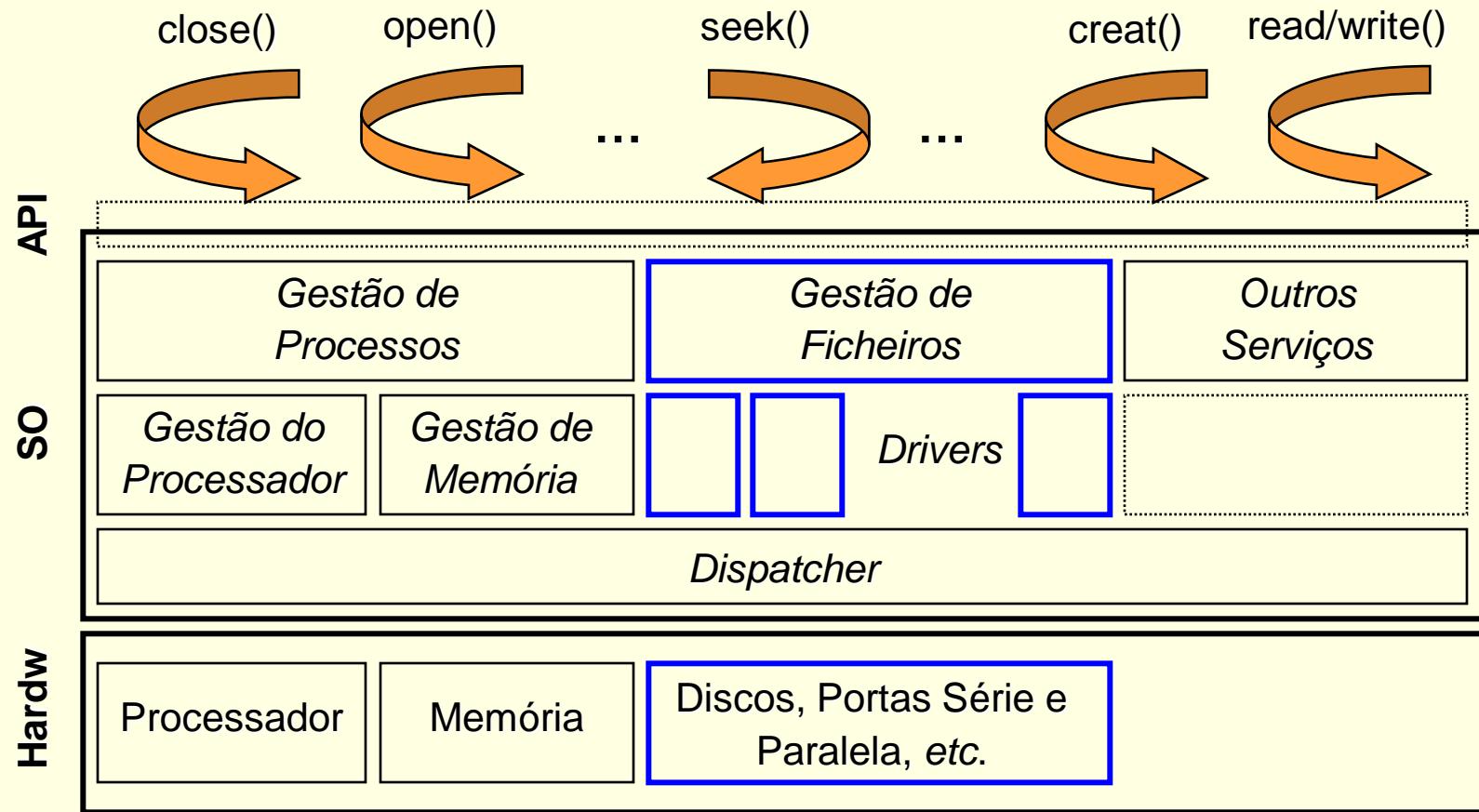


# *Fundamentos de Sistemas de Operação*

Unix Windows NT Netware Mac OS DOS/V/S Vax/VMS  
Linux Solaris HP/UX AIX Mach Chorus

