ACH2024

Organização de arquivos: alocação sequencial (parte 1)

Prof Helton Hideraldo Bíscaro

Aula passadas

- Organização interna de arquivos (delimitação de campos e registros)
  - Aproveitamento de espaço interno
  - Tempo para localizar um campo ou registro
- Estrutura física de discos rígidos

Memória primária x secundária

Mem. volátil

Mem. secundária

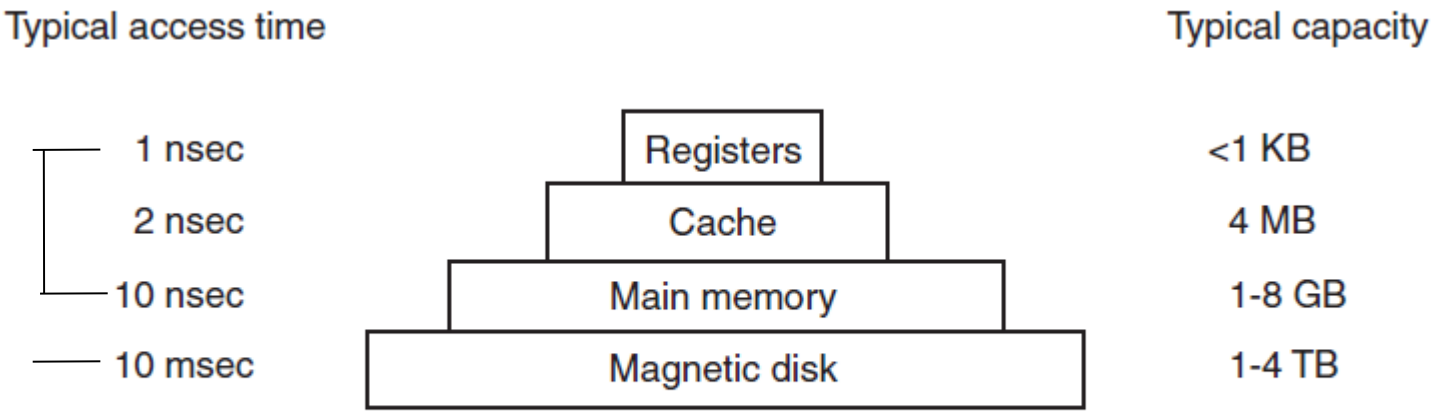


Figure 1-9. A typical memory hierarchy. The numbers are very rough approximations.

(TANEMBAUM & BOS, 2015)

Estrutura de um HD

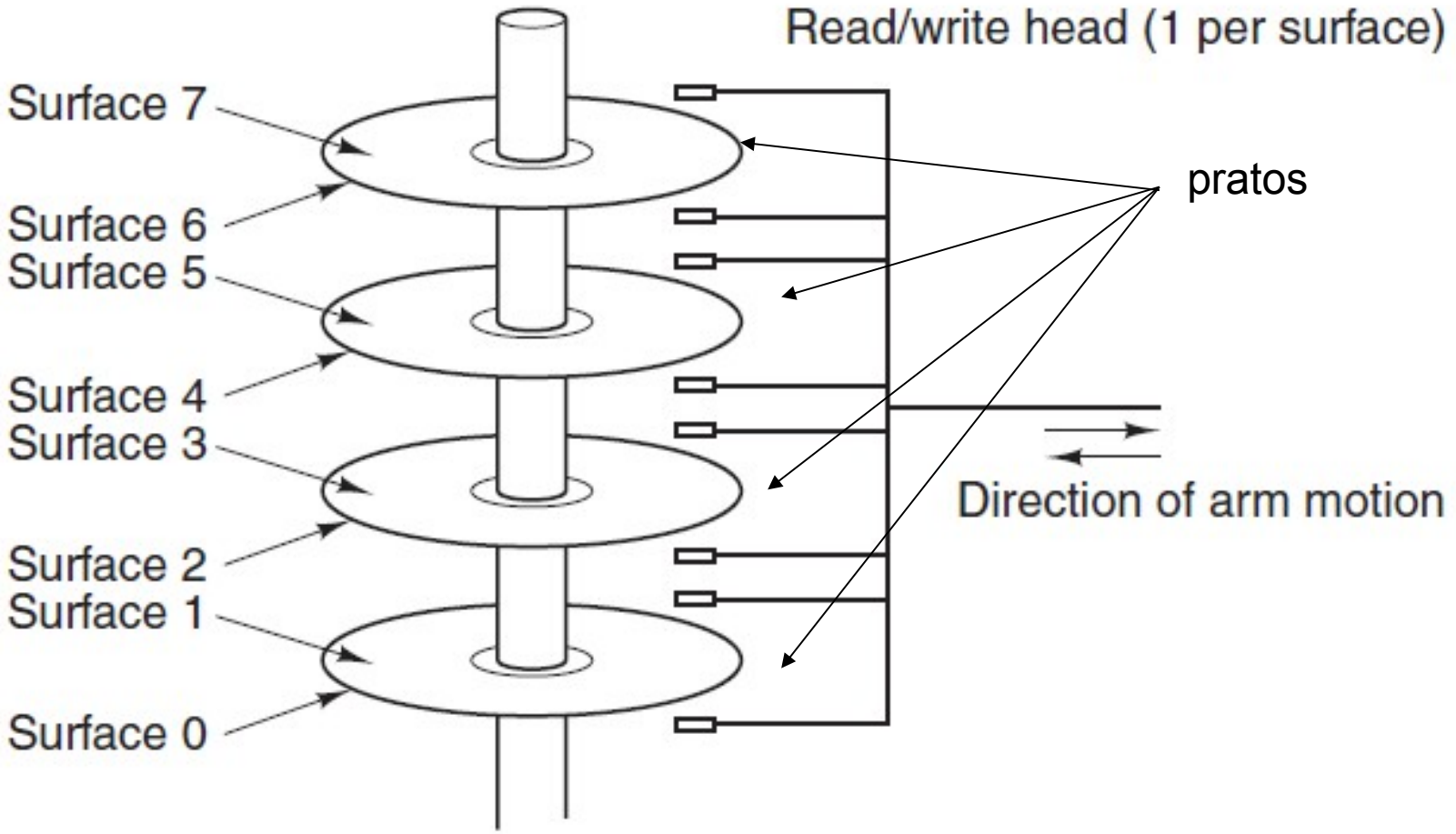
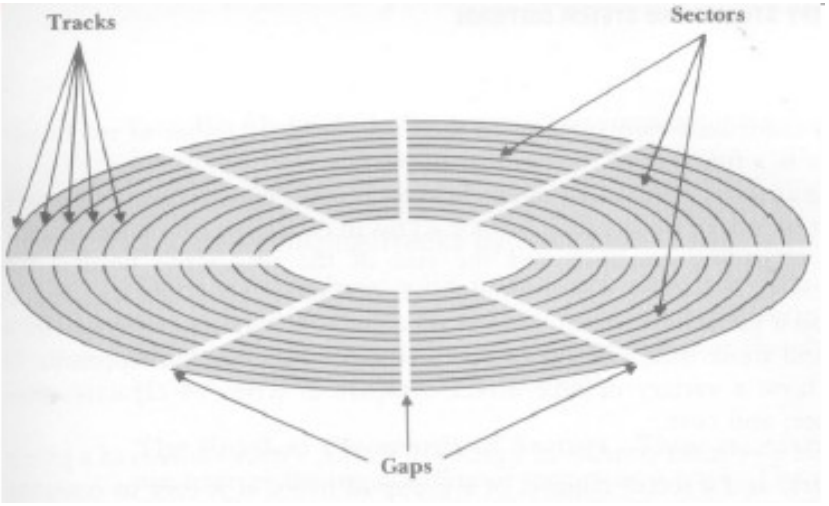


Figure 1-10. Structure of a disk drive.

