Índices

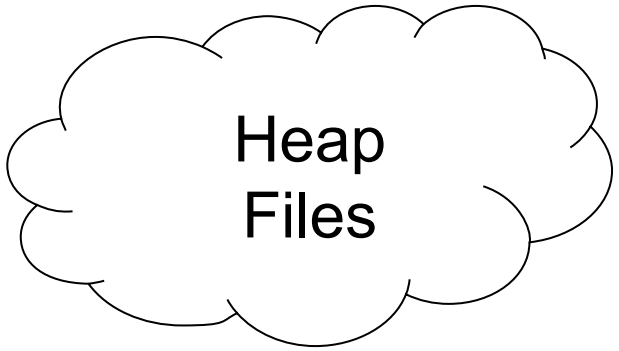
Banco de Dados II

(Capítulos 8, 10 e 11 – Ramakrishnan
Capítulo 12 – Silberschatz
Capítulo 14 – Garcia-Molina, Ullman, Widom)
Berkeley → http://www-inst.eecs.berkeley.edu/~cs186/

Introdução

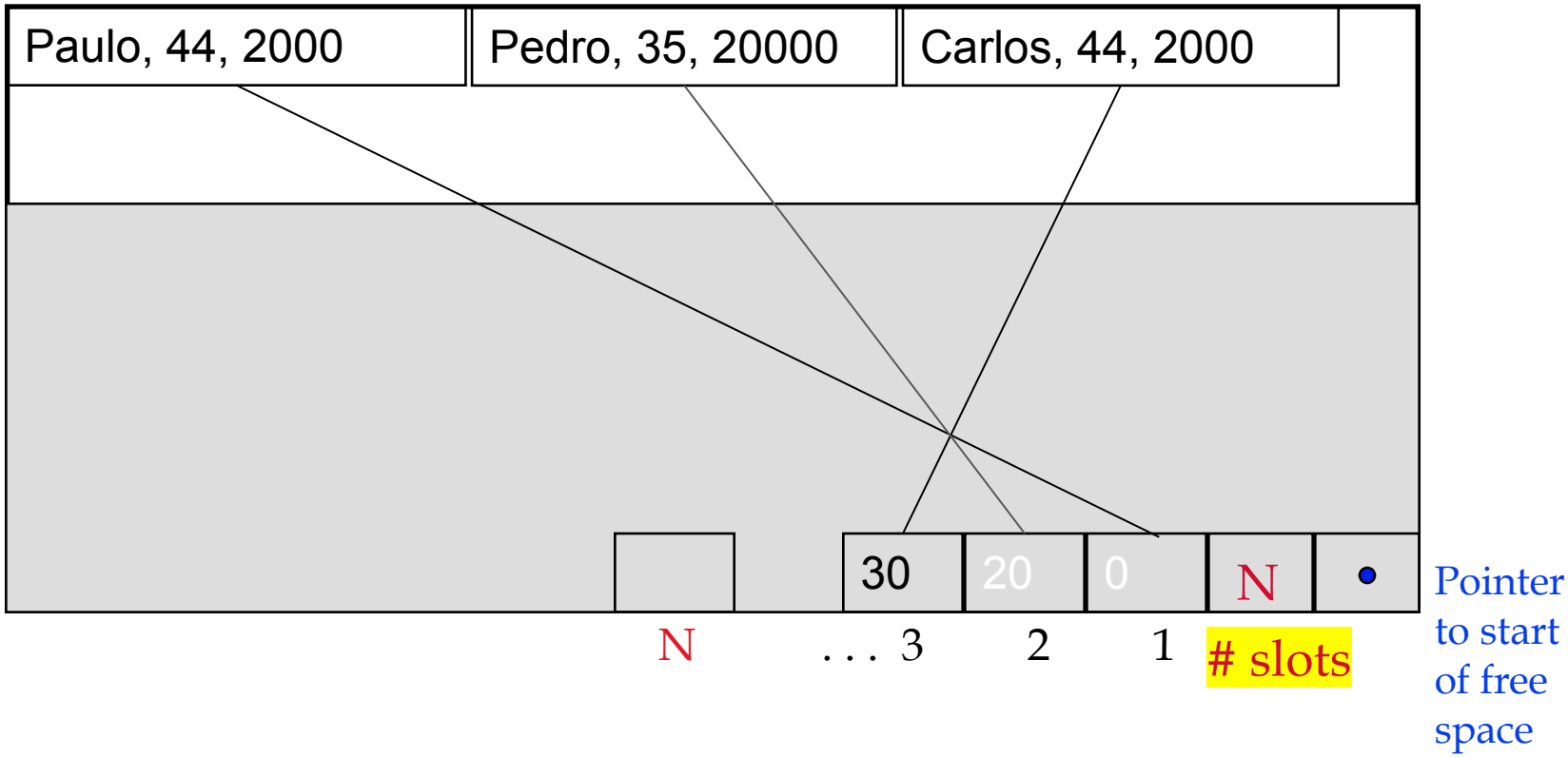
Dados sem organização

Paulo, 44, 2000
Pedro, 35, 20000
Carlos, 44, 2000
José, 40, 2500
João, 35, 3000
Ilmério, 40, 3500
Rodrigo, 40, 3500
Maria, 30, 4000
Sara, 35, 4000
Sabrina, 31, 5000



•

Heap Files


Paulo, 44, 2000
Pedro, 35, 20000
Carlos, 44, 2000
30
20
0
N
N
. . . 3
2
1
slots
Pointer to start of free space