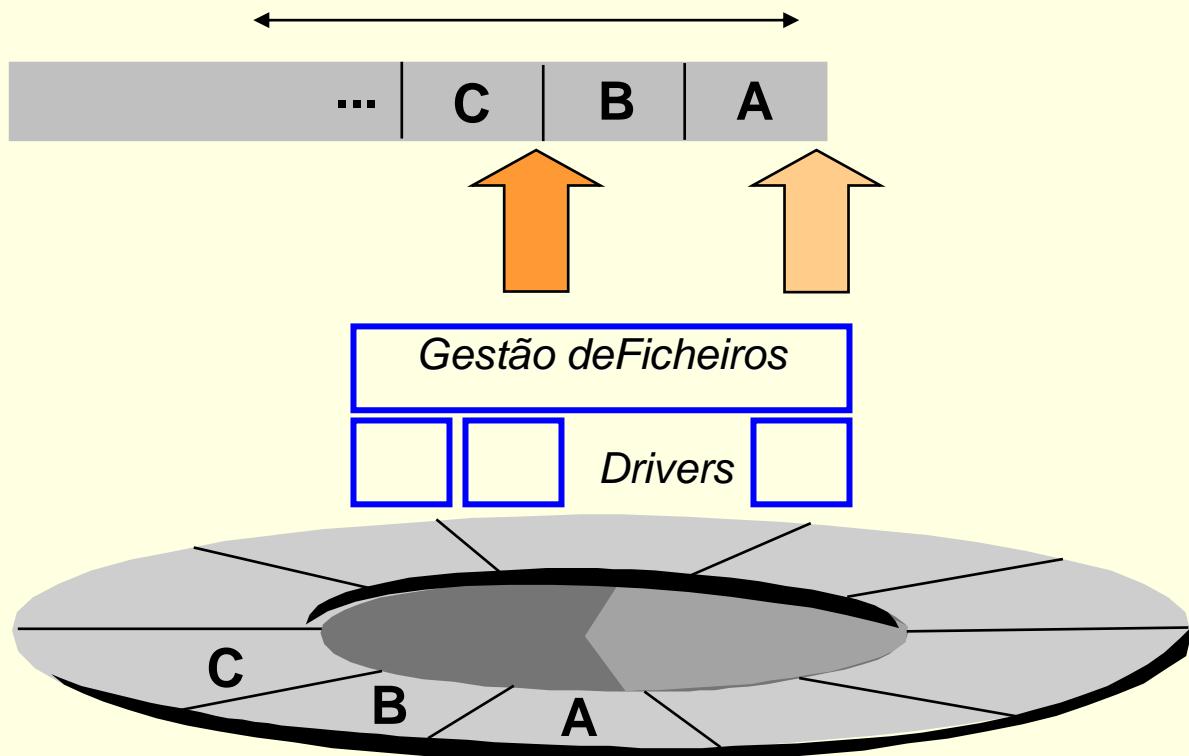
*Discos,*  
Os dispositivos que suportam os  
*Sistema de Ficheiros*

# O Sistema de Ficheiros



# *Sistemas de Ficheiros: Questões I*

- Como implementar sobre o disco o conceito de ficheiro?



# *Sistemas de Ficheiros: Questões...*

- Como identificar os ficheiros? Que regra para os nomes?
- Como saber onde começa e acaba um ficheiro?
- Como saber que blocos pertencem a um dado ficheiro?
- Dois elementos de dados sequencialmente colocados no ficheiro (lógico) têm de estar em posições contíguas do disco (físico)?