(TANEMBAUM & BOS, 2015)



Organização da informação no disco

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

Um **setor** é a menor porção endereçável do disco

A divisão de uma trilha em setores é definida pelo disco, e não pode ser mudada.

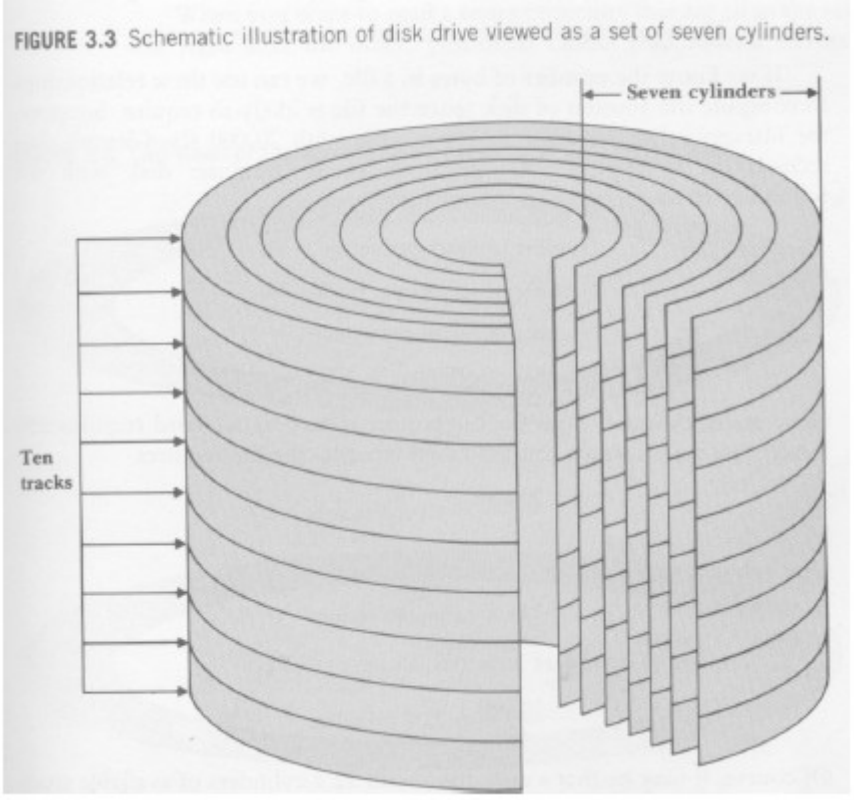
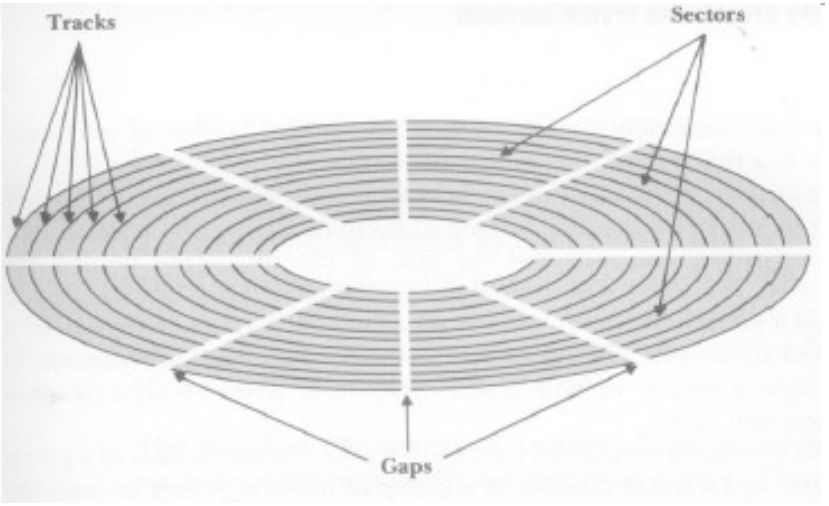


FIGURE 3.3 Schematic illustration of disk drive viewed as a set of seven cylinders.

O você acha que o computador deve fazer para ler algum dado do HD?



O você acha que o computador deve fazer para ler algum dado do HD?

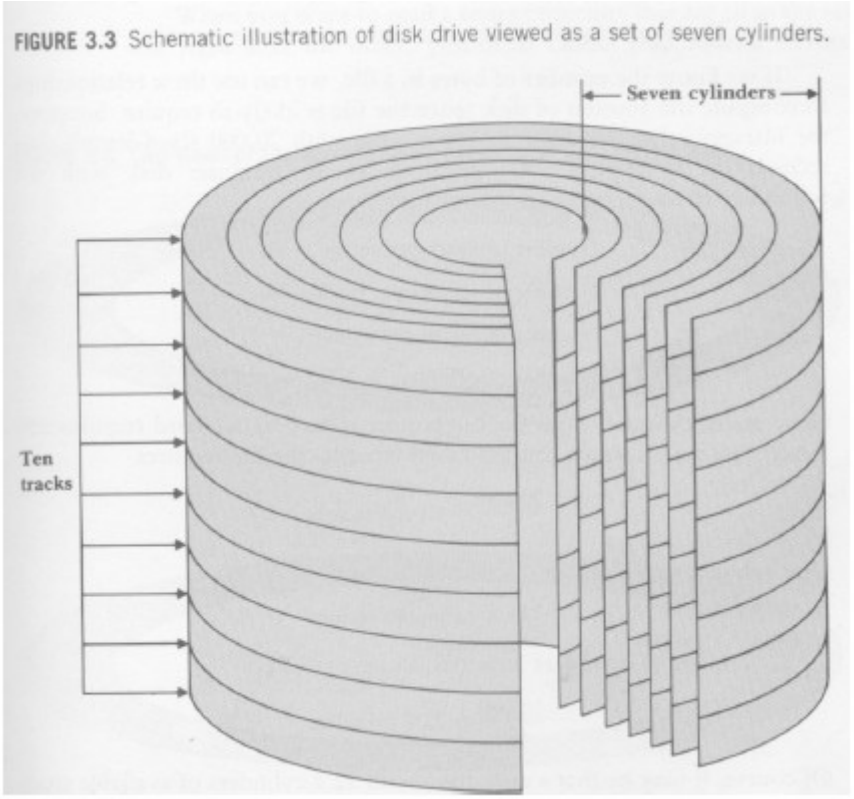
1) Identifica em que setor/trilha/superfície está a informação

2) Movimenta o braço de leitura para o cilindro correto (para poder acessa a trilha correta)

3) Rotaciona o prato para posicionar a cabeça de leitura sobre o setor correto

4) Faz a leitura de um certo nr de bytes

Passos 2 e 3 (chamados SEEK) são mecânicos! Por isso demora tanto!