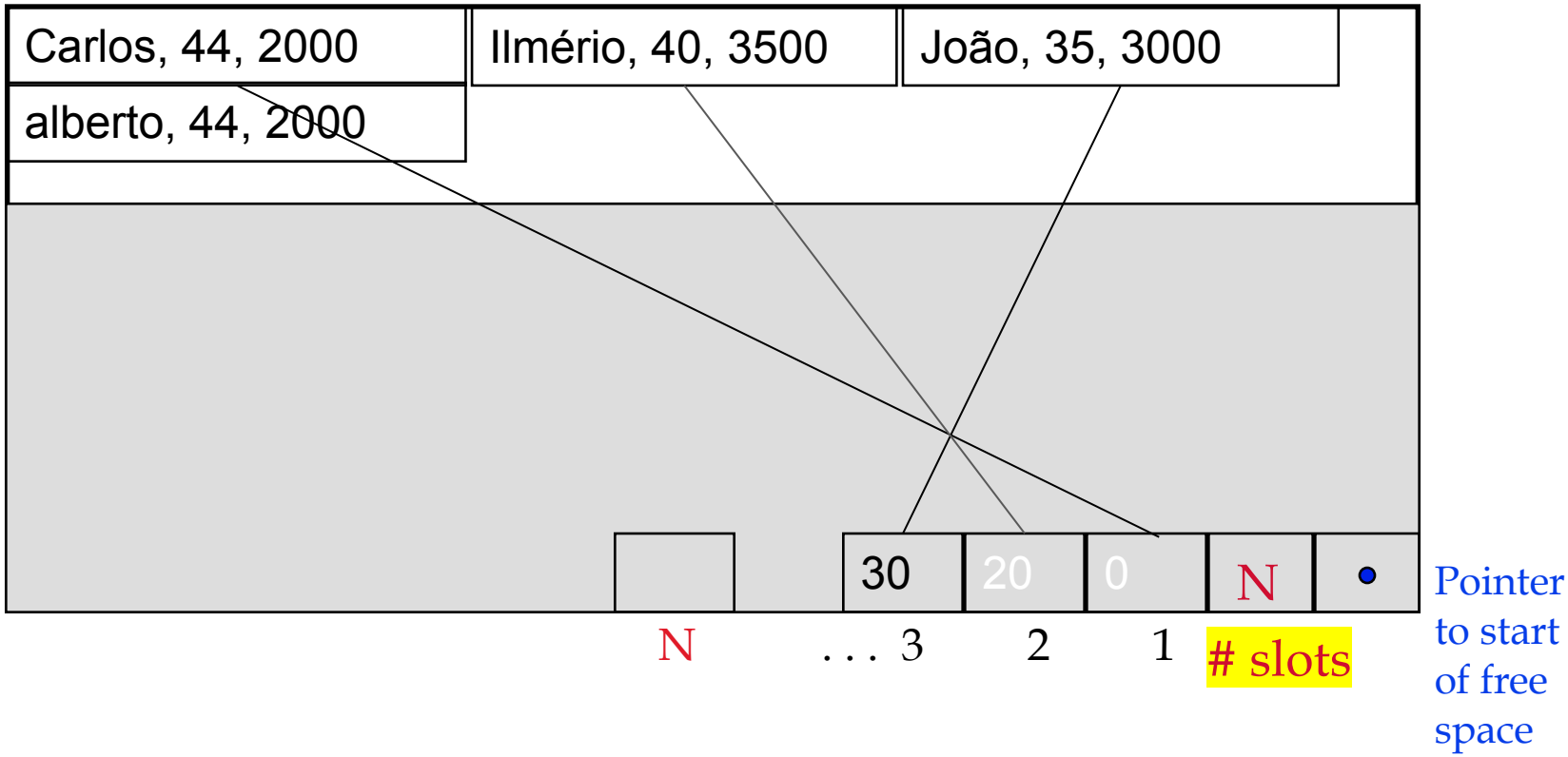
Introdução

Dados Organizados (nome)

Carlos, 44, 2000
Ilmério, 40, 3500
João, 35, 3000
José, 40, 2500
Maria, 30, 4000
Paulo, 44, 2000
Pedro, 35, 2000
Rodrigo, 40, 3500
Sabrina, 31, 5000
Sara, 35, 4000

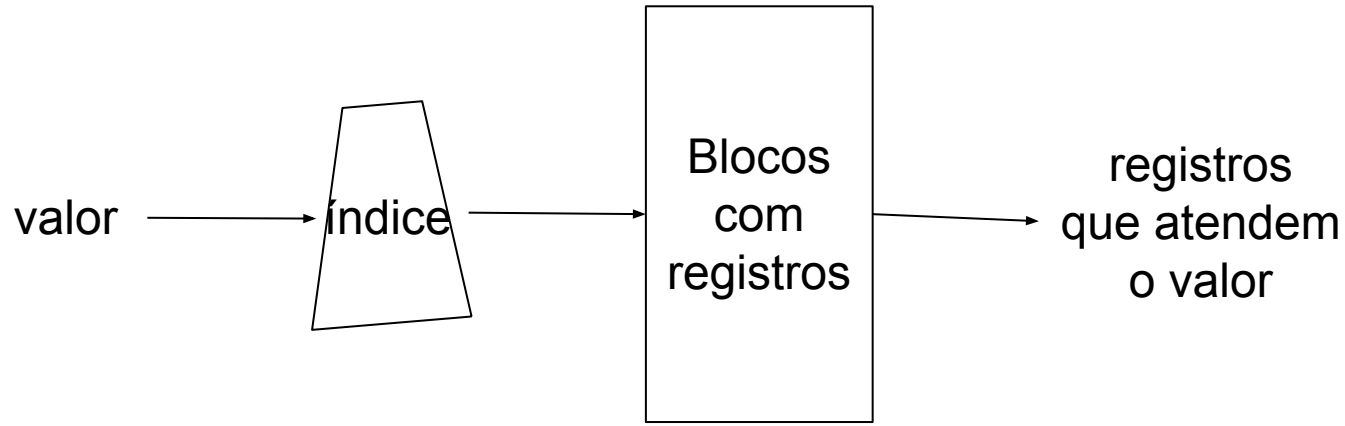
•

Heap Files - Exemplo



Índices

- estrutura auxiliar projetada para agilizar operações de busca, inserção e supressão
-



- Alteração nos dados pode levar à alteração no índice
- Espaço extra de armazenamento

