Algoritmos e Estruturas de Dados II
Aula 21 – Alocação Indexada

Prof. Luciano A. Digiampietri
digiampietri@usp.br
@digiampietri

2025

Alocação de Blocos na Memória Secundária

- Organização Interna de Arquivos
- Acesso à Memória por Blocos (*seeks*)
- Estratégias de alocação de blocos no disco e organização de registros pelos blocos devem considerar esse fato:
  - Sequencial não ordenado (*heap files*)
  - Sequencial ordenado (*sorted files*)
  - Por listas ligadas (ordenação pelas chaves)
  - Indexada
  - Árvores B / B+
  - *Hashing*

Alocação Indexada

- Lista ligada (**assunto da última aula**) aproveita espaço (resolve fragmentação externa), mas leitura aleatória fica horrível (sem tabela de alocação)

Alocação Indexada

- Lista ligada (**assunto da última aula**) aproveita espaço (resolve fragmentação externa), mas leitura aleatória fica horrível (sem tabela de alocação)
- Tabela de alocação acelera a leitura aleatória mas gasta muita memória

Alocação Indexada

- Lista ligada (**assunto da última aula**) aproveita espaço (resolve fragmentação externa), mas leitura aleatória fica horrível (sem tabela de alocação)
- Tabela de alocação acelera a leitura aleatória mas gasta muita memória
- Como diminuir esse último problema?

Alocação Indexada

- Lista ligada (**assunto da última aula**) aproveita espaço (resolve fragmentação externa), mas leitura aleatória fica horrível (sem tabela de alocação)
- Tabela de alocação acelera a leitura aleatória mas gasta muita memória
- Como diminuir esse último problema?
- Por que manter em memória as informações de arquivos que não foram abertos? Que tal "uma tabela" por arquivo?

Alocação Indexada

Alocação Indexada:

- Um ou mais blocos de índices contém ponteiros para os blocos de fato
- Blocos de índices são como uma tabela de alocação específica daquele arquivo
  - i-ésima entrada do primeiro bloco de índice contém o número do i-ésimo bloco de dado do arquivo
- Blocos de índice carregados na memória sob demanda (assim como arquivos de dados)

Alocação Indexada

- Cada arquivo é uma lista ligada de blocos

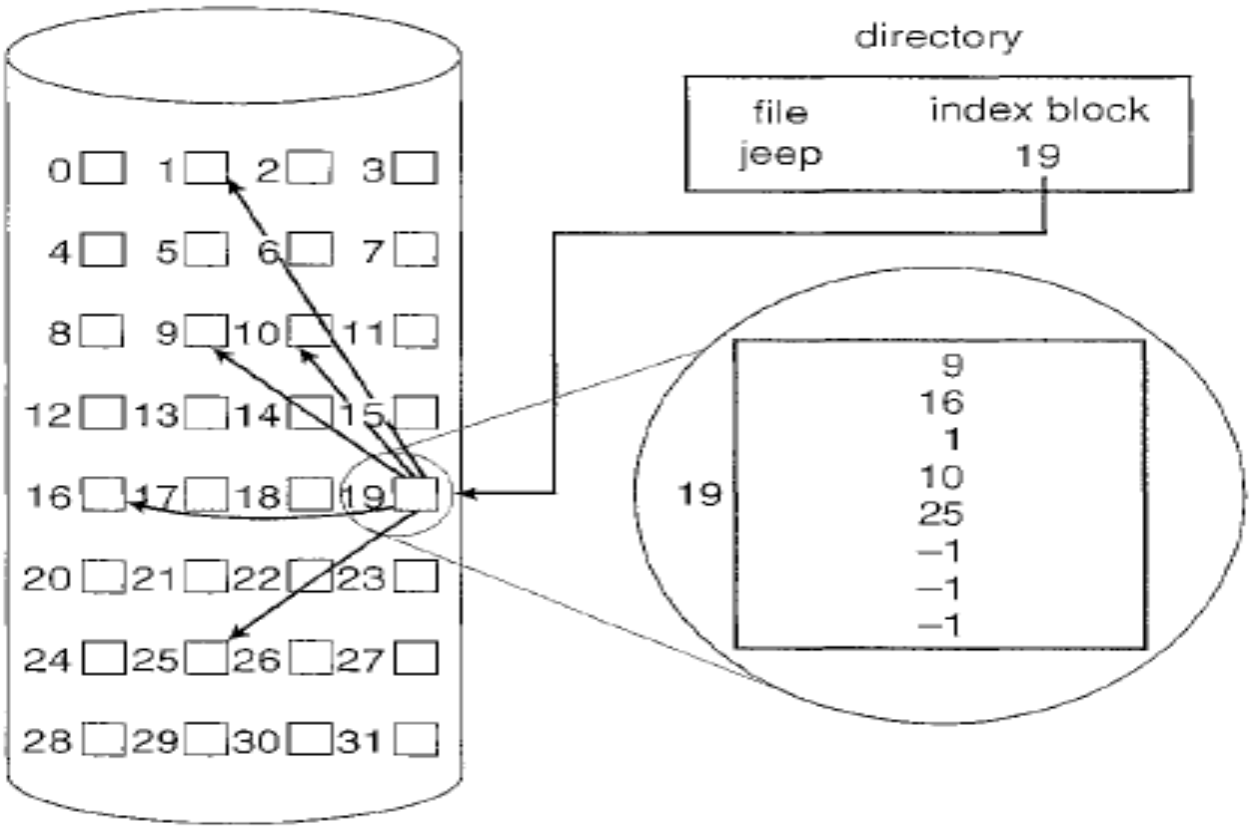
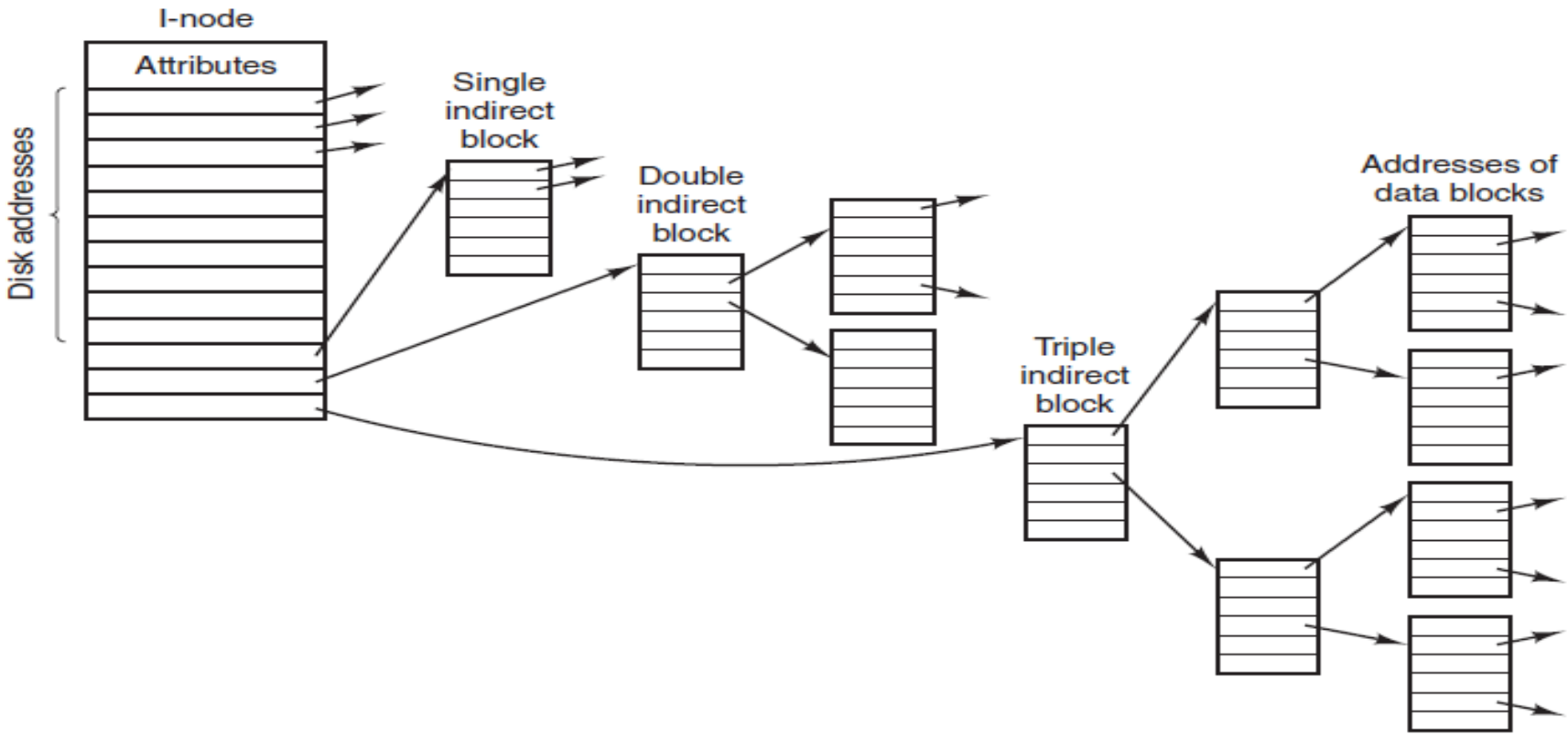