Como calcular tempo de acesso

- Se acesso a disco é caro e queremos escolher estruturas de dados que diminuam o tempo de acesso, precisamos poder calcular (ou estimar) o tempo de acesso de uma forma não muito complicada, pelo menos:

  - independente da distribuição e localização do arquivo em disco

  - independente da tecnologia

- Para simplificar os cálculos, podemos fazer a seguinte aproximação (pior caso):

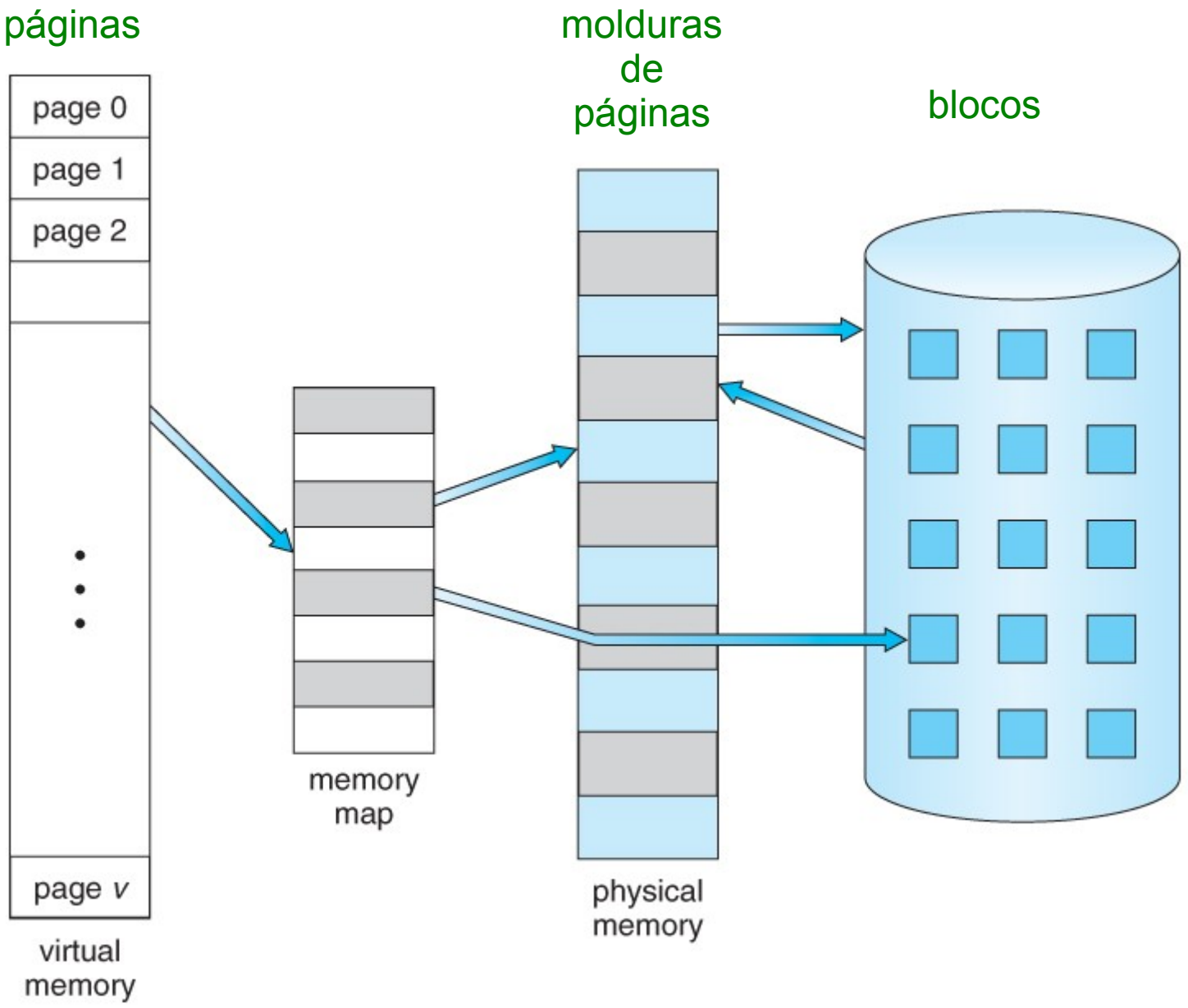
  - considera-se que é necessário um *seek* por "pedaço" de arquivo a ser lido:

    1) considerando que eu não preciso ler o arquivo inteiro em um dado instante, pois posso estar interessado em apenas um "pedaço"

    2) como há outros processos sendo executados, quando eu quiser ler outro "pedaço" do meu arquivo a cabeça de leitura do disco pode não estar no mesmo lugar onde parou a última leitura desse meu arquivo

  - **tempo de acesso total = nr de acessos (*seeks*) * tempo de um acesso**

De que tamanho tem que ser esse pedaço?
Pode ser um byte? E se meu programa quiser ler um byte de cada vez?
E quando eu leio um "pedaço", armazeno onde essa informação?



https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html

Aula de hoje

Memória virtual – mapeamento de endereços

Seja N o espaço de endereçamento (lógico – tudo o que eu quero endereçar) e M o espaço de memória principal (física)

f : N → M é um mapeamento podendo N >> M

Como fazer esse mapeamento de endereço lógico (do programa) a um endereço físico da memória principal?

Memória virtual – mapeamento de endereços

Lembrando que pode |N| >> |M|

Um endereço de N (lógico, virtual): parte dos bits é o **número** da página (p) e a outra parte é a posição (*offset*) dentro da página (b)