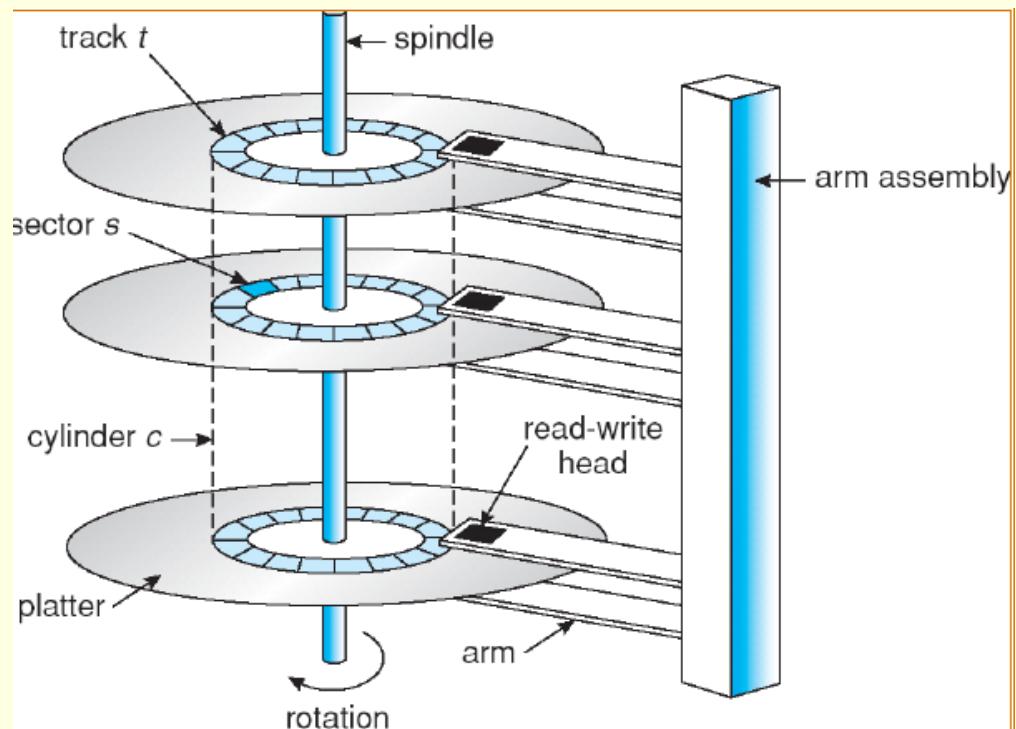
# *Organização do Disco: Questões*

- Como é organizado um disco? E são todos organizados da mesma forma? E, se não são, pode-se esconder as diferenças?
- Como é que o (software do) Sistema de Ficheiros (SF) vê o disco? O modelo apresentado pelo controlador e/ou pelo *driver* ao SF é o do verdadeiro disco físico, ou é um modelo abstracto e simplificado?

# Blocos base de armazenamento: 1 disco

## □ 1 Disco Magnético:

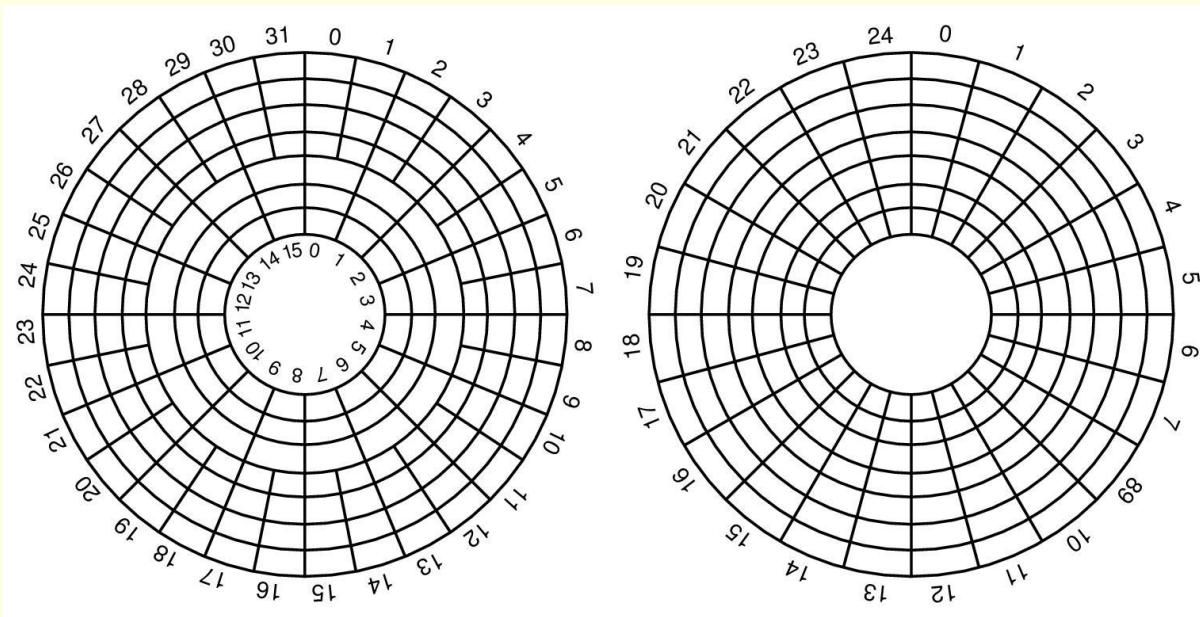
- 1) Vários pratos; em cada um,
- 2) duas superfícies de leitura/escrita; em cada uma,
- 3) várias pistas concêntricas e vários sectores radiais
- 4) Em cada superfície, uma cabeça móvel de leitura/escrita
- 5) Quando o braço se move, arrasta todas as cabeças
- 5) O desenho formado pelas pistas na mesma vertical é um cilindro
- 5) Quando o eixo gira, as cabeças podem ler/escrever todas ao mesmo tempo no cilindro



