



SISTEMAS OPERATIVOS

Gestão de discos

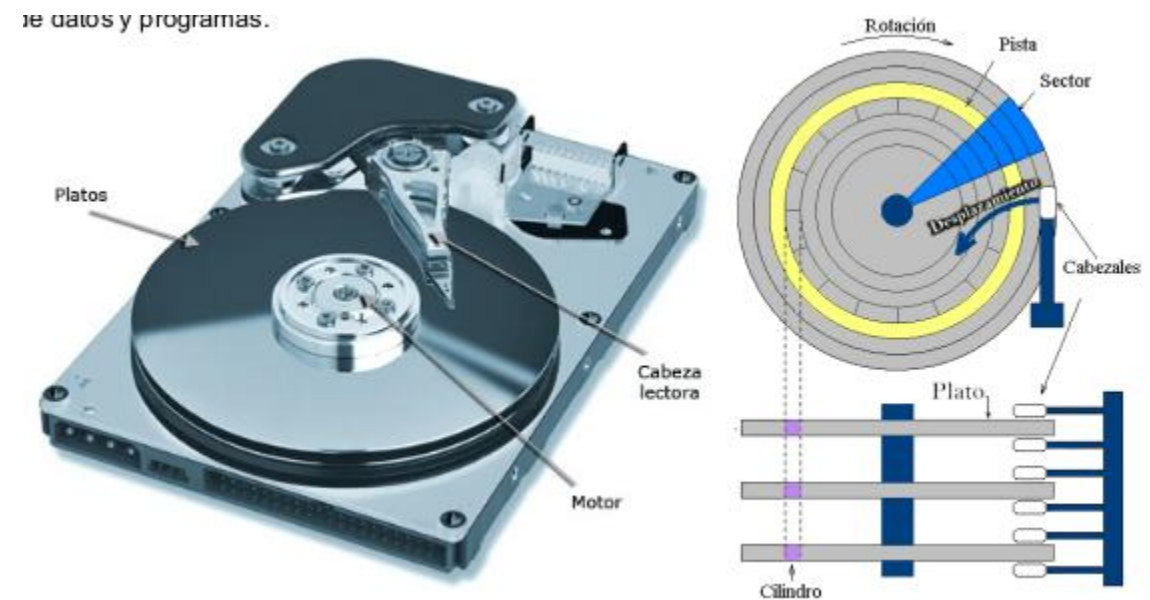
António Godinho

GESTÃO DE ACESSO A DISCOS: ORGANIZAÇÃO

Discos rígidos inventados pela IBM nos anos 50, nessa altura designados pelo nome de código Winchester, com 20" de diâmetro e capacidade de alguns MB.

Organização física dos discos:

São constituídos por vários discos magnéticos concêntricos fixos num eixo de rotação. As superfícies são divididas em pistas que por sua vez são divididas em sectores (normalmente de 512 bytes). As pistas das várias superfícies formam os cilindros.