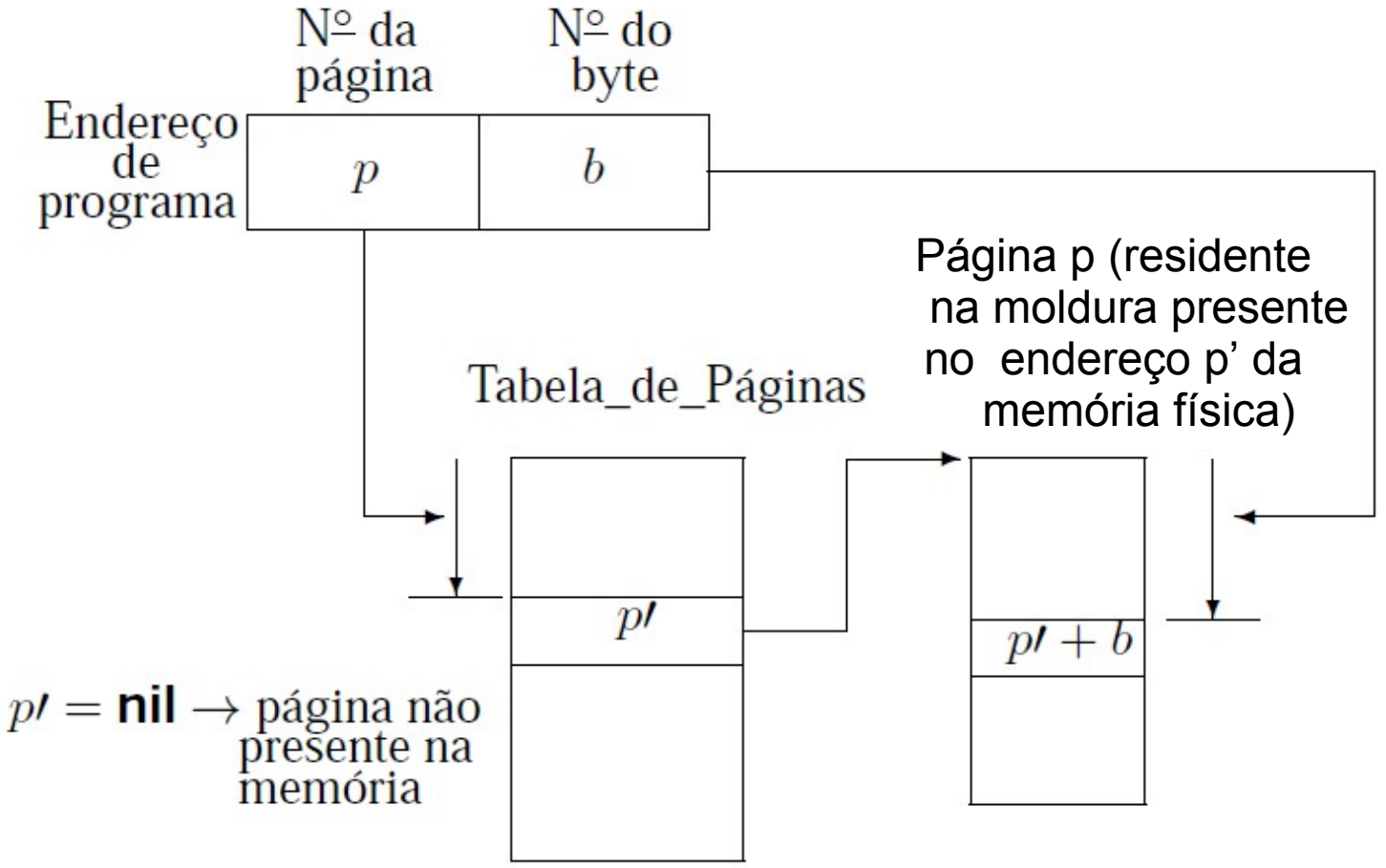
Lembrando que pode haver tantas páginas quanto forem necessárias para cobrir N

Tabela de Páginas: se a página de **número** p estiver na memória principal, a p-ésima entrada da Tabela de Páginas contém o **endereço da moldura** p' (na memória principal) que contém a página p

Isto é, um endereço n pertencente a N é pb (p concatenado com b) e o mapeamento f é:

f(n) = f(pb) = PageTable[p] + b = p' + b

Memória virtual – mapeamento de endereços



Memória Virtual: Reposição de Páginas

- Se não houver uma moldura de página vazia $\rightarrow$ uma página deverá ser removida da memória principal.
- Ideal $\rightarrow$ remover a página que não será referenciada pelo período de tempo mais longo no futuro.
  - tentamos inferir o futuro a partir do comportamento passado.

Memória Virtual: Políticas de Reposição de Páginas

- **Menos Recentemente Utilizada (LRU):**
  - um dos algoritmos mais utilizados,
  - remove a página menos recentemente utilizada,
  - parte do princípio que o comportamento futuro deve seguir o passado recente.
- **Menos Freqüentemente Utilizada (LFU):**
  - remove a página menos feqüentemente utilizada,
  - inconveniente: uma página recentemente trazida da memória secundária tem um baixo número de acessos e pode ser removida.
- **Ordem de Chegada (FIFO):**
  - remove a página que está residente há mais tempo,
  - algoritmo mais simples e barato de manter,
  - desvantagem: ignora o fato de que a página mais antiga pode ser a mais referenciada.

Organização de arquivos na memória secundária

- **Bloco**: unidade de transferência de dados entre memória principal e a memória secundária
  - Obs: Em sistema gerenciador de banco de dados (SGBD), um bloco é formado uma ou mais páginas (definido nos parâmetros de configurador de um SGBD)
  - Usaremos aqui o termo bloco de forma genérica, a não ser quando especificado (podemos simplificar aqui que um bloco tem o tamanho de uma página)

Cabeçalhos de arquivos

- Cabeçalho do arquivo (descritor) pode conter informações como:
  - Descrição dos formatos dos campos de um registro
  - Códigos de tipos de registros para registros de tamanho variável
  - Primeiro e último bloco
  - Informações para determinar os endereços dos seus blocos
- Abrir um arquivo significa trazer para a memória o cabeçalho do arquivo (blocos contendo essas informações que ficarão em memória até o arquivo ser fechado)

Organização de arquivos na memória secundária

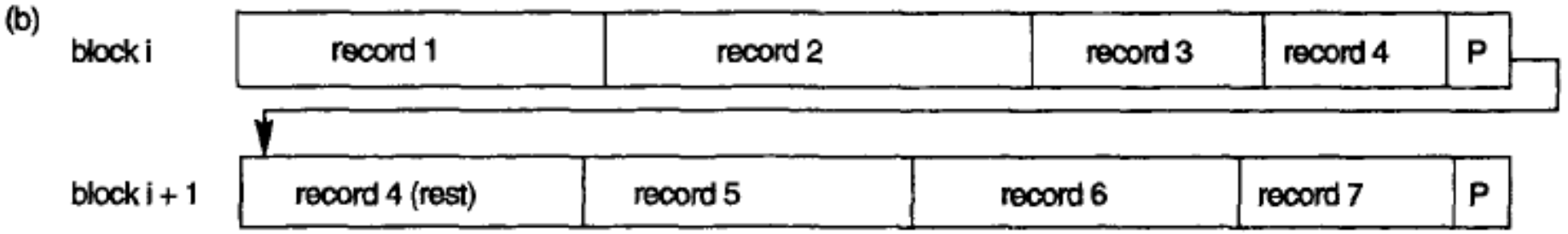
- É comum considerar que em cada bloco pode haver vários registros
- Se R (tamanho fixo do registro, para simplificar) e B (tamanho do bloco) e R ≤ B:

  fator de blocagem fb = floor(B/R)

