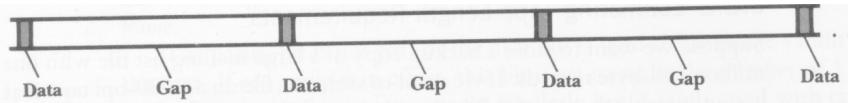
## Estimativa do Tamanho de Fita Necessário

- Supondo 1 bloco = 1 registro:  
 $S=1.000.000*(100/6.250+0.3)$   
 $S=316.000 \text{ pol} \sim 7.900 \text{ m}$
- Supondo 1 bloco=50 registros
  - $n=1.000.000/50=20.000 \text{ blocos}$
  - $b=5000/6250 \sim 0.8 \text{ pol}$
  - $S=20.000*(0.8+0.3)=22.000 \text{ pol} \sim 492 \text{ m}$
- Comprimentos típicos de fitas: 100 a 1.000 m

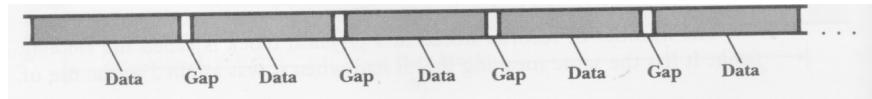
34

## Estimativa do Tamanho de Fita Necessário

- 1 registro por bloco



- 50 registros por bloco



35

## Estimativa de Tempos de Transmissão

- Taxa **nominal** de transmissão de dados:
  - densidade (bpi) \* velocidade (ips)
- Exemplo:
  - Fita de 6.250 bpi e 200 ips
  - taxa transmissão =  $6250 \times 200 = 1.250 \text{ Kb/s}$
- Taxa **real**:
  - Deve levar em conta os gaps

36



## 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/>
  - ...

37



## Bibliografia

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

38