# Como é organizado um disco: I

## □ Geometria real ( $esq^a$ ) e virtual ( $dt^a$ )

- Mantendo-se a mesma densidade de gravação da periferia para o centro, no centro haverá menos sectores por pista.



Contudo, o controlador pode apresentar ao resto do "sistema" uma geometria homogénea.

# Como é organizado um disco: II

- O formato de um bloco de disco:

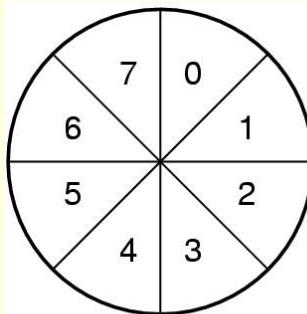


- Preâmbulo: sincronismo e informação de controle.
- Dados: tipicamente 512 bytes.
- ECC: detecção e correção de erros.

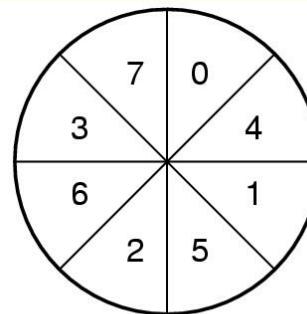
# Como é organizado um disco: III

## □ Optimização no acesso a sectores consecutivos

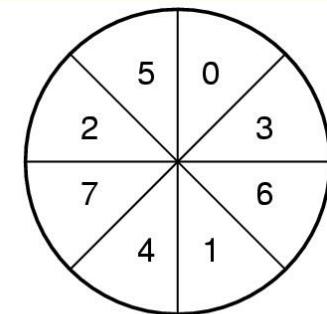
- Se o controlador não for suficientemente rápido, não tem tempo de processar um bloco e ainda assim ler o bloco imediatamente seguinte (sector consecutivo).



(a)



(b)

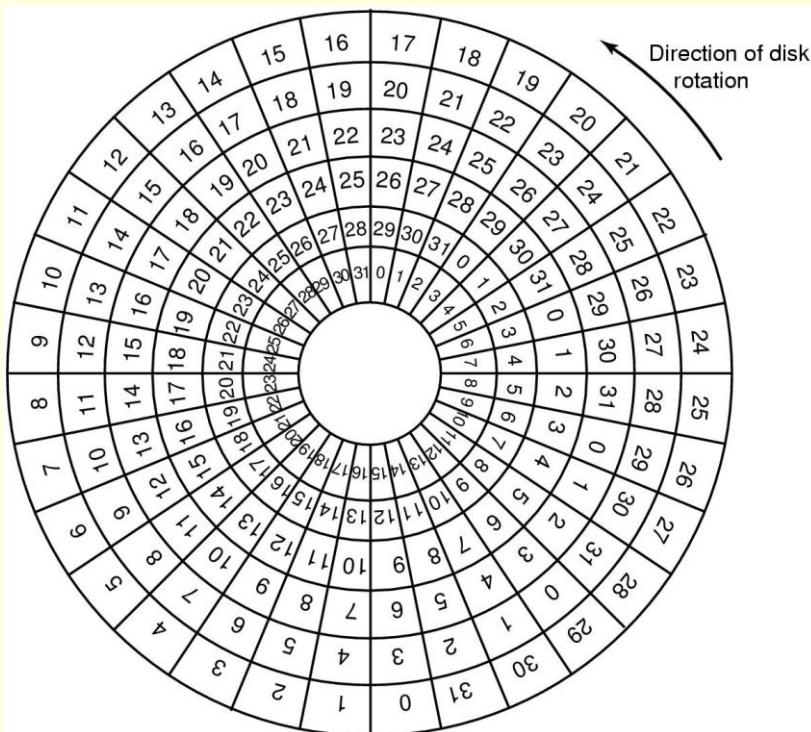


(c)

- Deixando um sector de intervalo (a – interleaving de um), dois (b – interleaving duplo), ou até mais, o controlador tem tempo de “recuperar”