  = número de registros inteiros que cabem em um bloco

Organização de arquivos na memória secundária

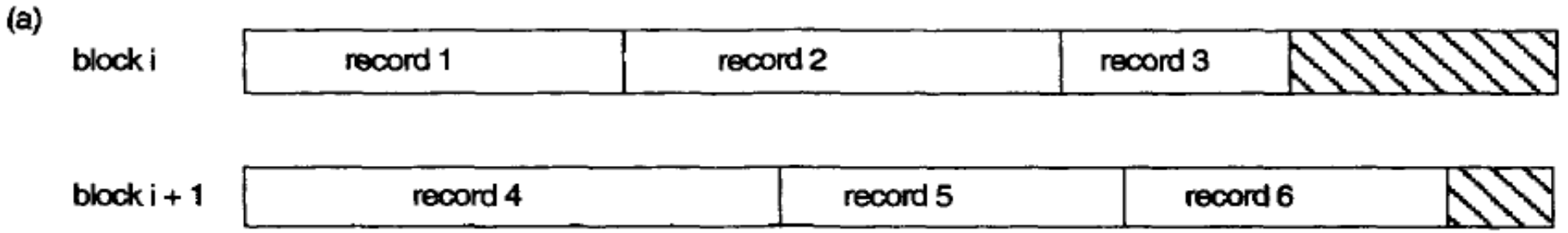
- **Organização *espalhada***: os blocos são totalmente preenchidos; se um registro não cabe inteiramente na parte vazia do bloco, coloca o que couber e um ponteiro para o próximo bloco



(ELSMARI & NATATHE)

Organização de arquivos na memória secundária

- **Organização não *espalhada***: registros não podem ser divididos. Cada bloco pode conter até fb registros.



(ELSMARI & NATATHE)

Organização de arquivos na memória secundária

- **Organização não *espalhada***: registros não podem ser divididos. Cada bloco pode conter até fb registros.
  - Se os registros tiverem tamanho fixo = R, blocos de tamanho B e taxa de blocagem = fb:

  Perda de espaço em cada bloco:

Organização de arquivos na memória secundária

- **Organização não *espalhada***: registros não podem ser divididos. Cada bloco pode conter até fb registros.
  - Se os registros tiverem tamanho fixo = R, blocos de tamanho B e taxa de blocagem = fb:

  Perda de espaço em cada bloco: B - (fb*R)

Alocação de blocos

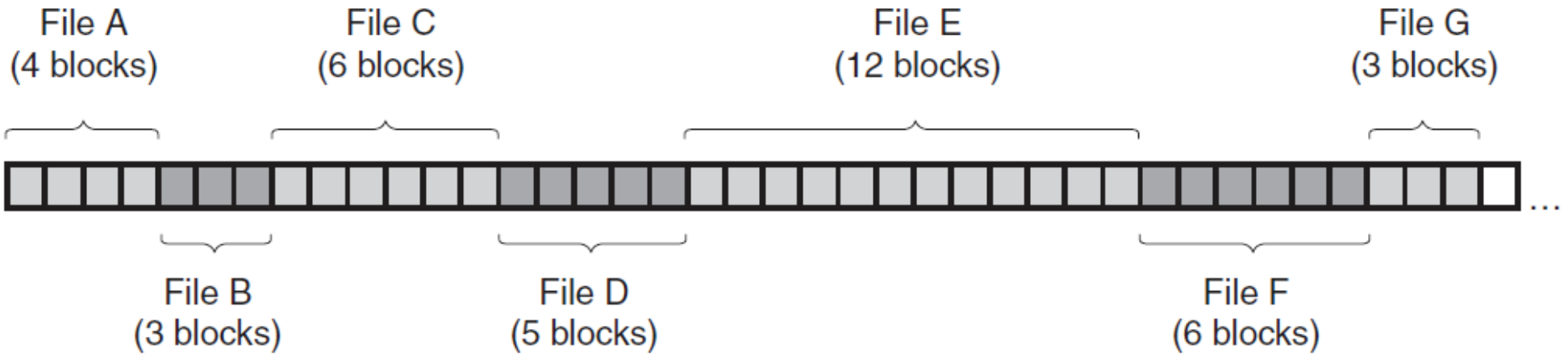
- Leitura, escrita, buscas, etc., são realizadas por blocos.
- Os arquivos não são estáticos, eles crescem e diminuem
- Estratégias de alocação de blocos no disco devem considerar esse fato
- Vamos estudar várias estratégias até o fim do semestre
- Para cada estratégia analisaremos complexidade de leitura sequencial, leitura aleatória (busca), inserção e remoção de registros
  - Complexidade em termos de número de ...

Alocação de blocos

- Leitura, escrita, buscas, etc., são realizadas por blocos.
- Os arquivos não são estáticos, eles crescem e diminuem
- Estratégias de alocação de blocos no disco devem considerar esse fato
- Vamos estudar várias estratégias até o fim do semestre
- Para cada estratégia analisaremos complexidade de leitura sequencial, leitura aleatória (busca), inserção e remoção de registros
  - Complexidade em termos de número de seeks (estimado no pior caso pelo número de blocos a serem lidos)

Alocação sequencial

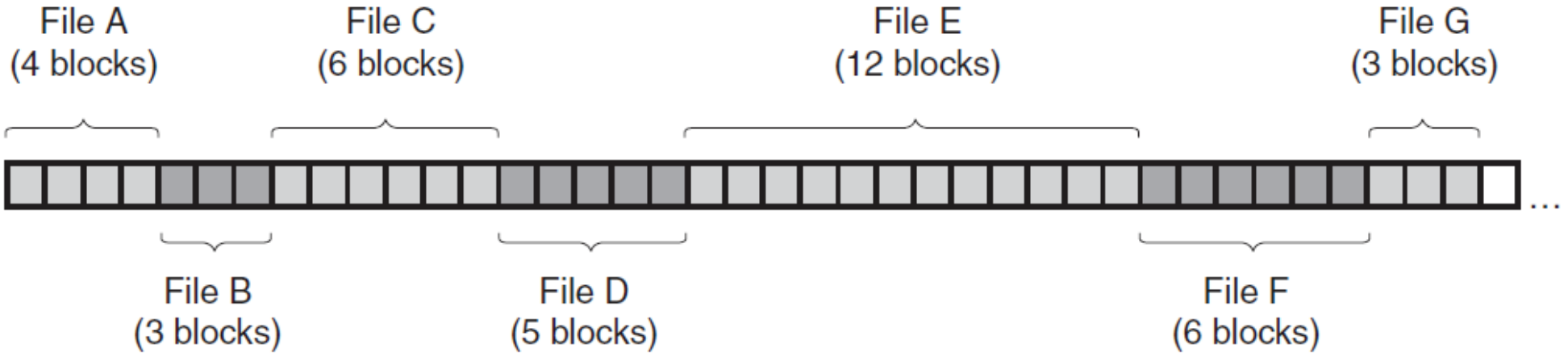
- Blocos alocados sequencialmente no disco (pelos cilindros)



- Vantagens e desvantagens?