Figure 11.8 Indexed allocation of disk space.

(SILBERCHATZ et al, 2009)

Alocação Indexada



(TANEMBAUM, 2015)

É capaz de lidar com arquivos gigantes.

Leitura Complementar

- Mais detalhes sobre o sistema de arquivos do Linux e Windows: cap 4, 10 e 11 do livro do Tanenbaum (referência no último slide)

Índices

Índices podem ser utilizados para:

- Organização física dos arquivos
- Estruturar caminhos secundários de acesso aos dados, de forma a acelerar buscas

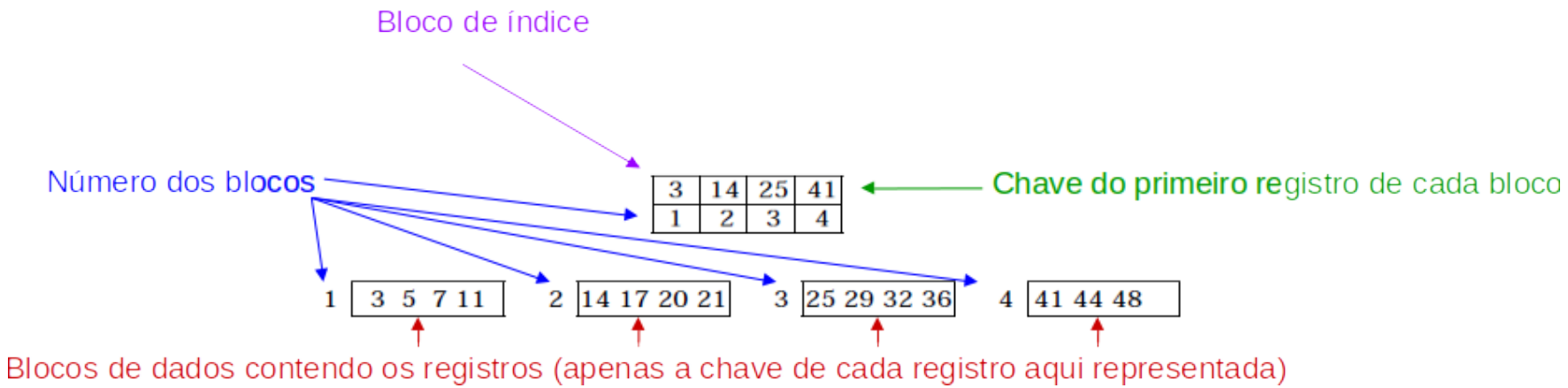
Índices

Índices podem ser utilizados para:

- Organização física dos arquivos
- Estruturar caminhos secundários de acesso aos dados, de forma a acelerar buscas

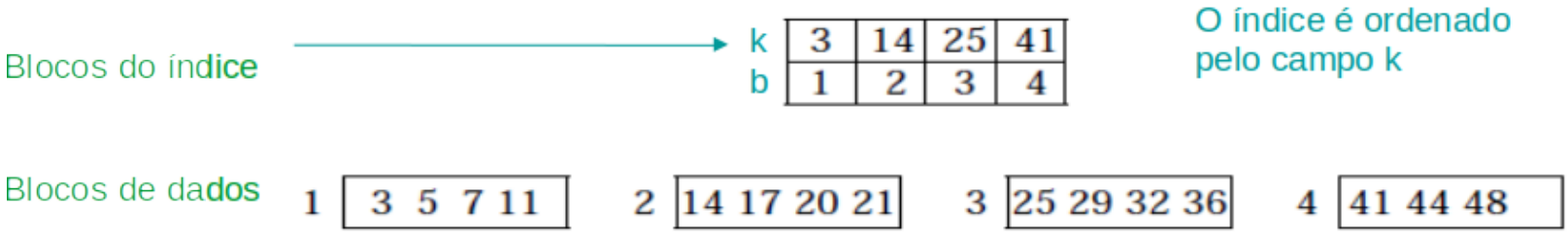
Alocação Indexada - Arquivos Ordenados

- Os blocos de índices possuem, além dos ponteiros para os blocos de dados, a chave do primeiro registro de cada bloco de dado



Alocação Indexada - Arquivos Ordenados

- Índice primário: arquivo ORDENADO de registros de tamanho fixo contendo os campos $< k, b >$, sendo k a chave (primária) do primeiro registro (âncora) do bloco b



Alocação Indexada - Arquivos Ordenados

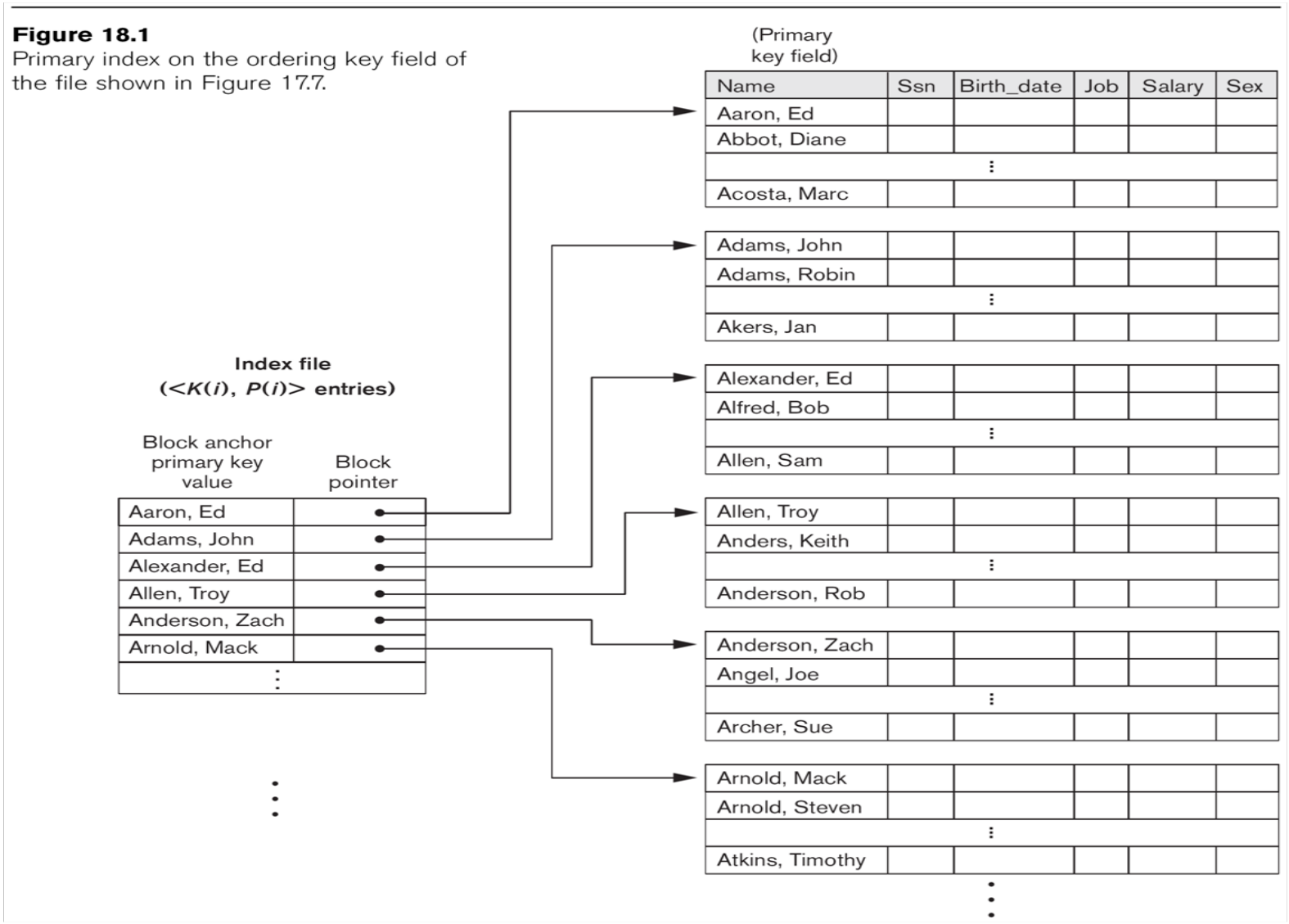
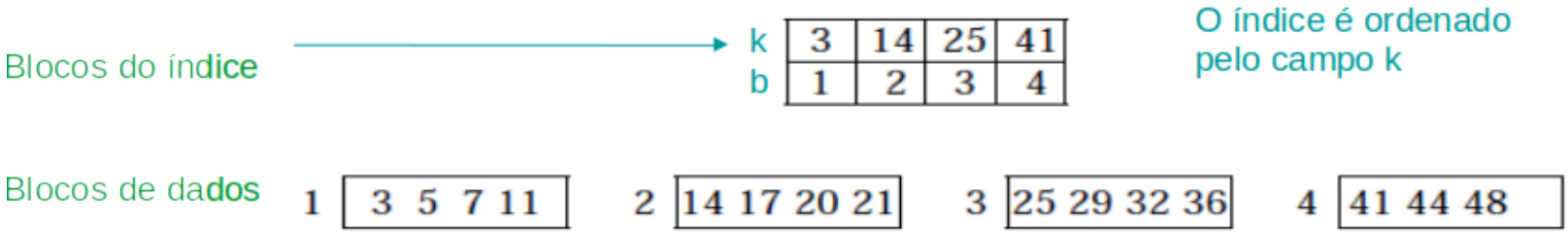