# Como é organizado um disco: IV

## □ Optimização no acesso a pistas consecutivas

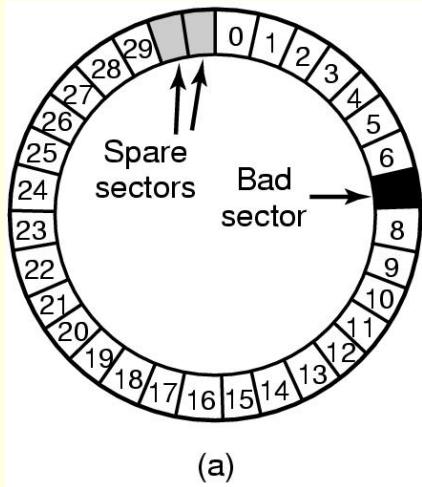


Quando o controlador acaba de ler uma sequência de blocos de uma pista n, e move a cabeça para a pista seguinte para continuar a ler os blocos seguintes, se estes não estiverem decalados (skew) em relação aos da pista anterior, então “já passaram” e o controlador terá de esperar quase uma volta inteira pela sua nova passagem “debaixo” da cabeça de leitura/escrita...

# Como é organizado um disco: V

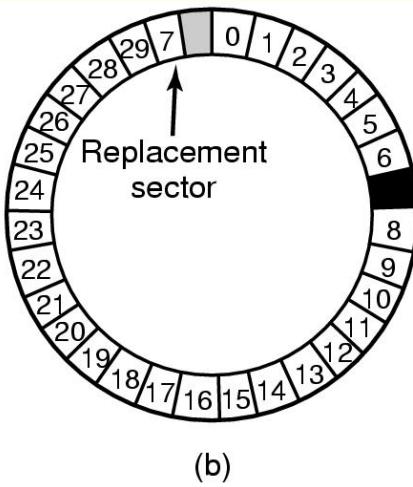
## □ Tratamento de erros:

- (a) O bloco 7 estraga-se.

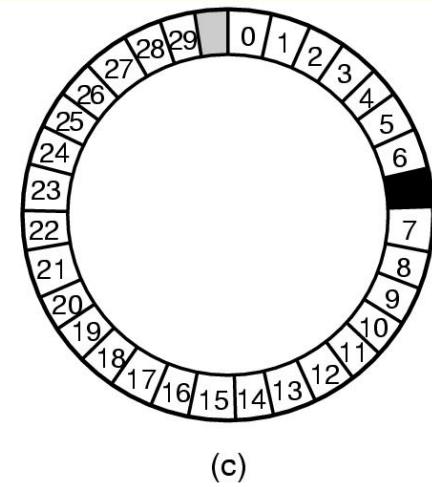


(a)

- (b) é usado um bloco de substituição.



(b)