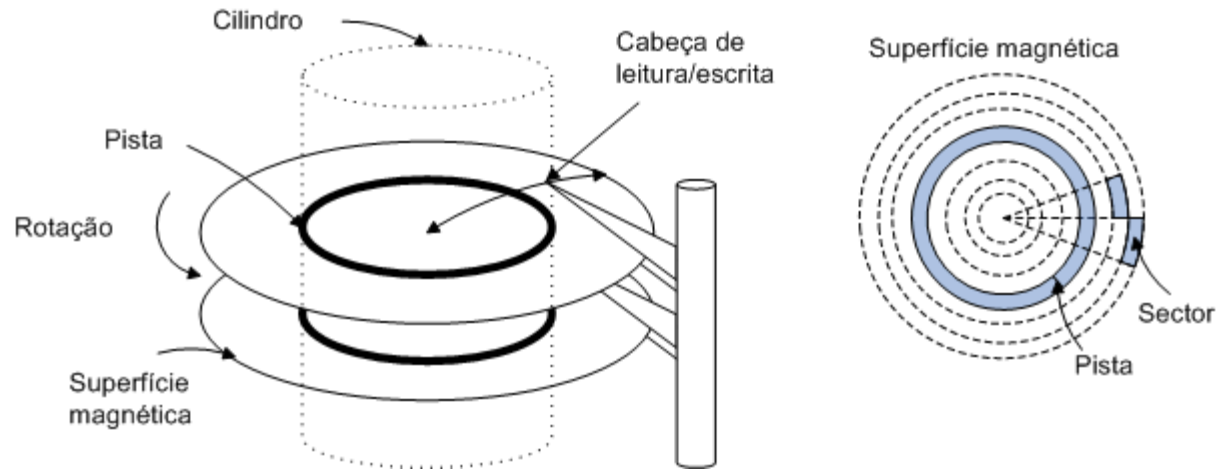


GESTÃO DE ACESSO A DISCOS: ORGANIZAÇÃO

Organização lógica:

Os discos são organizados, do ponto de vista lógico, como uma tabela unidimensional de blocos lógicos. O bloco lógico constitui a unidade mínima endereçável numa transferência de E/S.

O **sector 0** é o primeiro sector da primeira pista do cilindro mais exterior. O mapeamento prossegue ao longo dessa pista, continuando ao longo das pistas do mesmo cilindro e ao longo dos cilindros mais interiores.



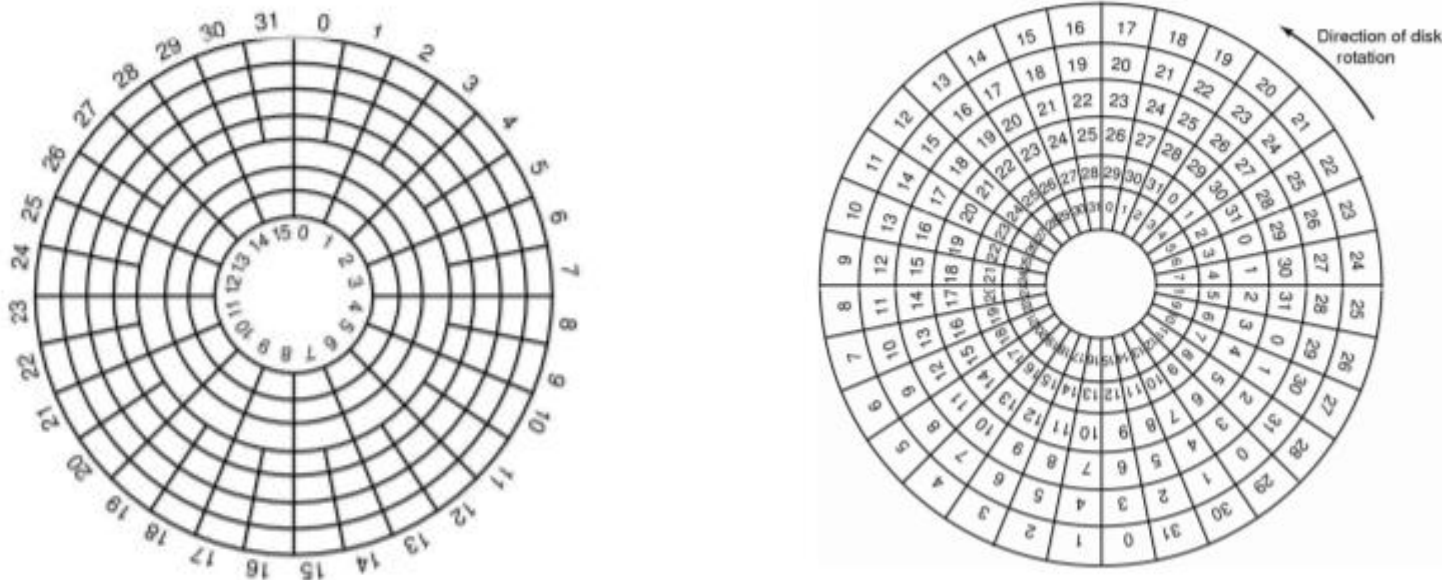
GESTÃO DE ACESSO A DISCOS: ORGANIZAÇÃO

Variação do perímetro modifica número de sectores por pista

Na figura apresenta-se uma distribuição por zonas, com maior número nas zonas exteriores