Figure 18.1
Primary index on the ordering key field of the file shown in Figure 17.7.

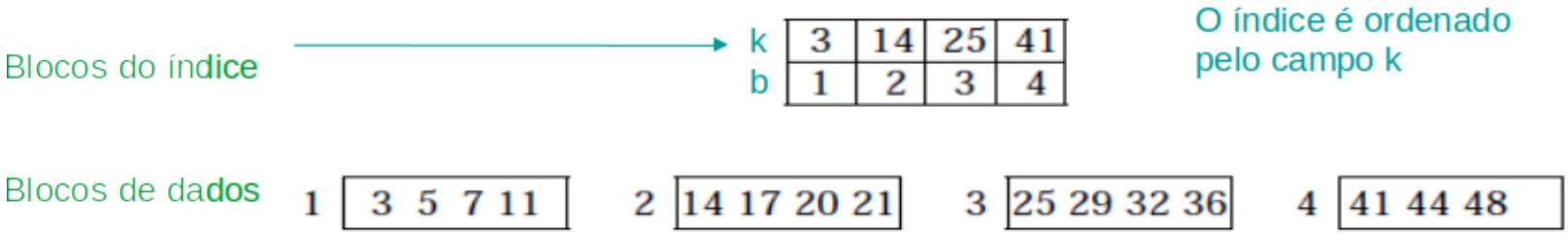
Alocação Indexada - Arquivos Ordenados

- Índice primário: arquivo ORDENADO de registros de tamanho fixo contendo os campos $< k, b >$, sendo k a chave (primária) do primeiro registro (âncora) do bloco b



Alocação Indexada - Arquivos Ordenados

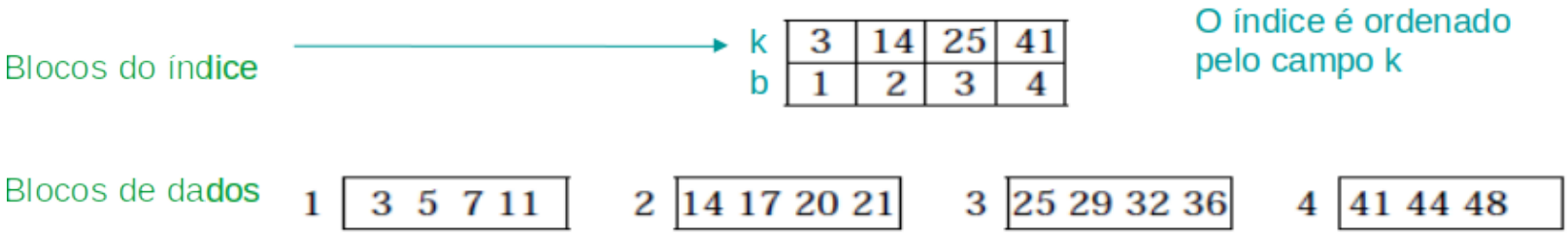
- Índice primário: arquivo ORDENADO de registros de tamanho fixo contendo os campos $< k, b >$, sendo k a chave (primária) do primeiro registro (âncora) do bloco b



- **Qual a Vantagem?**
  - Índice tem bi blocos, sendo bi << B (B = número de blocos de dados, no exemplo acima, bi = 1 e B = 4) Busca Binária no índice é muito mais rápida: O(log(bi))

Alocação Indexada - Arquivos Ordenados

- Índice primário: arquivo ORDENADO de registros de tamanho fixo contendo os campos $< k, b >$, sendo k a chave (primária) do primeiro registro (âncora) do bloco b



- **Observação:**
  - Este é um índice esparso
  - Se fosse denso, haveria uma entrada por registro

Alocação Indexada - Arquivos Ordenados

Inserção / Remoção

- Potencialmente altera a posição em disco de vários registros: $O(b + bi) = O(b)$
  - Altera âncora de vários blocos
  - Altera várias entradas do índice primário

Alocação Indexada - Arquivos Ordenados

Inserção / Remoção

- Potencialmente altera a posição em disco de vários registros: $O(b + bi) = O(b)$
  - Altera âncora de vários blocos
  - Altera várias entradas do índice primário
- Formas de contornar o problema:
  - Remoção por bits de validade
  - Usar um arquivo desordenado de *overflow* (para as inserções, se necessário)
    - Ou uma lista ligada de overflow (os registros do bloco e da lista podem ser ordenados para melhorar a busca)
  - Reorganizações periódicas são necessárias

Índice de Clustering

E se o campo de ordenação (chave) não tiver valores únicos?

Índice de Clustering

E se o campo de ordenação (chave) não tiver valores únicos?

- Índice de clustering: arquivo ORDENADO de registros de tamanho fixo contendo os campos $< c, b >$, sendo c um campo de classificação física que não possui valores distintos

Índice de Clustering

E se o campo de ordenação (chave) não tiver valores únicos?