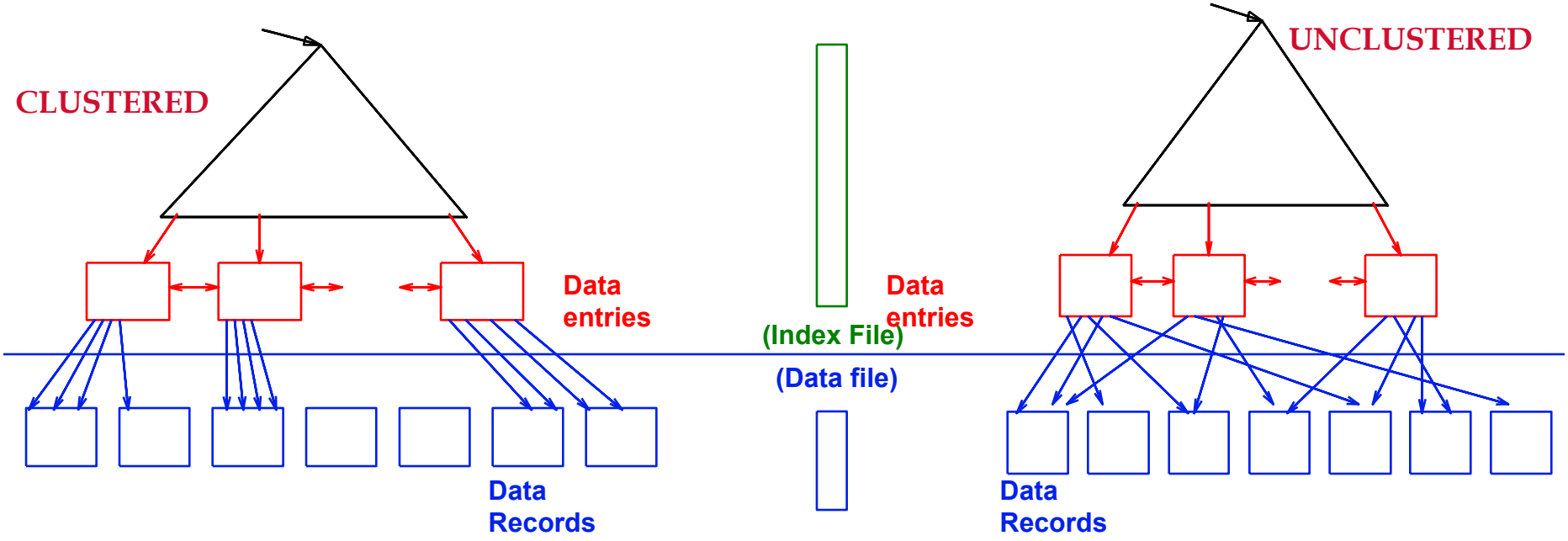
Classificação dos índices

- *Clustered (agrupado)*: se a ordem dos registros são os mesmos da ordem do índice de dados de entrada.
- Unclustered (desagrupado): os registros são armazenadas sem um ordenamento

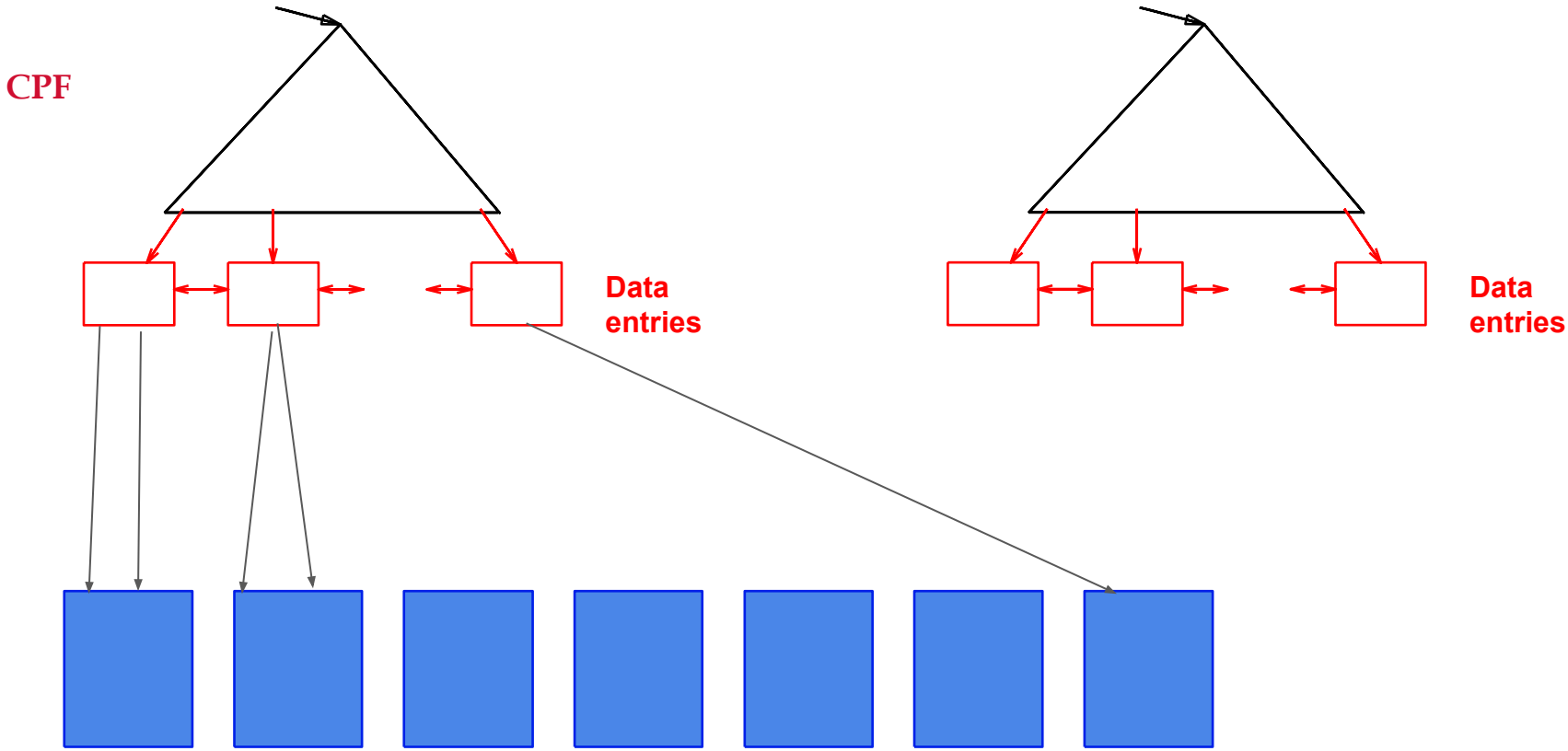
Um arquivo pode ser clusterizado por somente uma chave

Alternativa 1 é clusterizada ou não?

Clustered vs. Unclustered Index



Clustered vs. Unclustered Index



Clustered

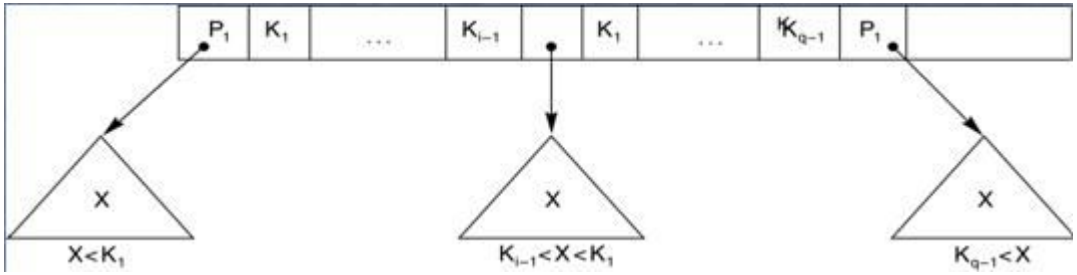
- Clustered Pros
  - Eficiente em buscas entre faixas (> <)
  - Pode ser executado algum tipo de compressão
  - Possível benefício na localidade dos dados
- Clustered Contras
  - Caro para manter
  - Espaço extra de armazenamento

Índices

- Primário
  - A chave do índice é composta pela chave primária da tabela
    - A maioria dos SGB cria índices primários automaticamente
  - Não permitem duplicatas
- Secundário
  - Outras colunas da tabela participam
  - Permitem duplicatas

Índices

- Os índices de múltiplos níveis podem formar uma árvore



- Atualização nos dados, implica na atualização em todos os níveis

Índices B+

- Os SGBs implementam índices multiníveis através de árvores B+
  - Atualização dos níveis mais eficientes
  - Cada nível elimina vários acessos
  - O grau (ou ordem) da árvore indica o número de acessos

Índices B+

-Insert/delete com custo de I/O $\log_f$ (N)

Onde N= número de folhas e F= ordem ou grau da árvore

-Mínimo de 50% de ocupação (exceto nó root).