Todos os sectores do HDD são etiquetados, por forma a que a cabeça de escrita/ leitura saiba a sua posição. Formas de etiquetagem:

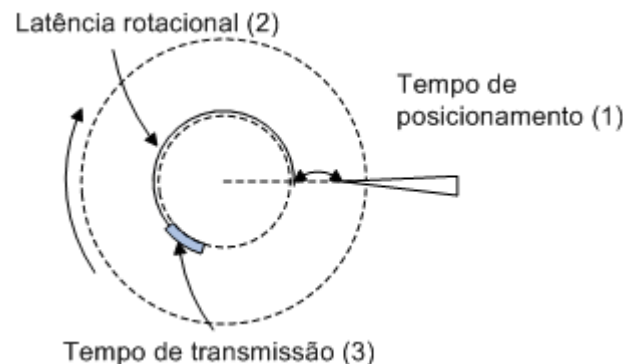
Cilindro/pista/sector – Número sequencial (usado pelos discos atuais): na pista a ordenação é sequencial, com deslocamento entre pistas para otimizar tempo de acesso de ficheiros longos



GESTÃO DE ACESSO A DISCOS: TEMPOS DE ACESSO

O tempo de acesso a um sector é constituído por:

- Tempo de posicionamento (seek time) – corresponde ao tempo consumido na deslocação das cabeças de leitura/escrita para a pista pretendida.
- Latência rotacional (rotational latency) – corresponde ao tempo consumido até que as cabeças de leitura/escrita estejam posicionadas sobre a área correspondente ao sector pretendido.
- Tempo transmissão – corresponde ao tempo de leitura/escrita.

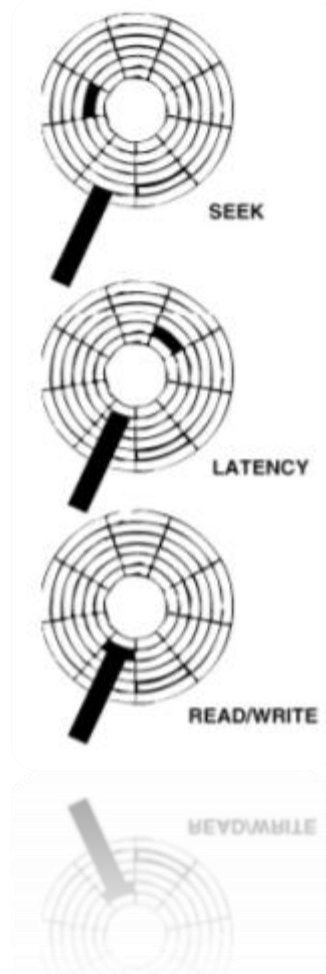


$$T_A = T_S + \frac{1}{2r} + \frac{b}{rN}$$

Tempo deslocamento

Tempo leitura

$\left\{ \begin{array}{l} r - \text{velocidade rotação} \\ b - \text{número Bytes lidos} \\ N - \text{número Bytes por faixa} \end{array} \right.$



GESTÃO DE ACESSO A DISCOS: TEMPOS DE ACESSO

Algoritmos de escalonamento de acessos: porquê?

- Um incremento no desempenho no acesso aos discos melhora significativamente o desempenho global do sistema computacional.
- Se os pedidos de acesso são atendidos por ordem de chegada, os sucessivos posicionamentos em localizações dispersas podem conduzir a tempos de acessos muito elevados, degradando significativamente o desempenho global.
- Objectivo: optimização dos tempos médios de acesso, por diminuição dos tempos médios de posicionamento ou de latência rotacional.
- As soluções podem ser baseadas em software (SO) ou hardware (controladores).

GESTÃO DE ACESSO A DISCOS: ESCALONAMENTO (1)

First-come-first-served (FCFS):

Os pedidos são atendidos por ordem de chegada.

Vantagens:

- Justeza no acesso;

- Garante a espera limitada;

- Algoritmo pouco complexo.