- Índice de clustering: arquivo ORDENADO de registros de tamanho fixo contendo os campos $< c, b >$, sendo c um campo de classificação física que não possui valores distintos
  - Uma entrada $< c, b >$ para cada valor distinto de c, sendo b o primeiro bloco da primeira ocorrência da chave com valor c

Índice de Clustering

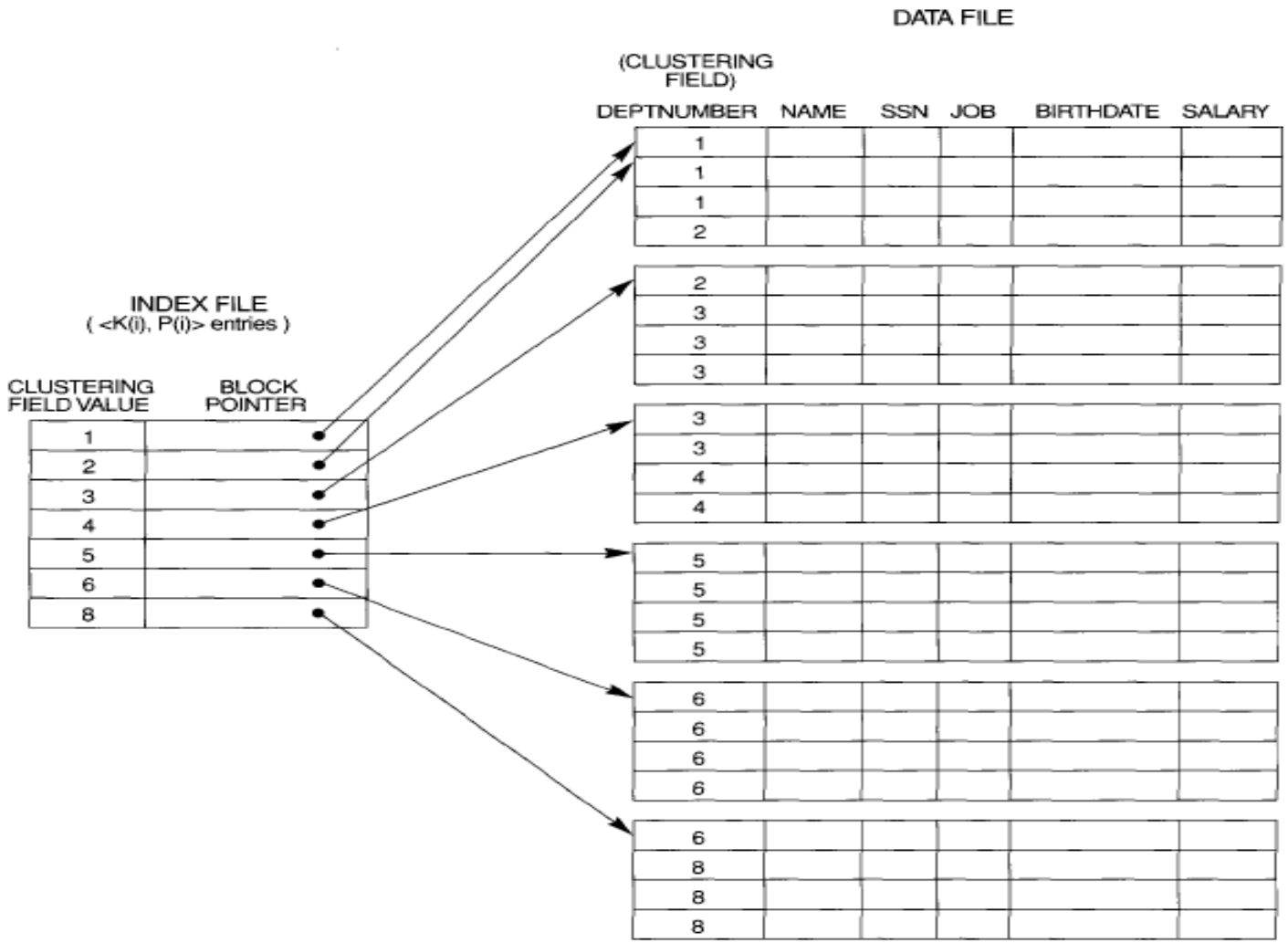


FIGURE 14.2 A clustering index on the DEPTNUMBER ordering nonkey field of an EMPLOYEE file.

Elmasri & Navathe, 2004

Índice de Clustering

- **Inserção / remoção**: ainda problemáticas, pois c ordena fisicamente os registros

Índice de Clustering

- **Inserção / remoção**: ainda problemáticas, pois c ordena fisicamente os registros
  - Alternativas:
  - Reservar um ou mais blocos para cada valor de c (ligados por ponteiros)
  - Remoção por bit de validade