-Exemplo: ordem 5-> número mínimo de chaves = (M-1)/2

número máximo de chaves = 4

-Suporte de seleção por igualdade e por range.

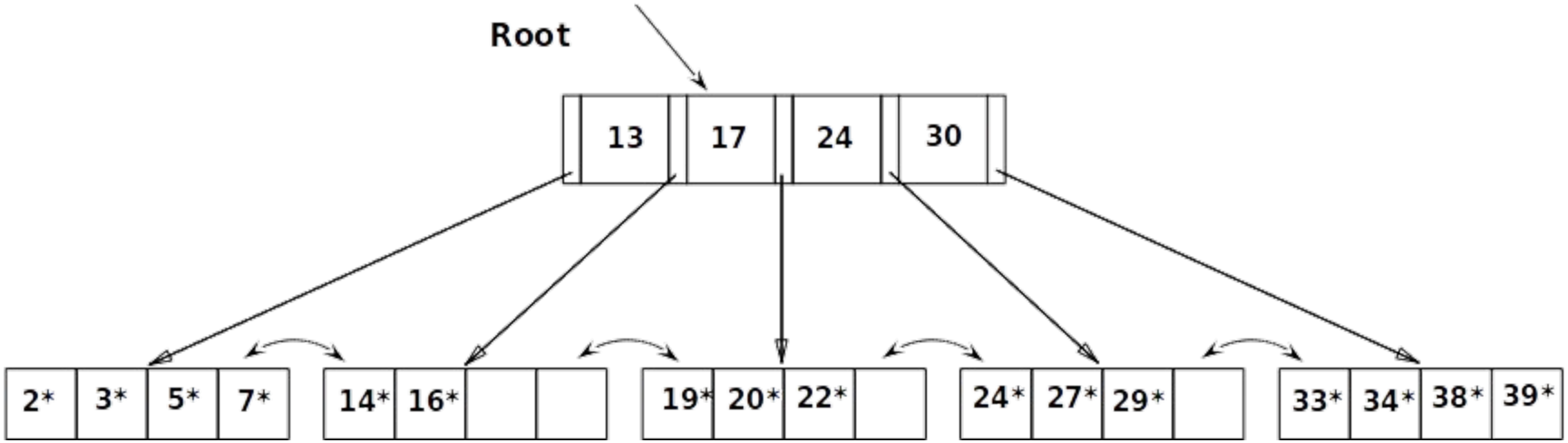
-Busca sempre no nó folha

Índices B+

Construir a árvore B+ para a seguinte sequência: 30, 20, 40, 50, 35,50,15,25

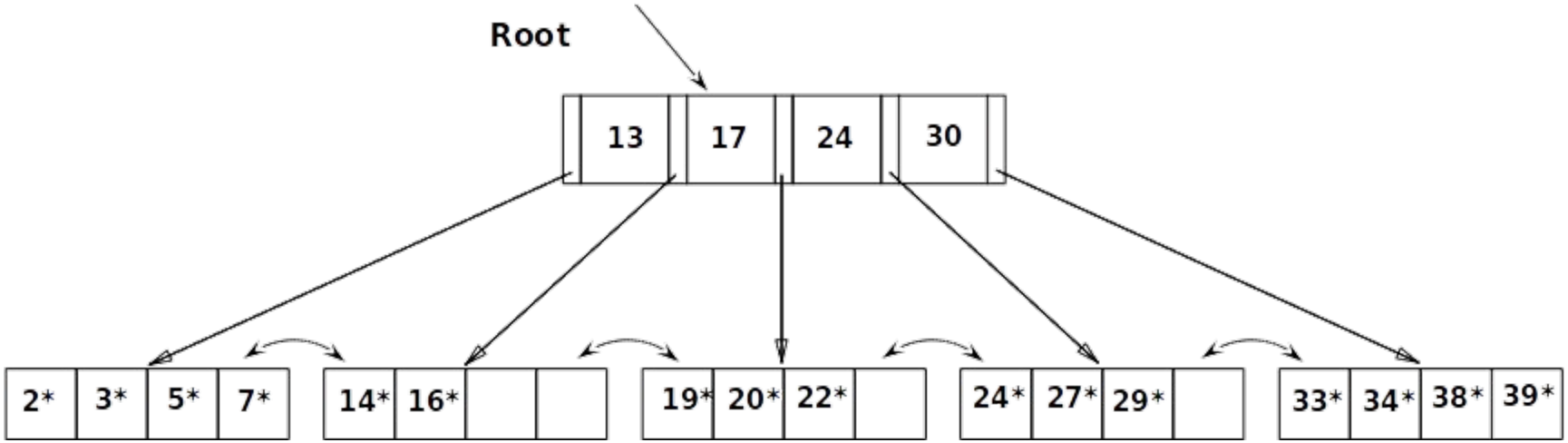
Índices B+

Ordem 5



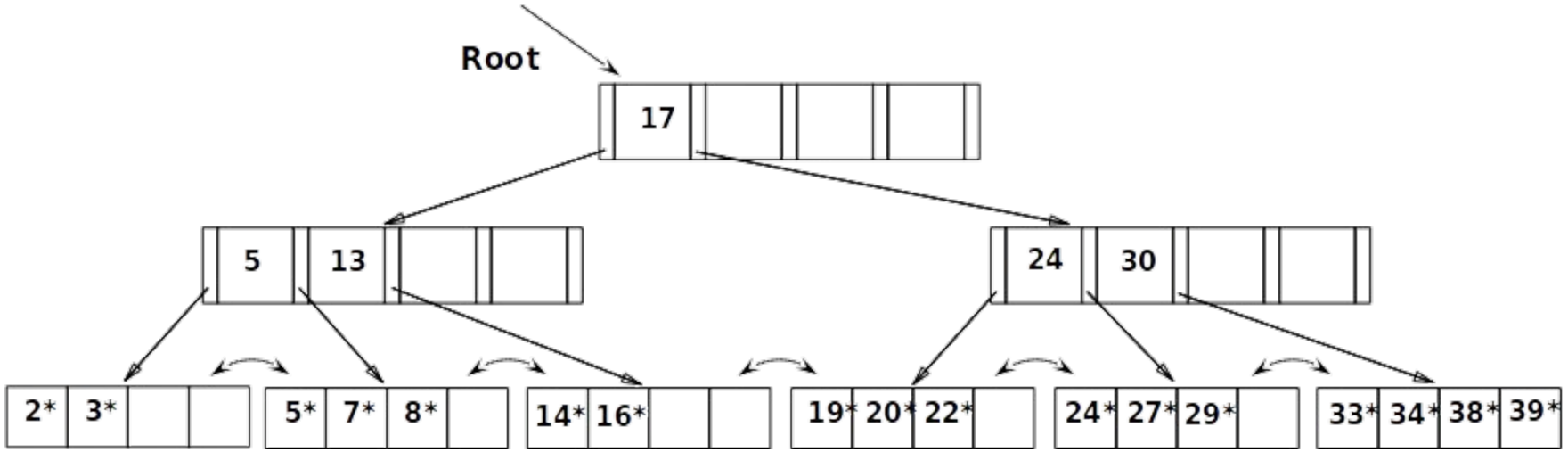
Índices B+

Inserte valor 8*



Índices B+

Deletar a entrada 19* e 20*



Índices B+

1- Inserir em um B+ de ordem 5 os seguintes valores: 10, 15, 20,25,30,35,40,45,50

2 -Remover os seguintes valores: 35 30 25 20

Índices B+

- Qual a altura máxima de uma árvore B+ com n chaves?
  - Esta questão é importante pois a altura da árvore dará o limite máximo de acesso ao disco
  - Assim, pior altura

$$h = \log_{\frac{ordem}{2}} n$$

Índices B+

- Dada uma árvores B+ de ordem b, os tempos serão:
  - Inserção

$$h = \log_b(n)$$

  - Busca

$$h = \log_b(n)$$

*Onde *n* é o número de chaves

Atividade