Alocação sequencial

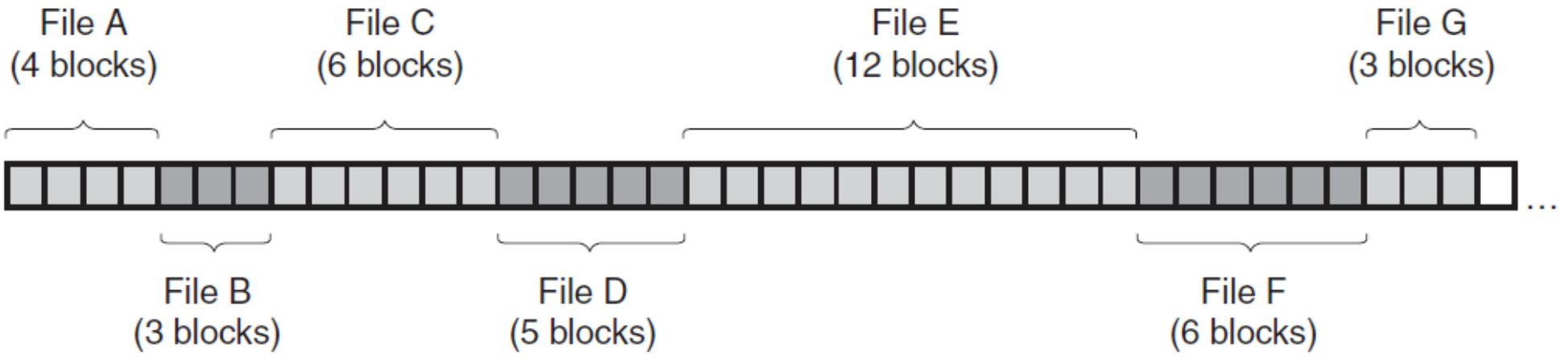
- Blocos alocados sequencialmente no disco (pelos cilindros)



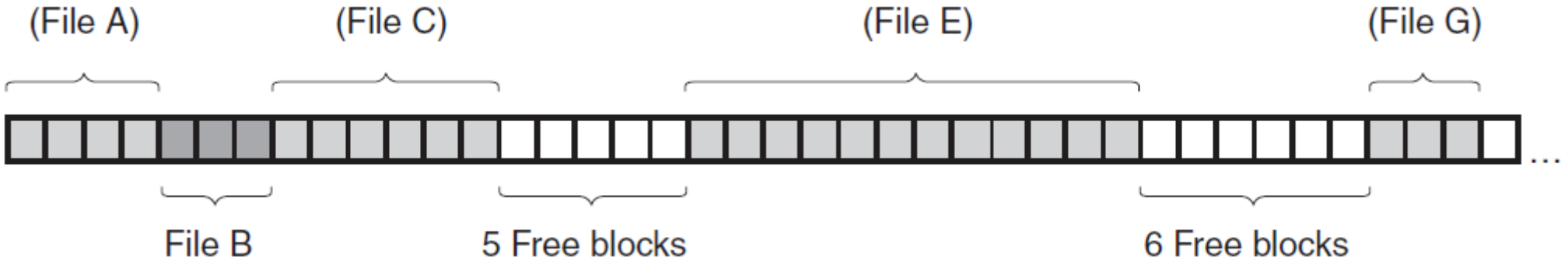
- Vantagens e desvantagens?
  - Leitura fácil (leitura sequencial é ótima, e na leitura aleatória depende da facilidade de localização do deslocamento do registro dentro do arquivo)
  - Expansão complicada: se não houver espaço disponível até o próximo arquivo tem que ser removido para outro local
  - Fragmentação externa (buracos entre os arquivos): maior ou menor dependendo da política de alocação

Alocação sequencial

- Blocos alocados sequencialmente no disco (pelos cilindros)



- Após algumas remoções

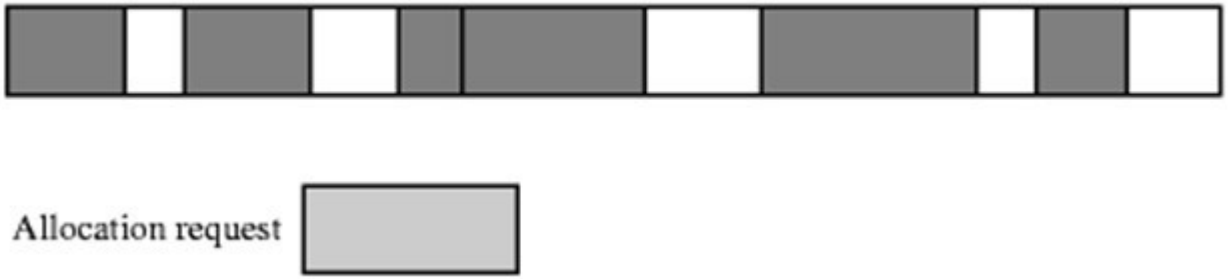


Fonte: (TANEMBAUM, 2015)

Alocação sequencial

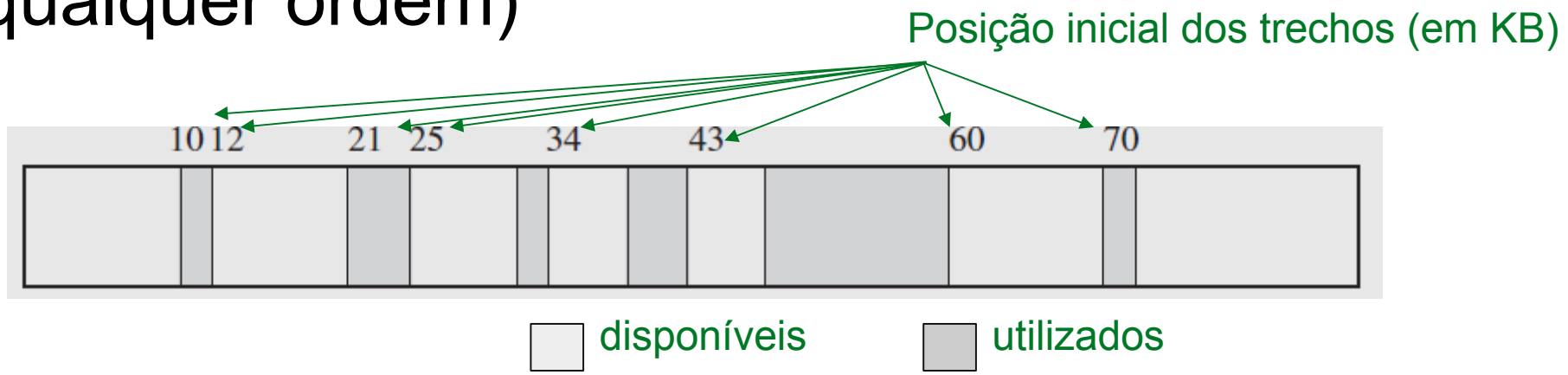
Fragmentação externa:

- Com o tempo (após alocações/desalocações sucessivas), o disco pode ficar fragmentado, isto é, com vários trechos disponíveis intercortados por trechos utilizados



Alocação sequencial - Métodos de ajuste sequencial

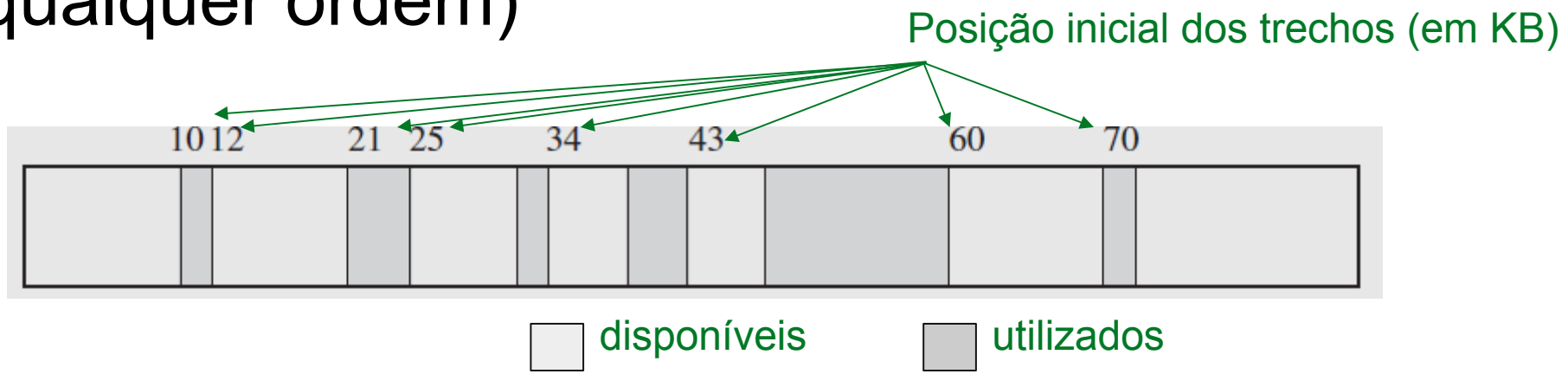
Vetor sequencial contendo trechos contíguos de memória utilizada e disponível (intercalados em qualquer ordem)



Se um trecho de tamanho R for requisitado (por exemplo 8 KB), onde ele será alocado?

Alocação sequencial - Métodos de ajuste sequencial

Vetor sequencial contendo trechos contíguos de memória utilizada e disponível (intercalados em qualquer ordem)



Se um trecho de tamanho R for requisitado (por exemplo 8 KB), onde ele será alocado?

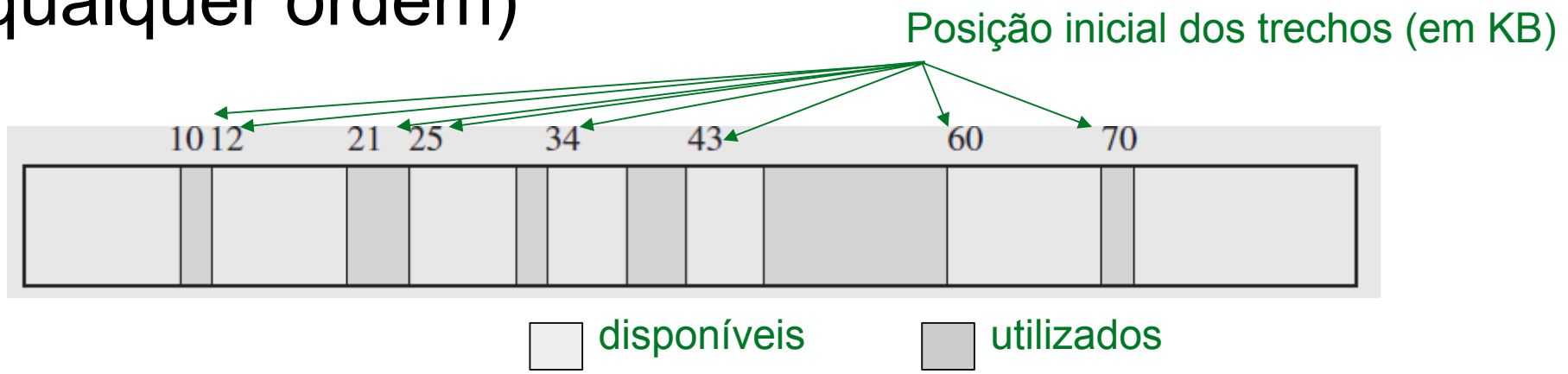
Depende de qual variação do método de ajuste sequencial está sendo utilizado...

Métodos de ajuste sequencial

- Primeiro ajuste: seleciona o primeiro trecho encontrado (a partir do início da lista) grande o suficiente
- Próximo ajuste: seleciona o próximo trecho grande o suficiente (a partir do índice "corrente", ajustado após a última alocação)
- Melhor ajuste: seleciona o menor trecho dentre os trechos grandes o suficiente
- Pior ajuste: seleciona o maior trecho de todos

Métodos de ajuste sequencial

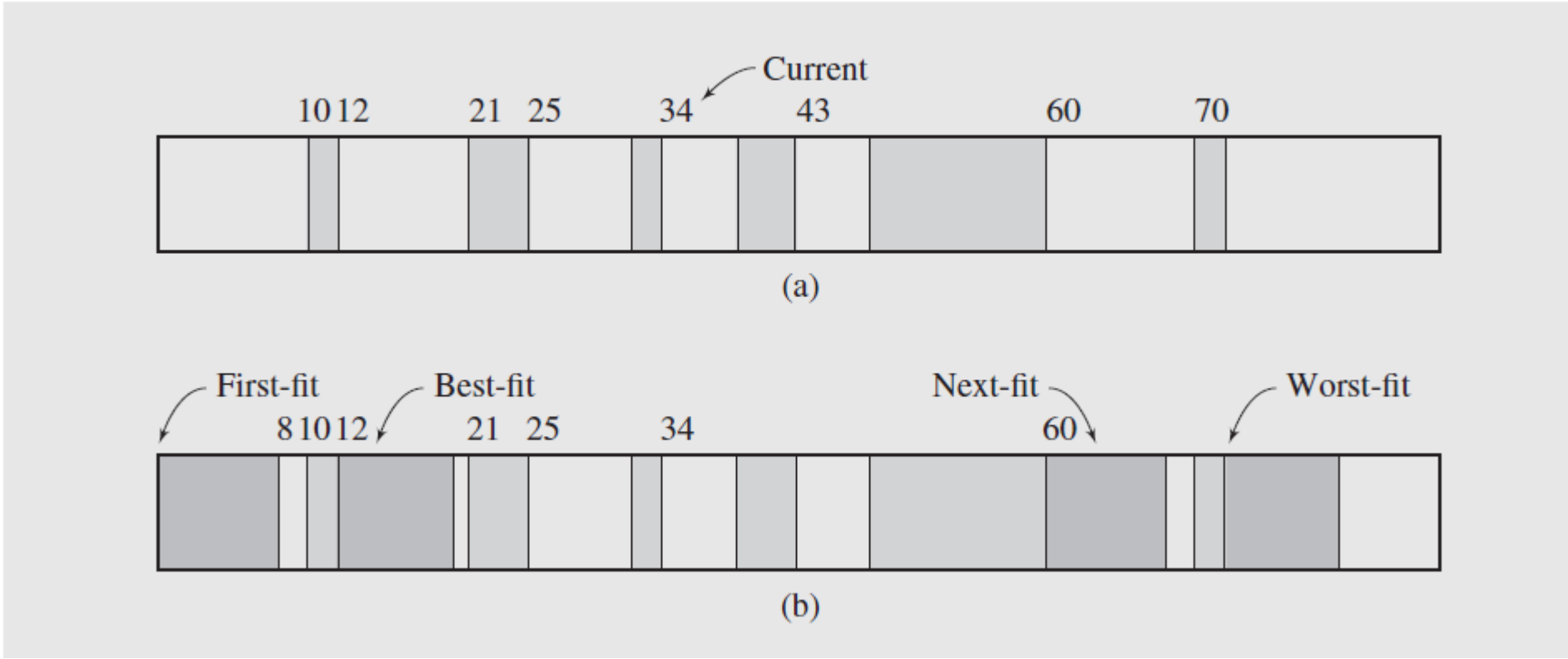
Vetor sequencial contendo trechos contíguos de memória utilizada e disponível (intercalados em qualquer ordem)