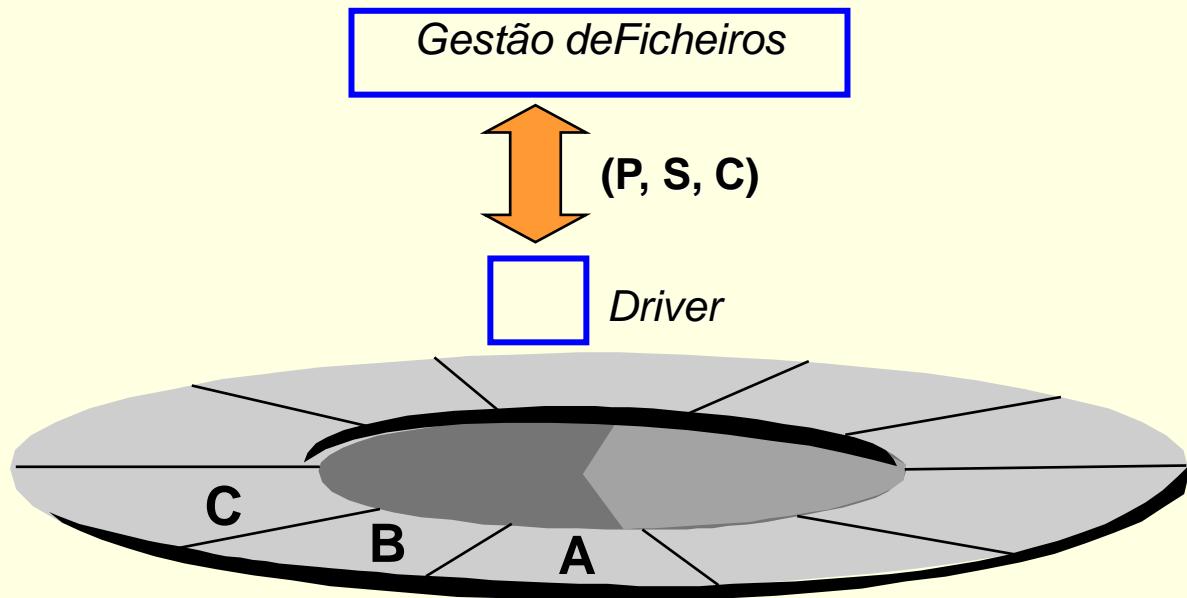


(c)

- (c) Alguns controladores poderão ser capazes de re-arranjar a numeração dos blocos para continuar a manter um bom desempenho.

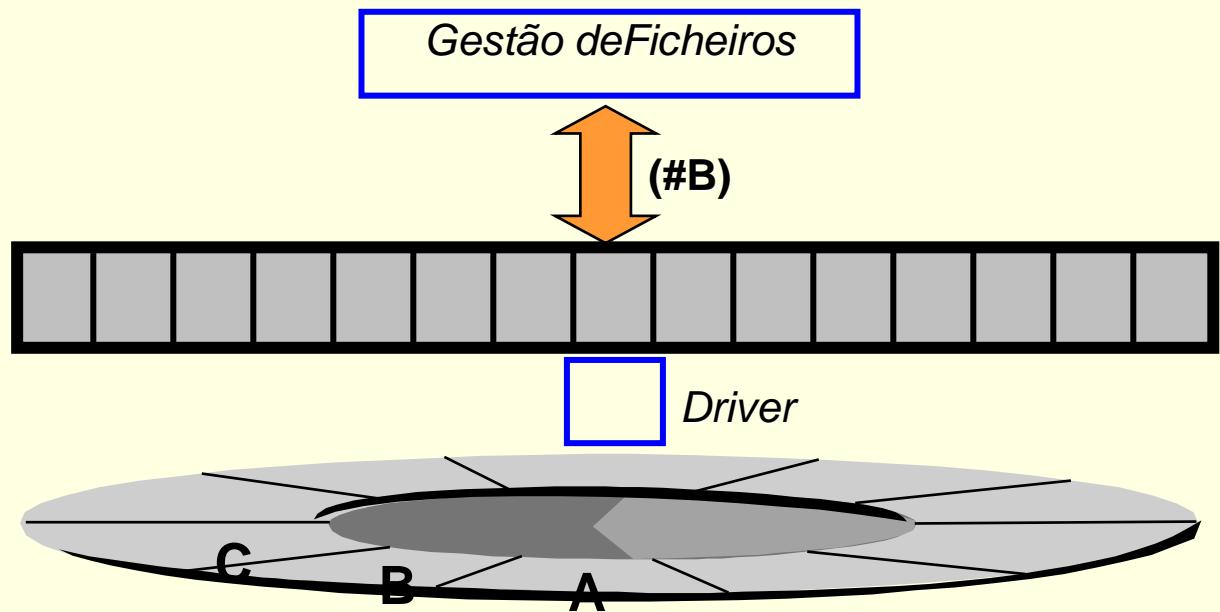
# *Modelo de Disco apresentado ao SF: I*

- O *driver* pode apresentar ao Sistema de Ficheiros um modelo (real ou virtual) em quando o acesso é pedido ao driver pelo SF este tem de especificar os números de pista, sector, cabeça... (discos antigos)



# Modelo de Disco apresentado ao SF: II

- Ou o *driver* pode, por outro lado, apresentar ao Sistema de Ficheiros um modelo de um **vector de blocos** em que o acesso é efectuado especificando apenas o número do bloco