Desvantagens:

Input:

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Output:

Seek Sequence is

176 → 79 → 34 → 60 → 92 → 11 → 41 → 114

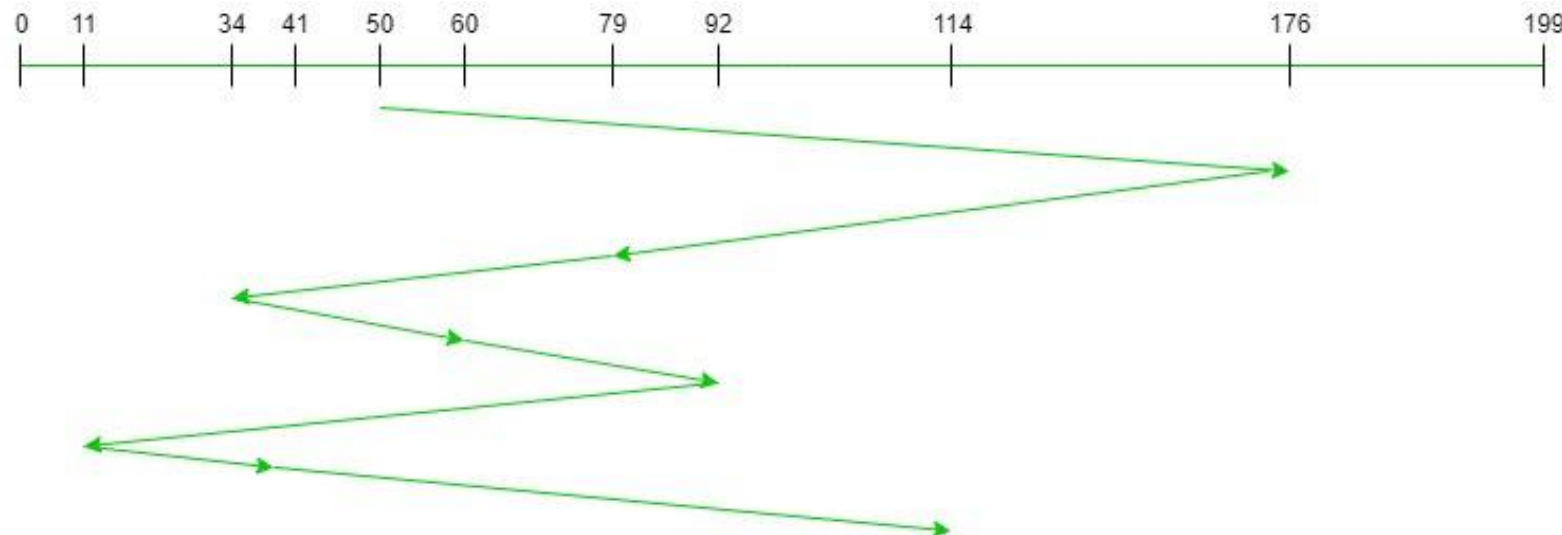
Cilindros percorridos

$(176-50)+(176-79)+(79-34)+(60-34)+(92-60)$

$+(92-11)+(41-11)+(114-41)$

= 510

Total number of seek operations = 510



GESTÃO DE ACESSO A DISCOS: ESCALONAMENTO (2)

Shortest-Seek-Time-First (SSTF):

Atende aos pedidos que correspondem ao menor tempo de posicionamento.

Vantagens:

- Apropriado para processamento em batch.

- Melhor desempenho do que o algoritmo de programação FCFS.

- Tem um tempo médio de resposta e de espera inferior.

Desvantagens:

- Não garante justiça no acesso (starvation);

- Não garante espera limitada;

- Tempos de resposta muito variáveis;

- Pouco apropriado para sistemas interactivos.

Input:

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Output:

Seek Sequence is

176 → 79 → 34 → 60 → 92 → 11 → 41 → 114

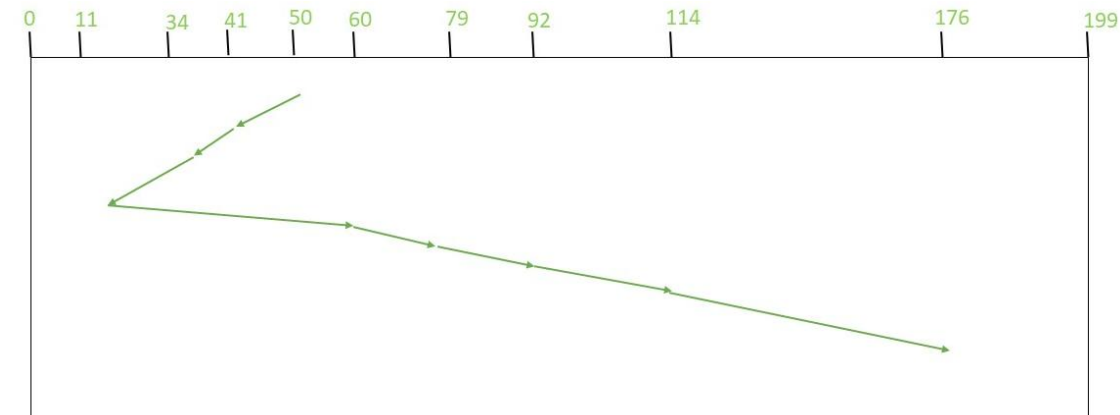
Cilindros percorridos

$= (50-41) + (41-34) + (34-11) + (60-11) + (79-60) + (92-79) + (114-92) + (176-114)$
 $= 204$

Total number of seek operations = 204

which can also be directly calculated as:

$(50-11) + (176-11)$



GESTÃO DE ACESSO A DISCOS: ESCALONAMENTO (3)

Shortest seek time in preferred direction (SCAN):

Algoritmo mais complexo de implementar

Efectua o varrimento dos cilindros, num e noutro sentido. Em cada um dos sentidos de varrimento, atende aos pedidos de acordo com o menor tempo de posicionamento.

Não garante espera limitada (pode ficar bloqueado sempre no mesmo cilindro).

Input:

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Direction = left (We are moving from right to left)

Output:

Seek Sequence is

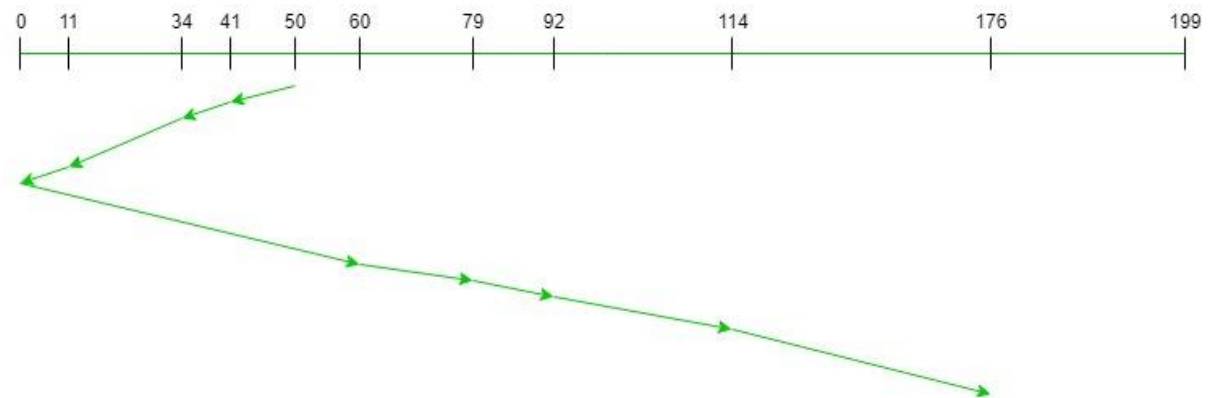
41 → 34 → 11 → 0 → 60 → 79 → 92 → 114

Cilindros percorridos

$= (50-41) + (41-34) + (34-11) + (11-0) + (60-0) + (79-60) + (92-79) + (114-92) + (176-114)$

$= 226$

Total number of seek operations = 226



<https://www.geeksforgeeks.org/scan-elevator-disk-scheduling-algorithms/>

GESTÃO DE ACESSO A DISCOS: ESCALONAMENTO (4)

Circular SCAN (C-SCAN):

Efectua o varrimento dos cilindros apenas num sentido. Atende aos pedidos de acordo com o menor tempo de posicionamento.

Não garante espera limitada (pode ficar bloqueado sempre no mesmo cilindro).