1) Inserir em um B+ com ordem 4 as seguintes entradas :2, 6, 17, 20, 24, 25, 27, 29, 30, 31, 32, 5, 21, 1, 40, 45, 50, 70
2) Remover os itens 25, 6, 5 e 20.
3) Qual a altura mínima e máxima de uma árvore de ordem 100 com 1 milhões de chaves.

4) Sabendo que a árvore B+ possui 1 milhão de chaves, a ordem é 1000, e o custo para carregar um bloco do disco é 1ms, quanto tempo é necessário para buscar uma chave no pior caso e no melhor caso ? (sabendo que os nós estão no disco, um nó é armazenado em uma página, o tempo de processamento em memória pode ser ignorado)

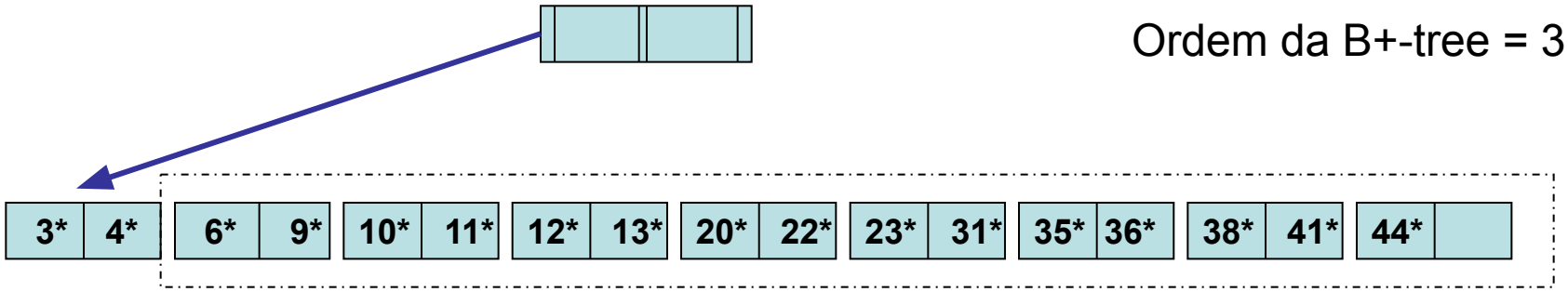
Índices

- Let's check it out:
  - Animação (here)
  - Or type http://www.cs.usfca.edu/~galles/visualization/BPlusTree.html

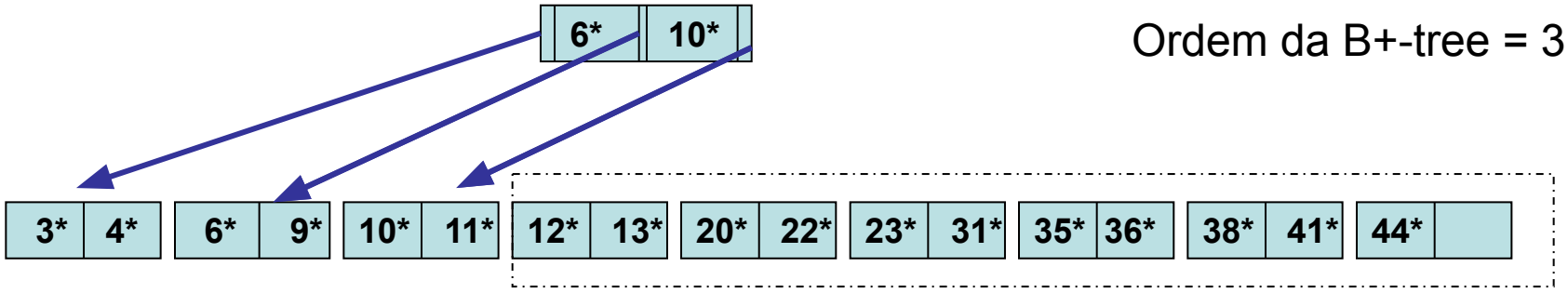
Bulking Load (Construção da B+ Tree)

- Utilizando o arquivo de índice denso (ou arquivo ordenado)
- Aloca-se uma página vazia para a raiz
- Insere nesta página um ponteiro para a primeira página do arquivo contendo as entradas.