Índice de Clustering

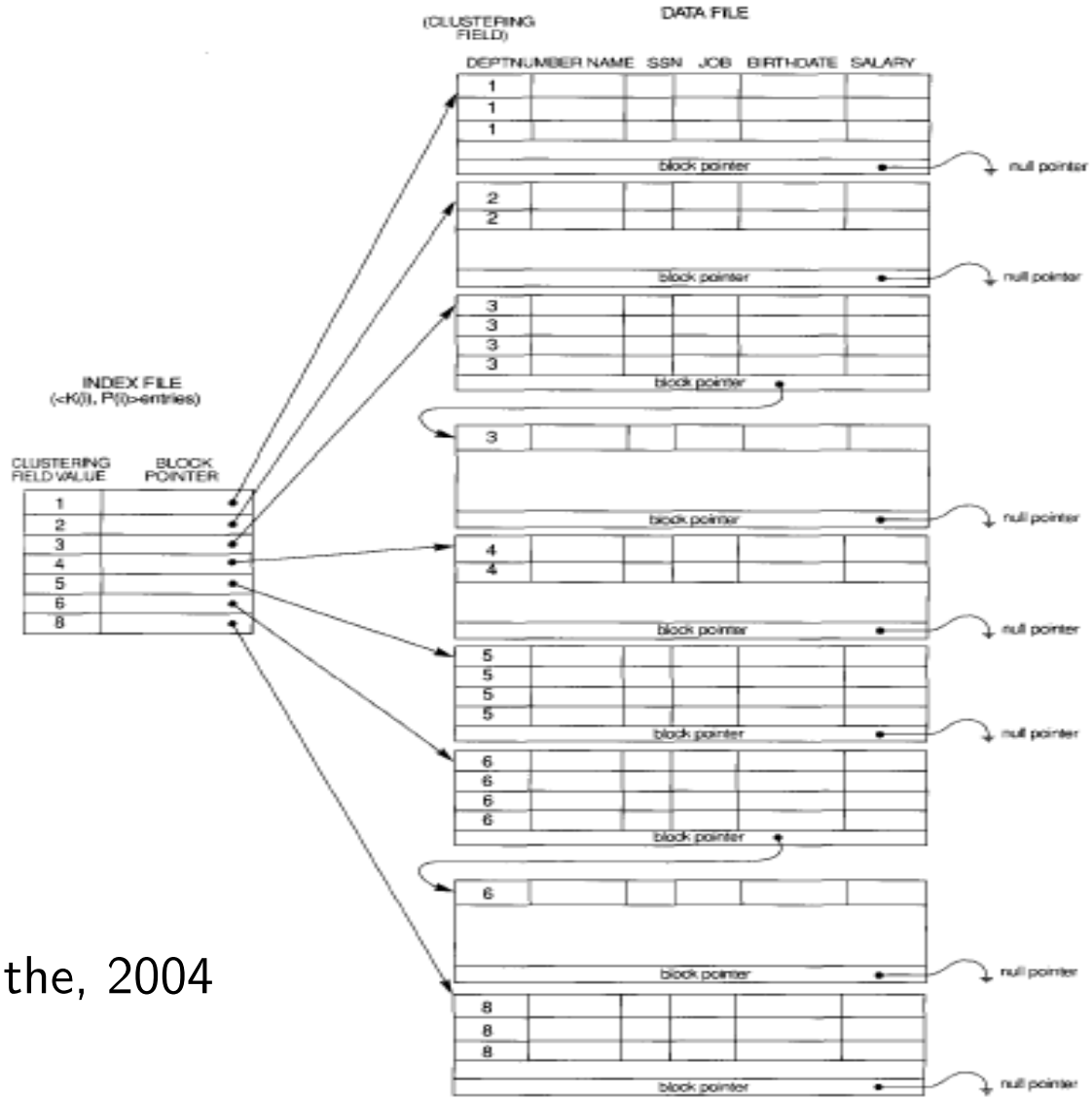


FIGURE 14.3 Clustering index with a separate block cluster for each group of records that share the same value for the clustering field.

Elmasri & Navathe, 2004

Índice de Clustering

- **Busca**:

Índice de Clustering

- **Busca**:
  - Binária nos blocos de índice
  - Busca sem sucesso: não acessa bloco de dado
  - Busca com sucesso: quer só o primeiro registro ou todos?
    - Primeiro: um bloco de dado (informado no índice)
    - Todos: tem que ler q blocos (todos daquele valor de chave)
  - Quanto maior o número de valores distintos, maior o tempo de busca binária nos índices

Índices Primários, de Clustering ...

- Índices primários e de clustering são baseados no campo de ordenação física de um arquivo (que é único)
  - Cada arquivo pode ter no máximo um índice primário OU um índice de clustering

Índices Primários, de Clustering ...

- Índices primários e de clustering são baseados no campo de ordenação física de um arquivo (que é único)
  - Cada arquivo pode ter no máximo um índice primário OU um índice de clustering

- E para os campos que não ordenam fisicamente o arquivo? Podemos ter algo semelhante?

Índices Primários, de Clustering ...

- Índices primários e de clustering são baseados no campo de ordenação física de um arquivo (que é único)
  - Cada arquivo pode ter no máximo um índice primário OU um índice de clustering

- E para os campos que não ordenam fisicamente o arquivo? Podemos ter algo semelhante?
  - Quantos **índices secundários** quisermos!

Índices Secundários

- Índice secundário: arquivo ORDENADO de registros de tamanho fixo contendo os campos $< i, w >$, sendo i um campo de indexação que NÃO ordena fisicamente os registros, podendo ter valores distintos ou não
  - w aponta para um bloco ou registro
- Podem existir vários índices secundários

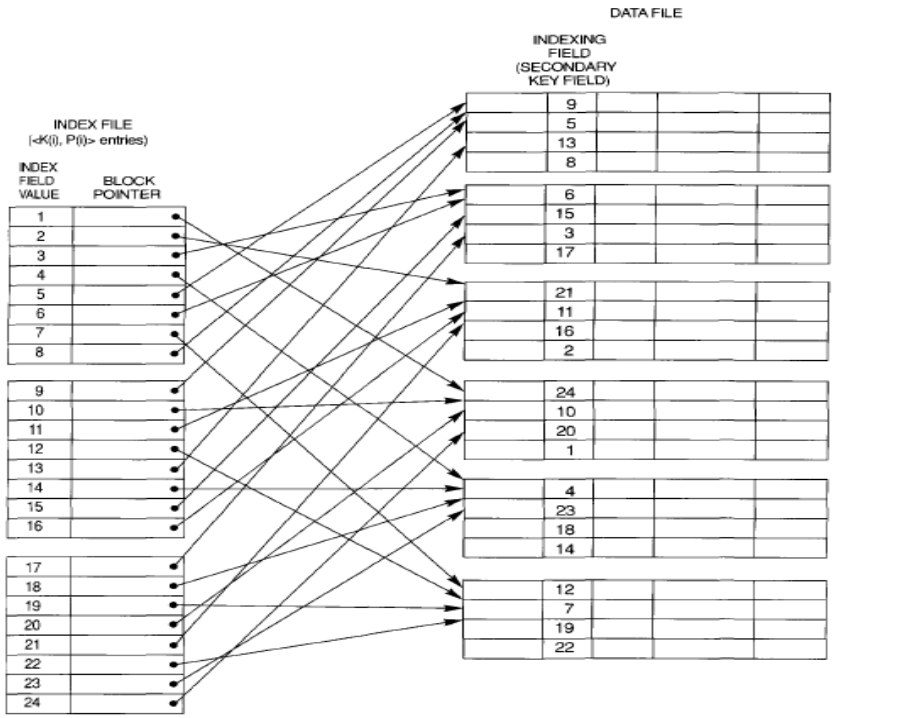


FIGURE 14.4 A dense secondary index (with block pointers) on a nonordering key field of a file.