Tem mais movimentos de procura em comparação com o algoritmo SCAN.

Input:

Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}

Initial head position = 50

Direction = right(We are moving from left to right)

Output:

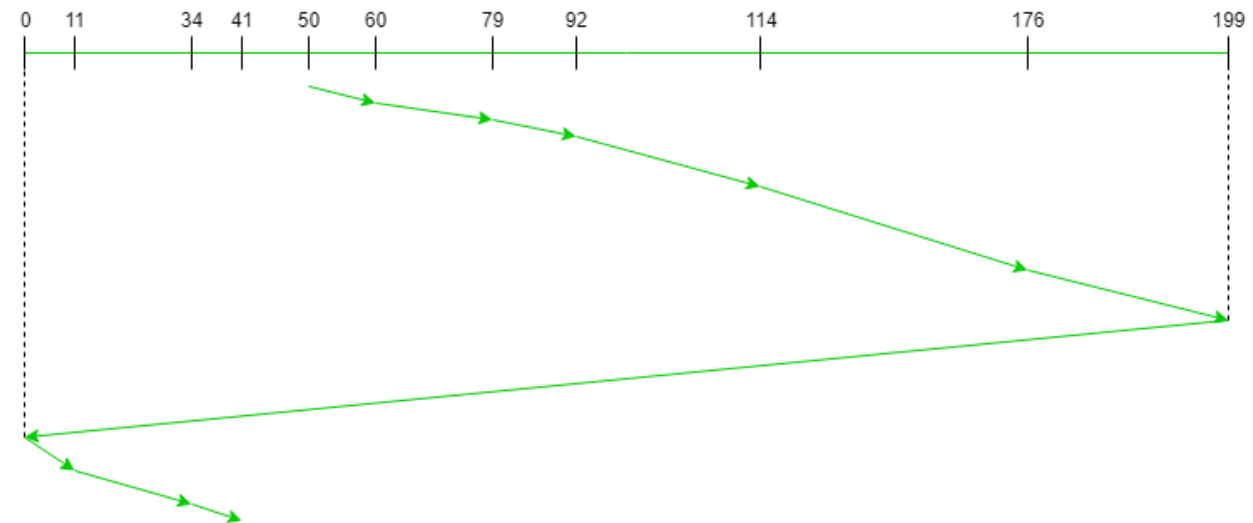
Seek Sequence is

60 → 79 → 92 → 114 → 176 → 199 → 0 → 11 → 34 → 41

Cilindros percorridos

$= (60-50) + (79-60) + (92-79) + (114-92) + (176-114) + (199-176) + (199-0) + (11-0) + (34-11) + (41-34)$

Total number of seek operations = 389



GESTÃO DE ACESSO A DISCOS: ESCALONAMENTO (5)

Freezing SCAN (FSCAN):

De modo a garantir a espera limitada, a lista de pedidos é periodicamente "congelada", isto é, os novos pedidos só serão atendidos depois dos pedidos da lista "congelada" serem todos atendidos por aplicação do algoritmo SCAN.

N-Step SCAN :

De modo a garantir a espera limitada, a lista de pedidos é dividida em grupos de N. A cada grupo de N pedidos é aplicado o algoritmo SCAN. Se $N=1$, o algoritmo degenera no FCFS.

LOOK:

Baseado no SCAN, mas, quando percorre um sentido, verifica se nesse sentido existe mais algum pedido para ser atendido. Caso não exista, inverte o sentido de varrimento.

Exemplo:

Sequência de pedidos: (63), 33, 72, 47, 8, 99, 74, 52, 75

Sequência percorrida: 63, 52, 47, 33, 8, 72, 74, 75, 99

Circular LOOK (C-LOOK):

Baseado no LOOK, mas aplica o C-SCAN (varrimento num só sentido).