Bulking Load (Construção da B+ Tree)



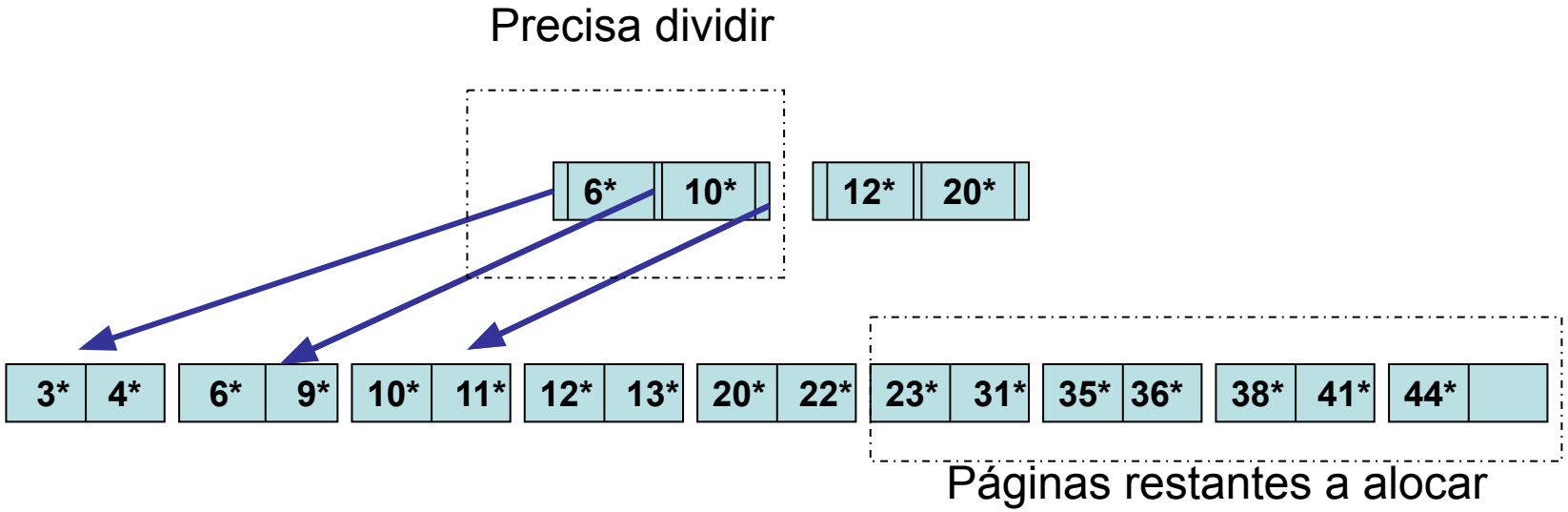
Bulking Load (Construção da B+ Tree)