# *Disco: velocidades (1)*

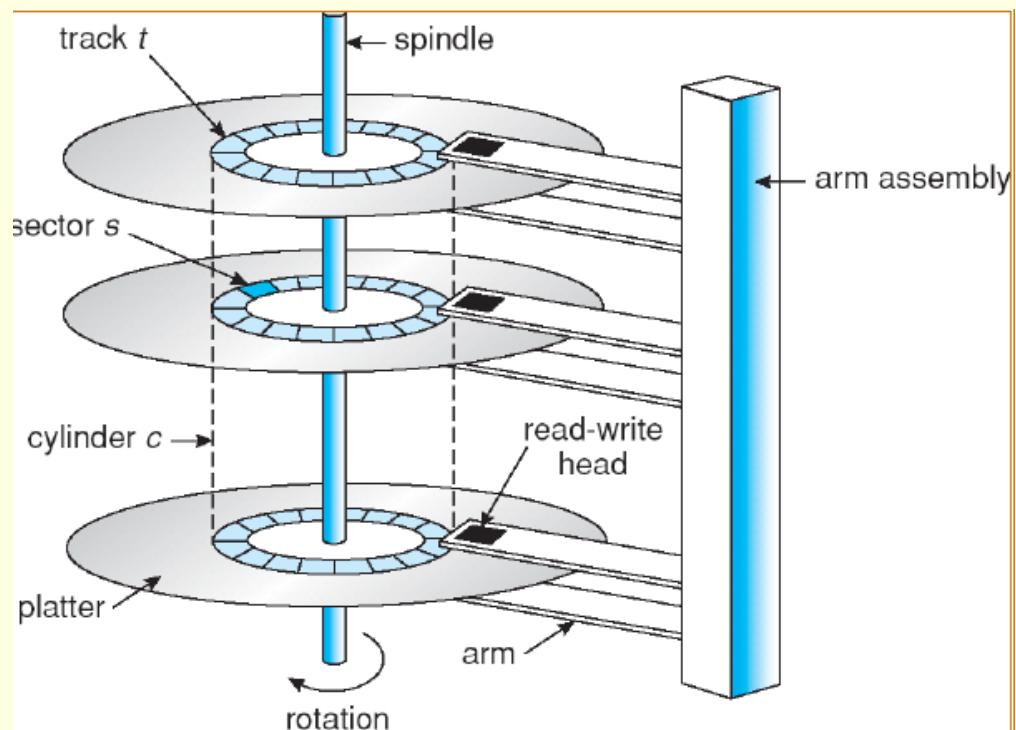
## □ Access time =

- 1) Seek time (mover braço p/ cilindro desejado) +
- 2) Latência de rotação (esperar p/ sector(es) alvo) +
- 3) Tempo de transferência disco-pa-controlador (sectores copiados p/ controlador) +
- 4) Tempo de transferência controlador-para-host (cópia da cache do controlador p/ RAM do sistema)

Exemplo: ST3300657FC

- 1) 0.2(R)/0.4(W) ms T2T<sup>1</sup>
- 2) 2ms avg. @ 15K rpm
- 3) 1450-2370 Mb/s
- 4) 400 MB/s (FC)

<sup>1</sup> Track-to-Track: Tempo de Seek para 1 pista vizinha seguido de Read ou Write



# *Disco: velocidades* (2)

- O comportamento (velocidade) de um disco,
  - É como de um carro numa pista de corridas ou na cidade: muito diferente...
- Assim, um disco pode ter um desempenho
  - Muito elevado, quando está a ler (ou escrever) sequencialmente, pois “apanha” os sectores todos seguidos e lê em todas as pistas ao mesmo tempo... É um teste usado para definir a largura de banda “*sustained*” (para o disco Seagate ST3300657FC, ~200MB/s)
  - Bastante sofrível quando lhe é pedido que, aleatoriamente (random) leia um bloco numa pista, outro noutra ao acaso (pode até estar “distante”). É um teste usado para definir o número de I/Os capaz de fazer por segundo (IOPS ST3300657FC < 200)

# *Disco: velocidades* (3)

## □ *Largura de Banda: muito baixa...*

- Um HDD rápido consegue  $\approx 200 \text{ MB/s}$
- A memória consegue  $\approx 200 \text{ GB/s}$
- 1 core consegue  $\approx 200 \text{ GB/s}$



*1000x increase*

## □ *Latência muito elevada*

- Um HDD rápido consegue  $\approx 2 \text{ ms}$
- A memória consegue  $\approx 100 \text{ ns}$
- Um registo consegue  $\approx 0.5 \text{ ns}$