Elmasri & Navathe, 2004

Índices Secundários

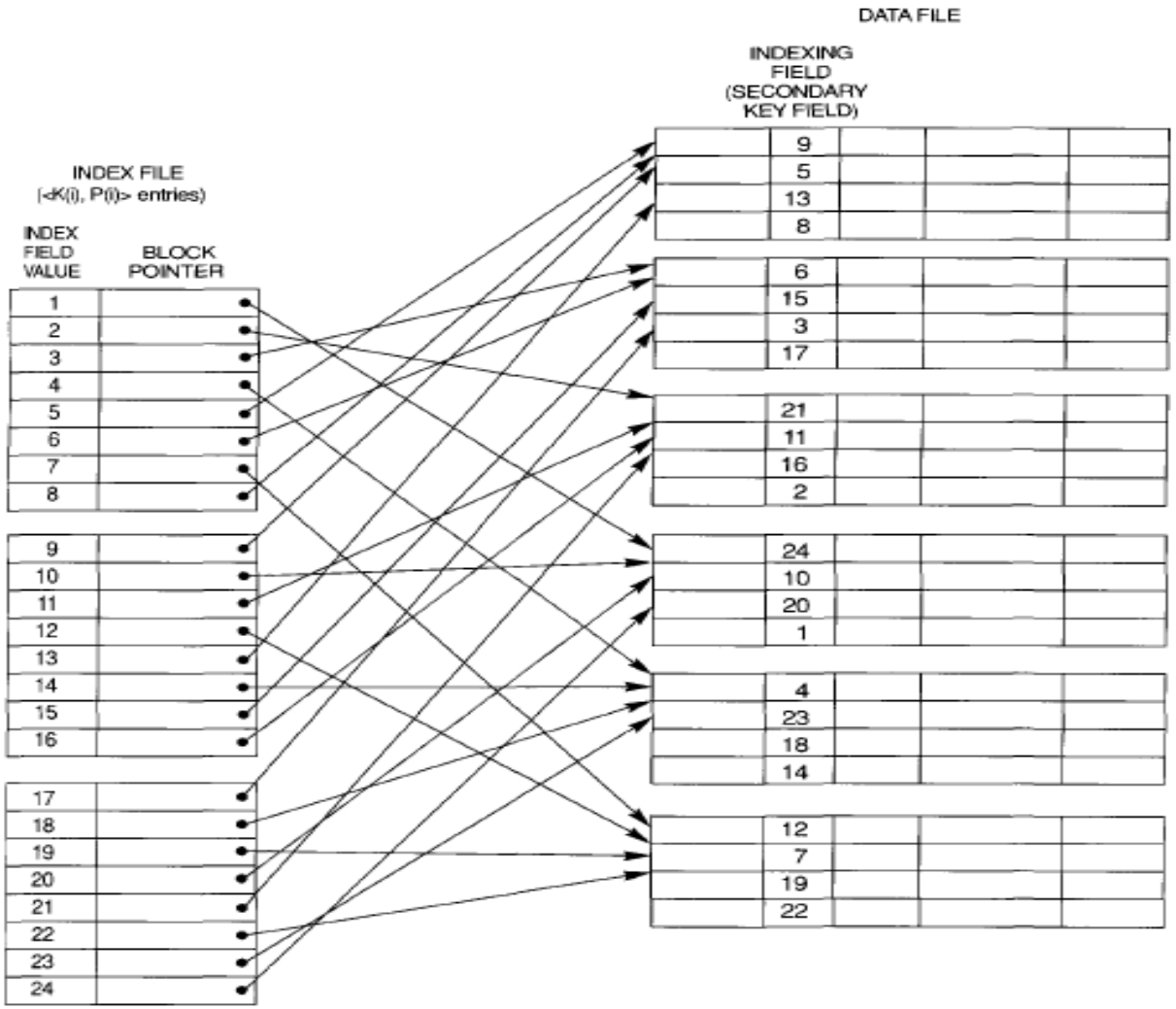


FIGURE **14.4** A dense secondary index (with block pointers) on a nonordering key field of a file.

Elmasri & Navathe, 2004

Índices Secundários

- Se i tem valores distintos, o índice é denso
- Se i tem valores duplicados:
  - Opção 1: diversas entradas para um mesmo i, cada uma com w apontando para um registro (denso)
  - Opção 2: 1 entrada para cada valor de i, e w multivalorado (campo de tamanho variável) aponta para blocos (esparso)
  - Opção 3: uma entrada para cada valor de i e w (campo de tamanho fixo) aponta para um bloco de ponteiros de registros (esparso) – mais usada