Se um trecho de tamanho R for requisitado (por exemplo 8 KB), onde ele será alocado?

Métodos de ajuste sequencial variações

FIGURE 12.1 Memory allocation using sequential-fit methods.



Métodos de ajuste sequencial: variações

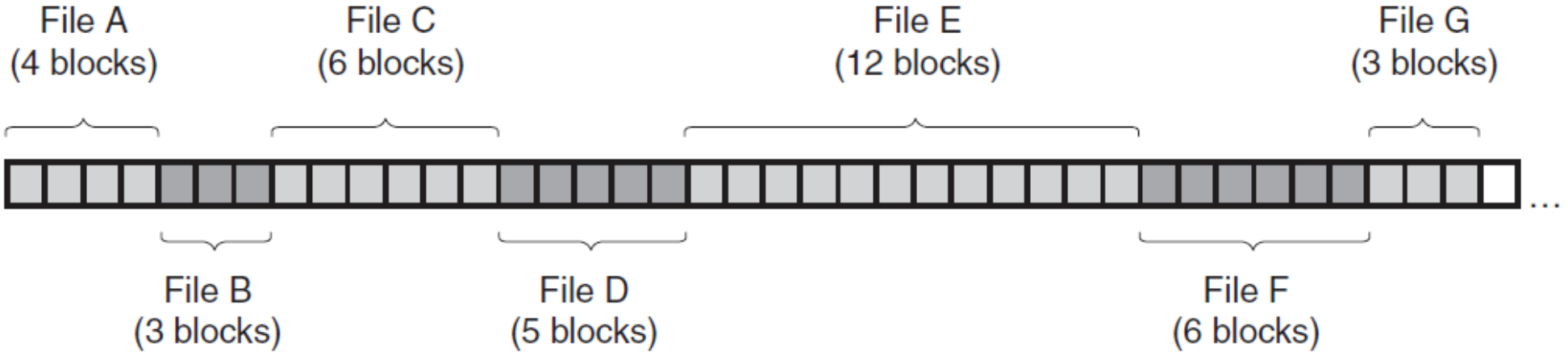
- Qual vocês acham que é melhor?

Métodos de ajuste sequencial: variações

- Qual vocês acham que é melhor?
- Melhor ajuste pode ser o pior: gasta tempo analisando tudo e, a menos que o ajuste seja perfeito, deixa sobrar normalmente trechos pequenos que não podem ser reutilizados
- Pior ajuste tenta evitar esse desperdício, deixando sobrar trechos maiores que podem ser ainda utilizados, e assim posterga a criação de blocos pequenos
- Primeiro ajuste é o mais eficiente: balanço entre tempo de achar um bloco de tamanho suficiente (retorna assim que achar o primeiro) e fragmentação (não deixa sobrar sistematicamente o menor ou maior trecho)
- Próximo ajuste, similar ao primeiro ajuste, mas chega mais rápido ao fim do heap. Nesse caso:
  - Ou gastará mais tempo tendo que rever todo o heap para ver se há trecho disponível
  - Ou considera-se que acabou a memória, e portanto pode ter fragmentado mais.

Alocação sequencial

- Blocos alocados sequencialmente no disco (pelos cilindros)



- Pode ter seus registros ordenados por uma chave de ordenação (sorted files) ou não (heap files)
- Vantagens e desvantagens de cada um?

Alocação sequencial (não ordenado)

- O arquivo, de r registros espalhados em b blocos, não está ordenado nem indexado
- **Inserção** : no final do arquivo. Há espaço disponível (dentro do último bloco ou após ele até o próximo arquivo)?
  - SIM: Eficiente: O(1)
    - Copia último bloco no buffer de memória
    - Insere registro
    - Reescreve bloco no disco
  - NÃO: Ineficiente: O(b) – tem que realocar todo o arquivo em outro lugar no disco

Alocação sequencial (não ordenado)

- **Busca**: sequencial não ordenada
  - Tenho que olhar todos os blocos – O(b)
  - Lembrando que a busca é por registros, não por blocos
- **Remoção**:
  - Precisa achar onde está o registro – O(b)
  - Exclui registro do bloco (que está no buffer) ou resetar bit para inválido
  - Reescrever bloco de volta ao disco (com um espaço vazio) – O(1)

    => Demanda uma reorganização periódica

Alocação sequencial (não ordenado)

- **Modificação de registro de tamanho variável:**
  - Pode exigir a remoção de outros registros a serem inseridos em outros blocos

- **Leitura ordenada**: MUITO INEFICIENTE
  - Exige uma ordenação primeiro! (exigirá ordenação externa se o arquivo inteiro não couber na memória)

Referências

- Slides da Profa. Graça (ICMC) - http://wiki.icmc.usp.br/index.php/SCC-203_(gracan) (Arquivos 8, 9 e 12)

- Slides do cap 6 do Ziviani

- GOODRICH et al, **Data Structures and Algorithms in C++.** Ed. John Wiley & Sons, Inc. 2nd ed. 2011. Seção 14.2

- ELMASRI, R.; NAVATHE, S. B. **Fundamentals of Database Systems**. 4 ed. Ed. Pearson-Addison Wesley. Cap 13 (até a seção 13.7).

- TANEMBAUM, A. S. & BOS, H. **Modern Operating Systems**. Pearson, 4th ed. 2015. Cap 4

- RAMAKRISHNAN & GEHRKE. **Data Management Systems**. 3ed. McGrawHill. 2003 Cap 8 e 9.