*20 000x melhor que o HDD*



*200x melhor que a RAM*

# Números que todos devem saber...

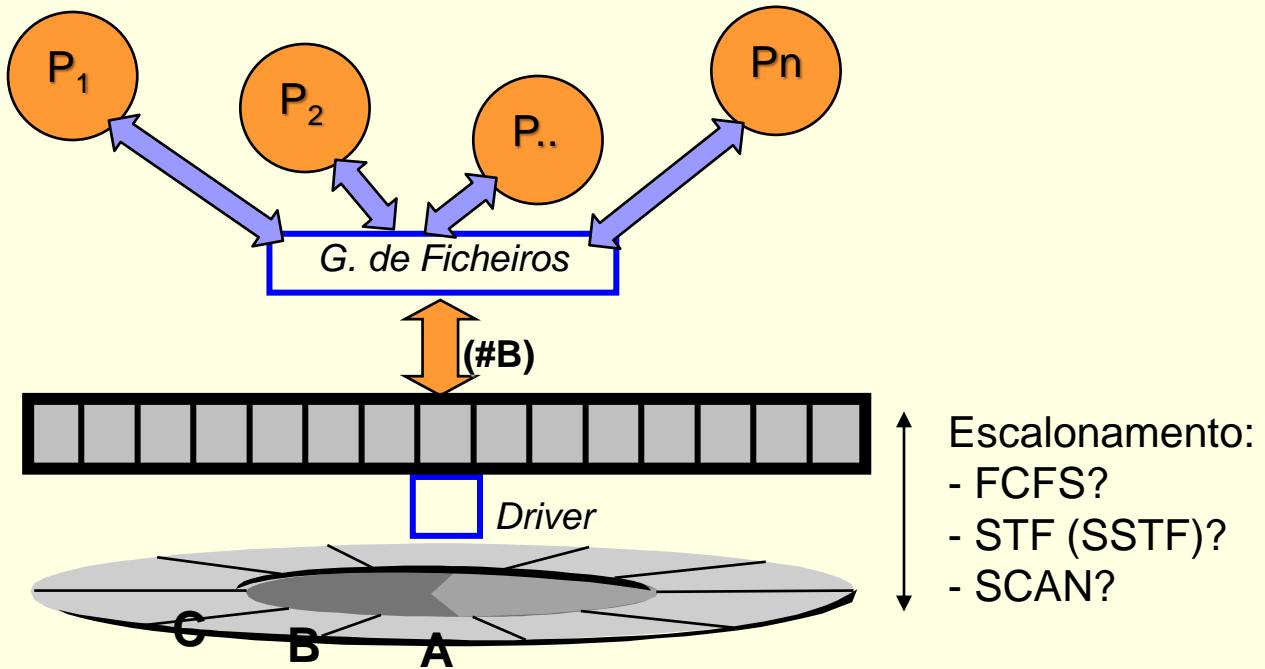
0.5 ns - CPU L1 dCACHE reference  
1 ns - speed-of-light (a photon) travel a 1 ft (30.5cm) distance  
5 ns - CPU L1 iCACHE Branch mispredict  
7 ns - CPU L2 CACHE reference  
71 ns - CPU cross-QPI/NUMA best case on XEON E5-46\*  
100 ns - MUTEX lock/unlock  
100 ns - own DDR MEMORY reference  
135 ns - CPU cross-QPI/NUMA best case on XEON E7-\*  
202 ns - CPU cross-QPI/NUMA worst case on XEON E7-\*  
325 ns - CPU cross-QPI/NUMA worst case on XEON E5-46\*  
10,000 ns - Compress 1K bytes with Zippy PROCESS  
20,000 ns - Send 2K bytes over 1 Gbps NETWORK  
250,000 ns - Read 1 MB sequentially from MEMORY  
500,000 ns - Round trip within a same DataCenter  
10,000,000 ns - DISK seek  
10,000,000 ns - Read 1 MB sequentially from NETWORK  
30,000,000 ns - Read 1 MB sequentially from DISK  
150,000,000 ns - Send a NETWORK packet CA -> Netherlands

		ns
	us	
ms		

<https://stackoverflow.com/questions/4087280/approximate-cost-to-access-various-caches-and-main-memory>

# Acesso ao Disco: aleatório ou sequencial?

- O acesso aleatório é o mais frequente pois a) há vários processos a aceder a ficheiros e, mesmo que cada um fosse sequencial, para os satisfazer (escalonar) de forma justa, tem de se servir uns e outros; b) como os ficheiros não estão contíguos, mesmo um acesso sequencial acaba por ser ... aleatório



# *Demo*: disk.py

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