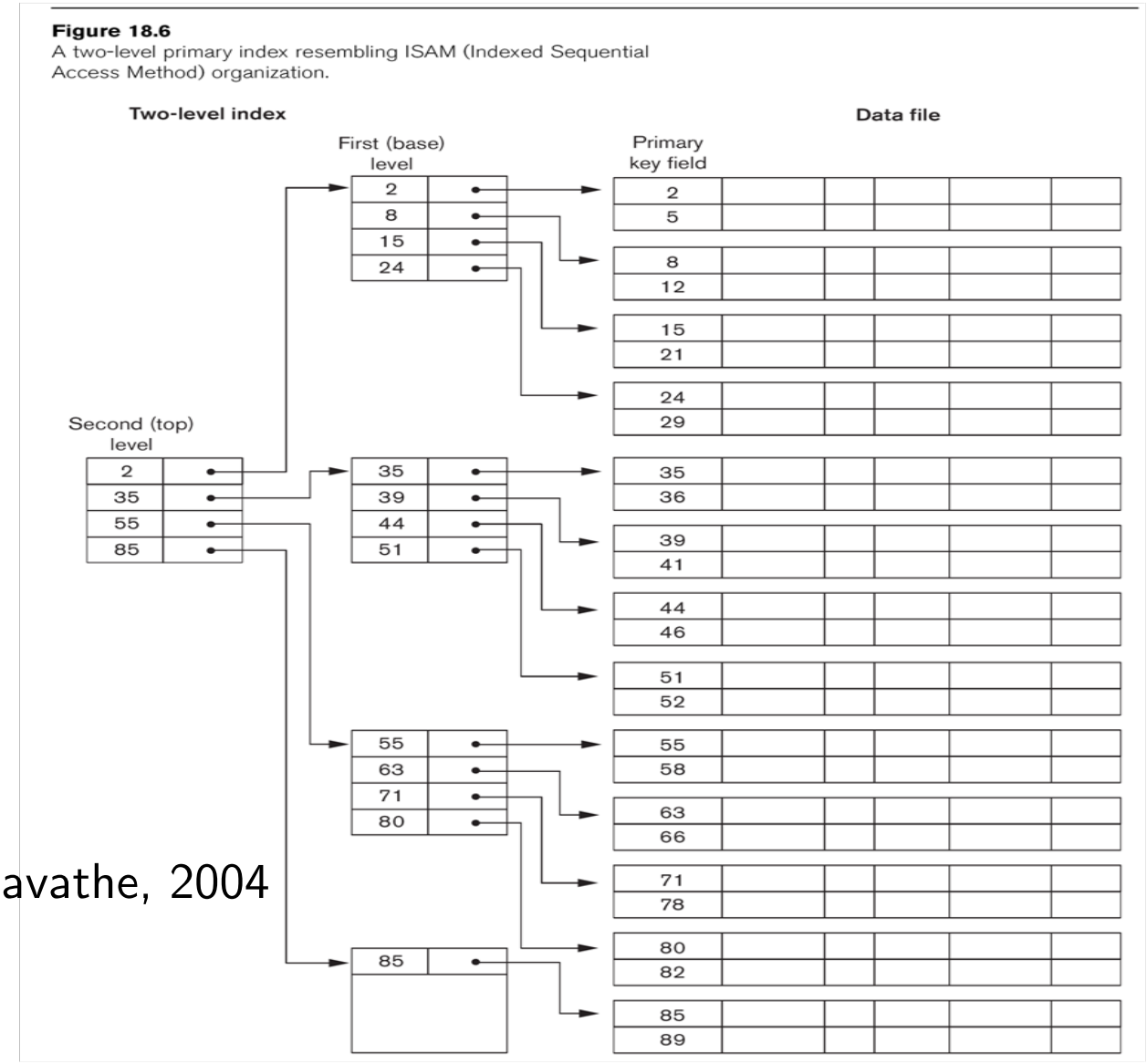
Para esses três tipos de índices:

- Busca binária – O(log bi)
- O que dá para fazer se o arquivo é muito grande e o próprio índice ficou grande (com muitos blocos, isto é, grande bi)?
  - Índice do índice

Organização Indexada Multiníveis

- Nível 1: arquivo de índices para os dados
- Nível 2: arquivo de índices para o arquivo de índices nível 1
- Se no nível 2 precisar de mais de um bloco, criar nível 3
- ...

Organização Indexada Multiníveis



Elmasri & Navathe, 2004

Organização Indexada Multiníveis

- **Busca**:

Organização Indexada Multiníveis

- **Busca**: 1 acesso em cada nível (rápida!) - $O(t)$ - precisa acessar t blocos, sendo t o número de níveis

Organização Indexada Multiníveis

- **Busca**: 1 acesso em cada nível (rápida!) - $O(t)$ - precisa acessar t blocos, sendo t o número de níveis
  - $t = \lceil (log_{fbi}(r1)) \rceil$
    - fbi = número de registros que cabem em um bloco de índice (fator de blocagem dos blocos de índice)
    - $r1$ = número total de registros de índice no nível 1

Organização Indexada Multiníveis

- **Inserção / Exclusão**:

Organização Indexada Multiníveis

- **Inserção / Exclusão**: cada vez mais complexas

Organização Indexada Multiníveis

- **Inserção / Exclusão**: cada vez mais complexas
  - Podemos precisar alterar tudo!

Organização Indexada Multiníveis

- **Inserção / Exclusão**: cada vez mais complexas
  - Podemos precisar alterar tudo!

- Como ter uma busca eficiente e inserções e remoções com complexidade satisfatória?

Organização Indexada Multiníveis

- **Inserção / Exclusão**: cada vez mais complexas
  - Podemos precisar alterar tudo!

- Como ter uma busca eficiente e inserções e remoções com complexidade satisfatória?
- Assuntos para as próximas aulas!

Referência

- Slides baseados no material da profa. Ariane Machado Lima - ACH2024
- ELMASRI, R.; NAVATHE, S. B. Fundamentals of Database Systems. 4 ed. Ed. Pearson-Addison Wesley. Cap 13 (até a seção 13.7), 2004
- GOODRICH et al, Data Structures and Algorithms in C++. Ed. John Wiley & Sons, Inc. 2nd ed. 2011. Seção 14.2
- ELMARIS, 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

Algoritmos e Estruturas de Dados II
Aula 21 – Alocação Indexada

Prof. Luciano A. Digiampietri
digiampietri@usp.br
@digiampietri

2025