Exemplo:

Sequência de pedidos: (63), 33, 72, 47, 8, 99, 74, 52, 75

Sequência percorrida: 63, 52, 47, 33, 8, 99, 75, 74, 72

GESTÃO DE ACESSO A DISCOS: OUTRAS TÉCNICAS

✓ Caches:

- ✓ Armazenam cópias dos dados em memória mais rápida (memória principal ou na memória do controlador de disco).
- ✓ Melhora os tempos médios de acesso.
- ✓ Potencia a inconsistência de dados, nomeadamente, quando ocorre uma falha de energia.
- ✓ Write-back caching: Os dados são guardados em cache e só copiados para disco periodicamente. Bom desempenho.
- ✓ Write-through caching: Os dados são guardados simultaneamente em cache e disco. Menor desempenho mas garante a consistência da informação.

✓ Desfragmentação:

- ✓ Coloca dados relacionados (conteúdo de ficheiros) em sectores contíguos.
- ✓ Diminui o número de operações de posicionamento.

✓ Compressão:

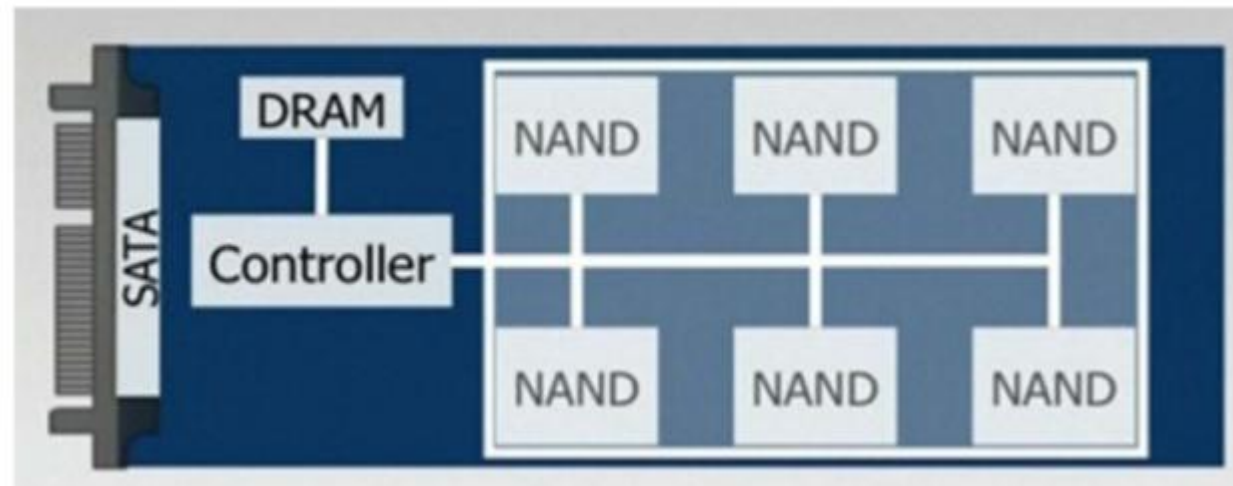
- ✓ Consome menos espaço em disco.
- ✓ Diminui tempos de acesso e transferência de dados.
- ✓ Aumenta o esforço computacional para realizar a compressão/descompressão.

GESTÃO DE ACESSO A DISCOS

- ✓ Que algoritmos escolher?
- ✓ Para sistemas operativos com pouca carga, usar o SSTF.
- ✓ Para sistemas operativos de carga elevada (ex: servidores), usar o C-SCAN ou C-LOOK.
- ✓ O Linux adotou o algoritmo C-LOOK.

SSD

- ✓ Não possuem arquitetura mecânica
- ✓ Baseado no conceito de memórias voláteis, sem que a informação se perca ao perder a energia, o SSD utiliza de um banco de portas lógicas do tipo NAND para armazenar os dados
- ✓ memória do tipo DRAM (Dynamic Random Access Memory), que cumpre a função de cache de dados



SSD

✓ Controlador de memória

- ✓ Proporciona o caminho mais apropriado para o fluxo de dados entre as memórias flash e a interface de comunicação, bem como o protocolo para este transporte.
- ✓ Maximiza a velocidade de transferência de dados, de forma também a garantir a consistência dos mesmos
- ✓ Os módulos de interface flash, ou "FIM", são responsáveis por mapear os setores do banco de NANDs, transformando endereços lógicos que chegam na interface em endereços físicos. Para este fim, o componente faz uso de uma tabela de endereços, ou até mesmo de ponteiros