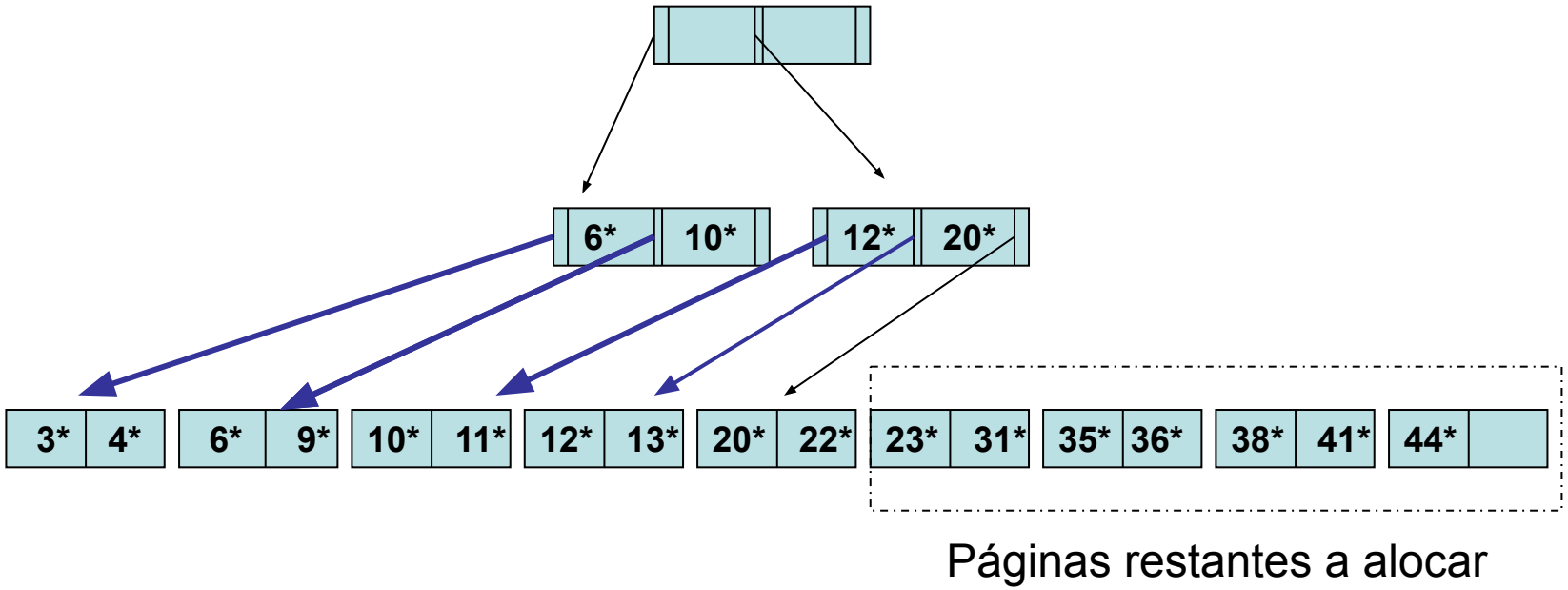
Ordem da B+-tree = 3

Páginas restantes a alocar

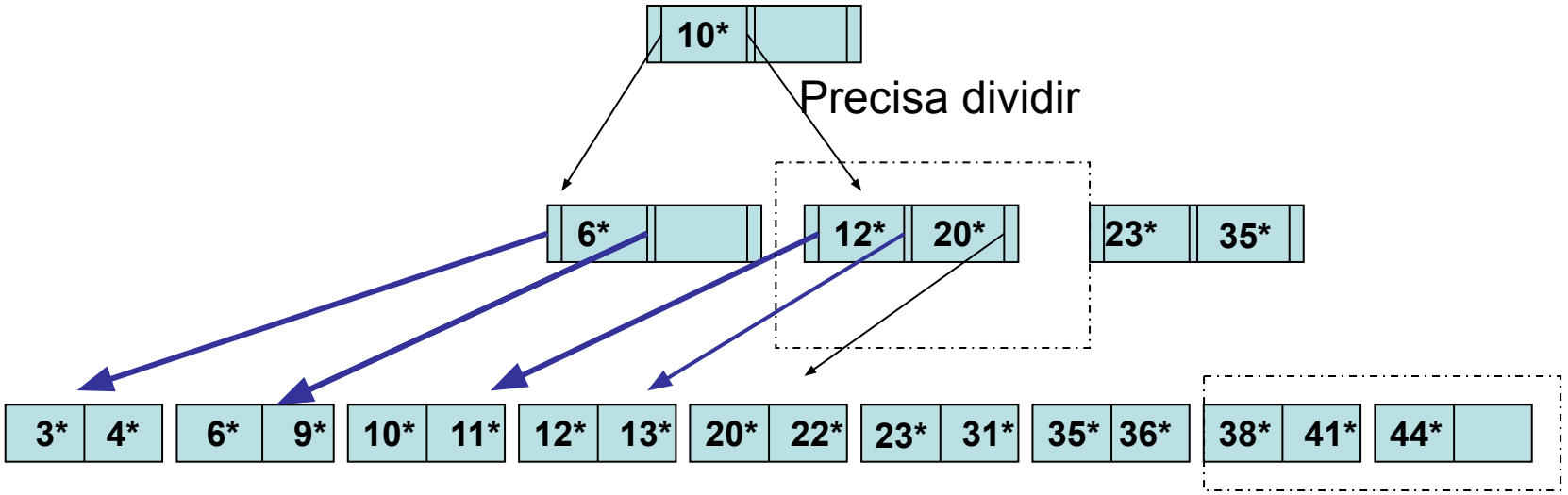
Bulking Load (Construção da B+ Tree)



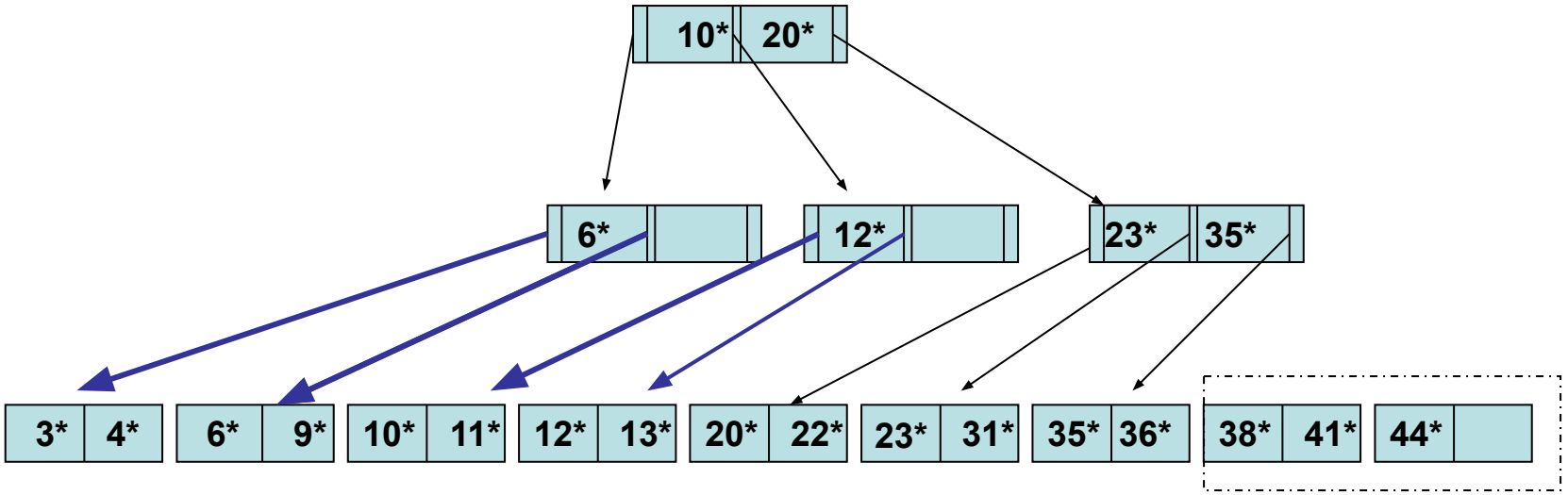
Bulking Load (Construção da B+ Tree)



Bulking Load (Construção da B+ Tree)

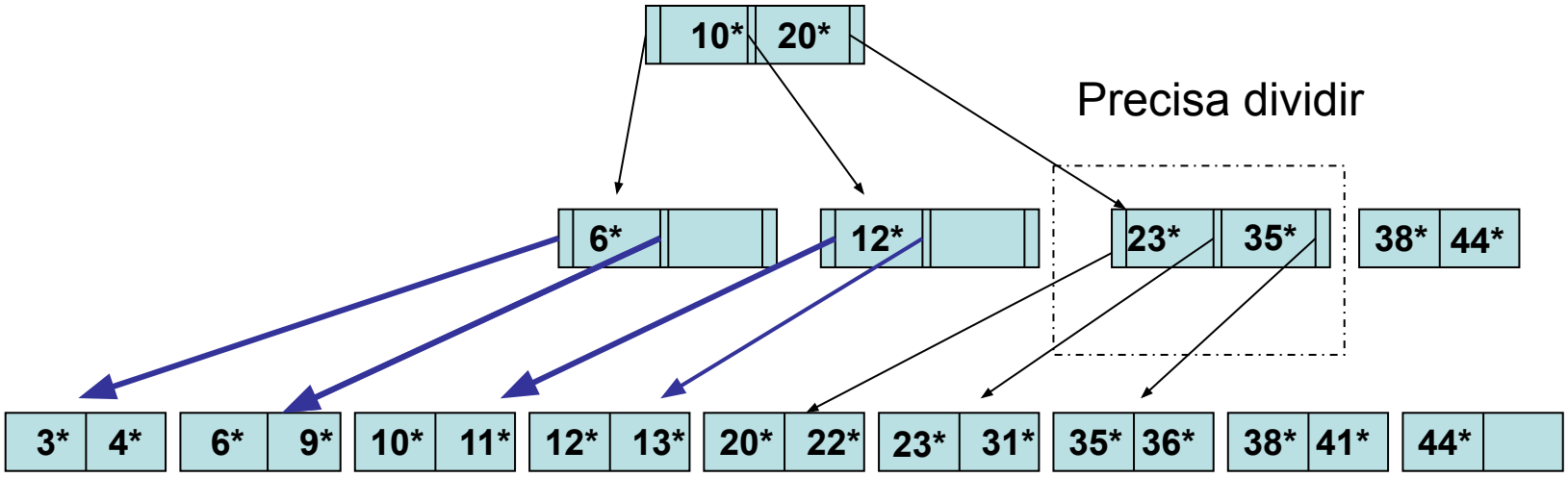


Bulking Load (Construção da B+ Tree)

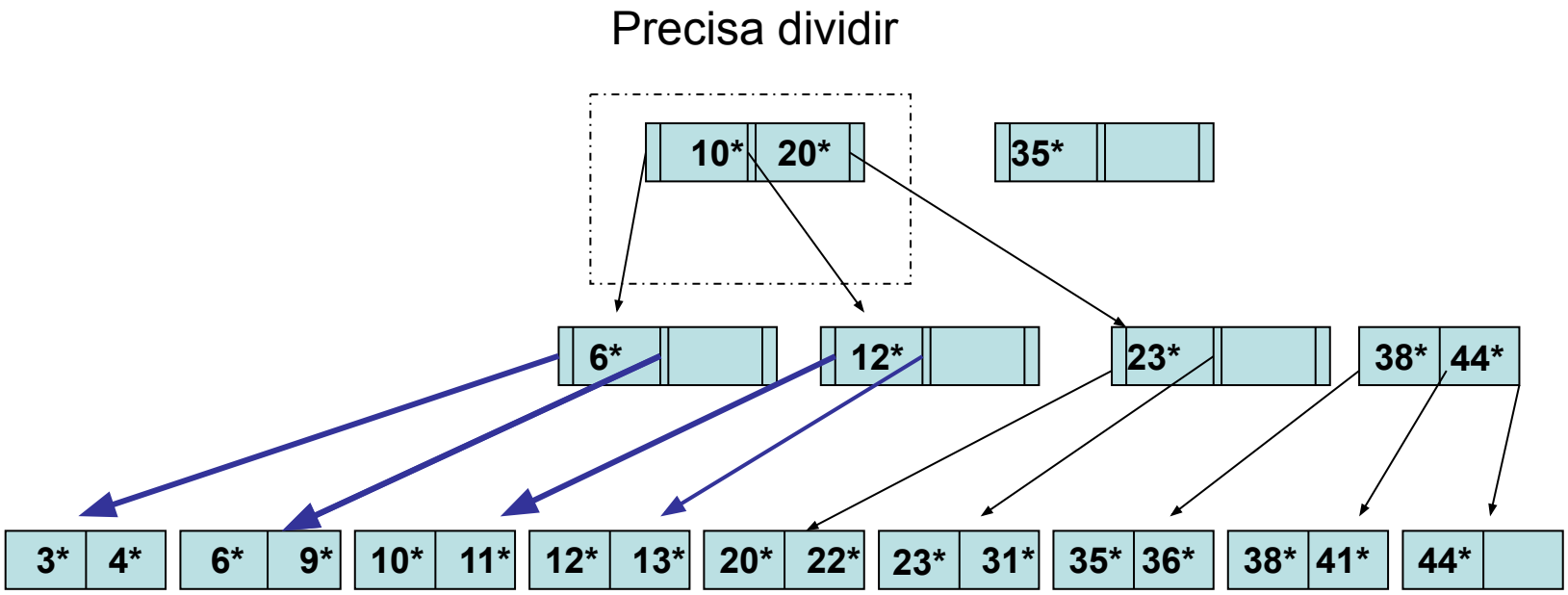


Páginas restantes a alocar

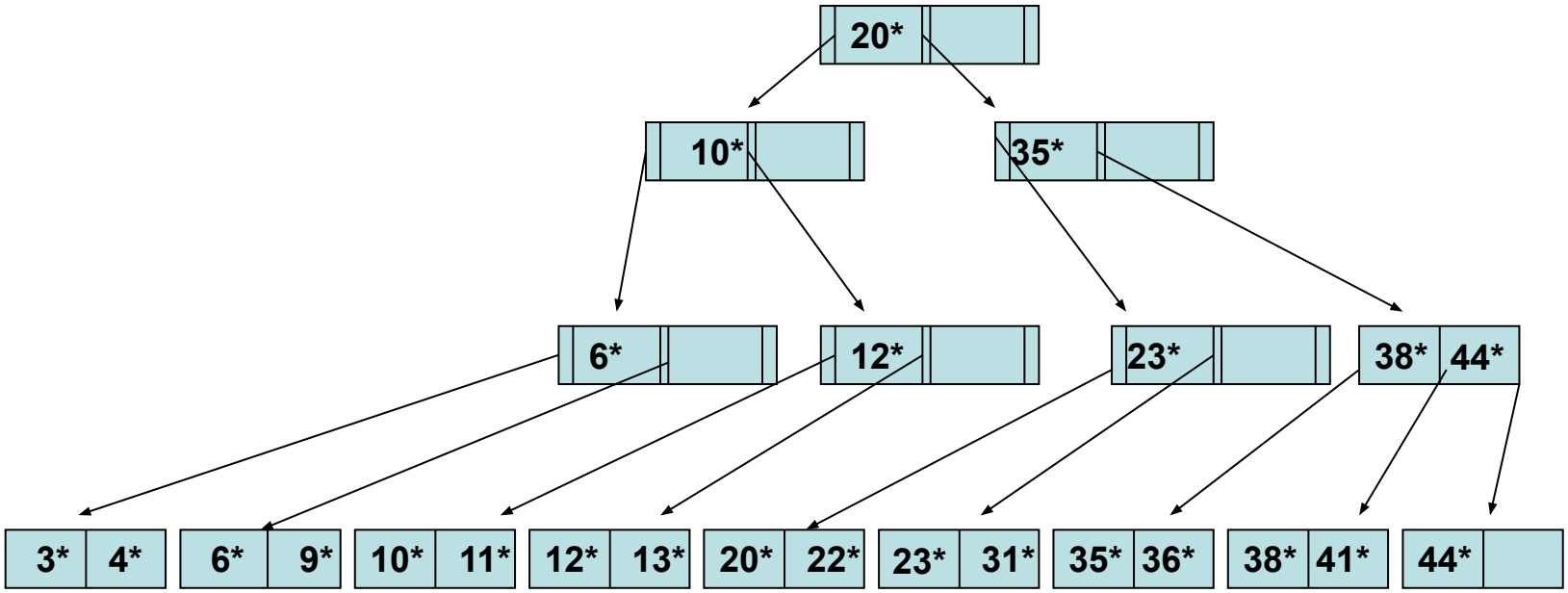
Bulking Load (Construção da B+ Tree)



Bulking Load (Construção da B+ Tree)



Bulking Load (Construção da B+ Tree)



Árvore construída!

Atividade

1) Faça a leitura usando bulking load com a ordem 4 dos seguintes valores: 1 ,2 ,5 ,10 ,15 , 20, 40,60, 80 ,100 ,120 ,140,

Índices

- Conclusões
  - Os dados são mais acessados que atualizados
  - Necessário existir uma estrutura auxiliar para melhorar o desempenho das consultas
  - Para dados relacionais árvores B+ são os índices mais utilizados
  - O Otimizador de consultas utiliza índices sempre que possível