

Sistemas de Gerência de Bancos de Dados

Módulo 2 - Indexação

PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO



Tópicos

- Introdução
- Árvores-B⁺
- Hashing Expansível

Introdução

- Índice Primário (ou índice de clustering):
 - ▶ índice cuja chave especifica a ordem sequencial do arquivo
 - ▶ índice denso:
 - há uma entrada no índice para cada valor de chave que ocorre em um registro de dados
 - a entrada aponta para o primeiro registro que contém aquele valor de chave
 - ▶ índice esparso:
 - há uma entrada no índice apenas para alguns valores de chave
 - a entrada aponta para o primeiro registro que contém aquele valor de chave
 - para localizar um registro com chave K, procura-se a entrada E do índice com o maior valor de chave menor ou igual a K e pesquisa-se o arquivo a partir do registro apontado por E

Introdução

- Índice Secundário:
 - ▶ índice cuja chave não é aquela da ordem sequencial do arquivo
 - ▶ organização:
 - há uma entrada no índice para cada valor de chave que ocorre em um registro de dados
 - a entrada aponta para **todos** os registros que contém aquele valor de chave x

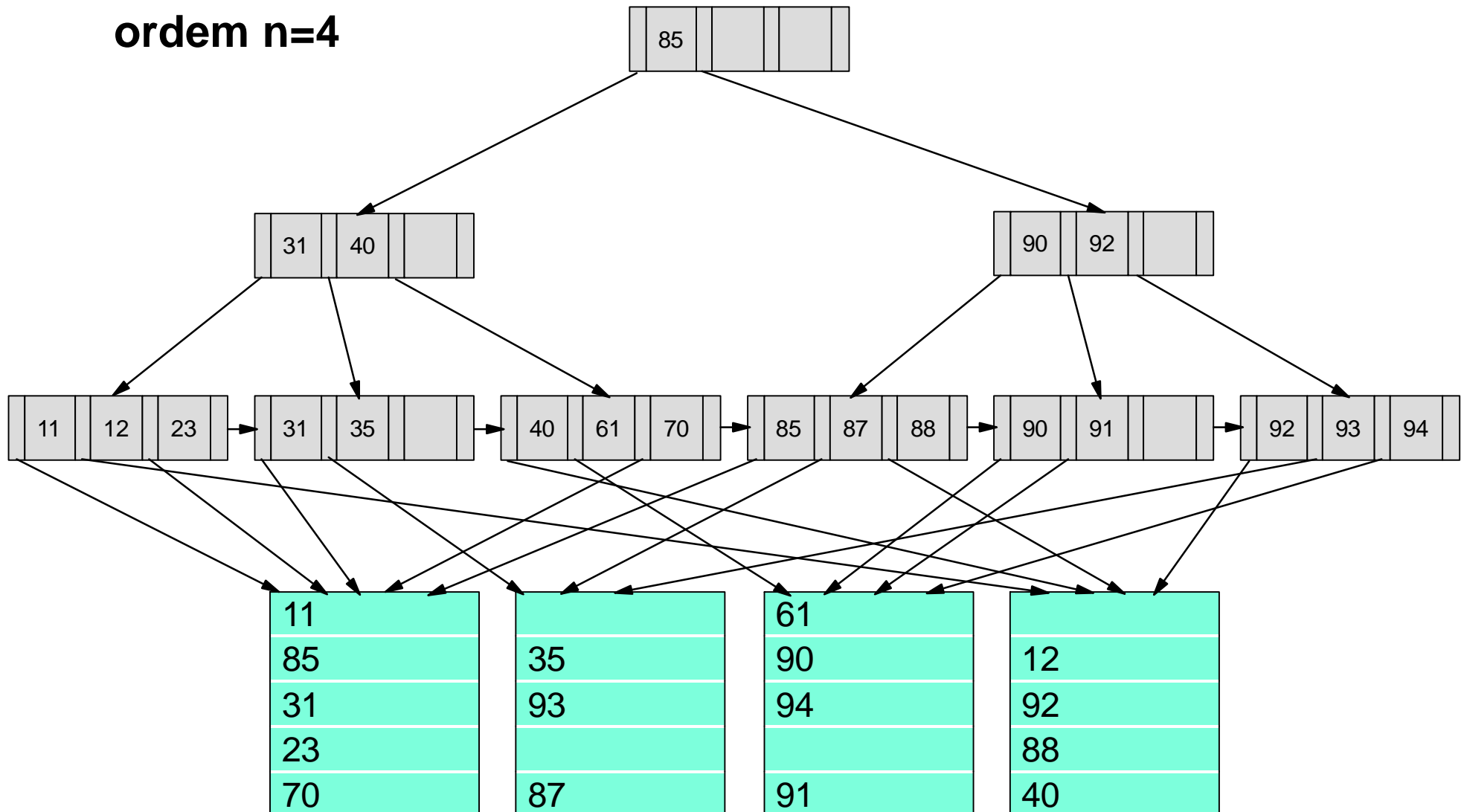
Árvores-B⁺

■ Definição:

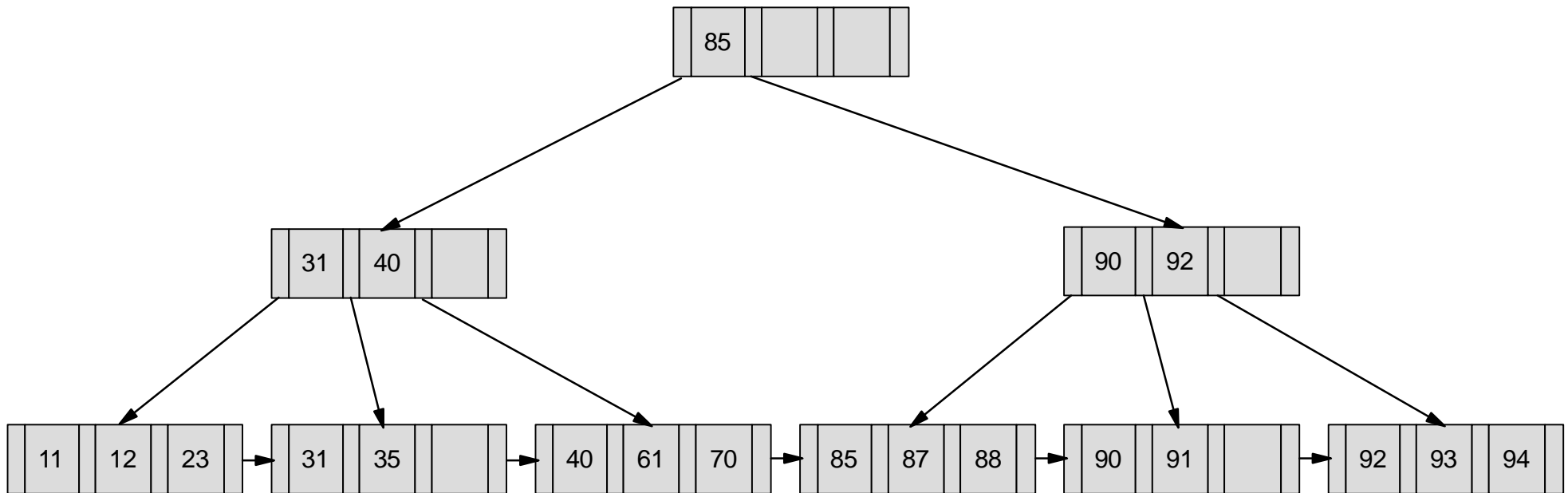
- ▶ árvore de busca de ordem n
- ▶ a raíz possui no mínimo 2 filhos
- ▶ cada folha possui no mínimo $\lceil (n-1)/2 \rceil$ valores de chave
- ▶ cada nó interior possui no mínimo $\lceil n/2 \rceil$ filhos
- ▶ todos os ramos tem o mesmo comprimento

Árvores-B⁺

ordem n=4

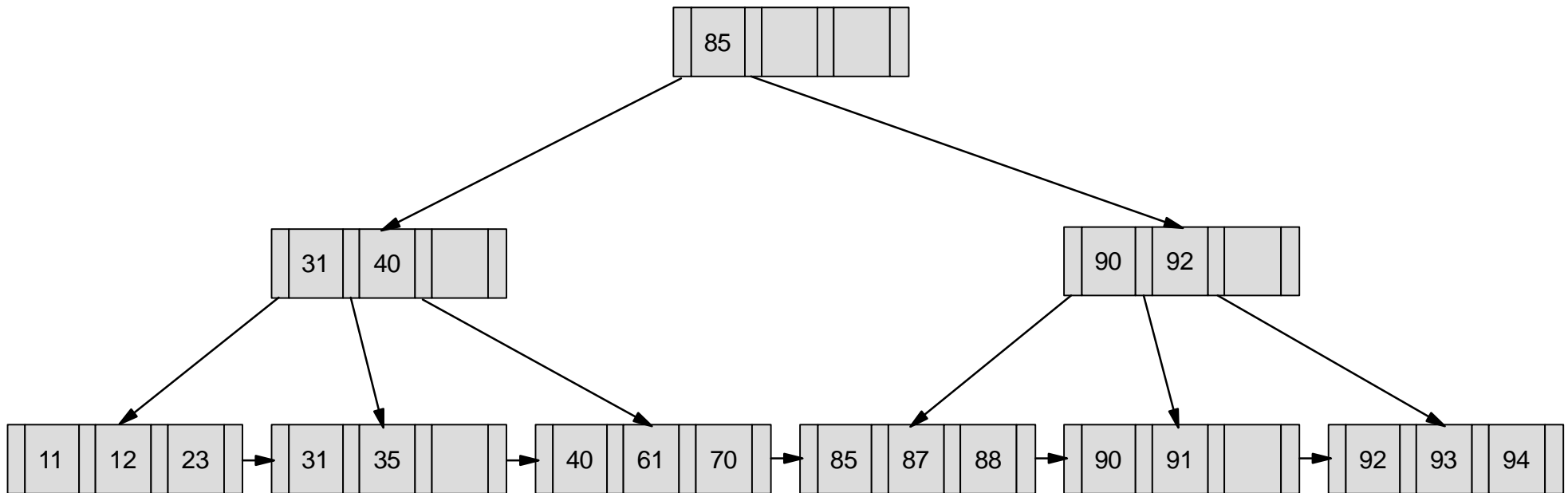


Árvores-B⁺



- as folhas formam um índice denso para o arquivo:
 - ▶ contém todos os valores de chave, em ordem ascendente
 - ▶ o ponteiro precedendo um valor de chave k aponta para um bloco de ponteiros apontando para os registros com chave k
 - ▶ o último ponteiro de uma folha aponta para a próxima folha
- os nós internos formam um índice esperso para as chaves contidas nas folhas

Árvores-B⁺

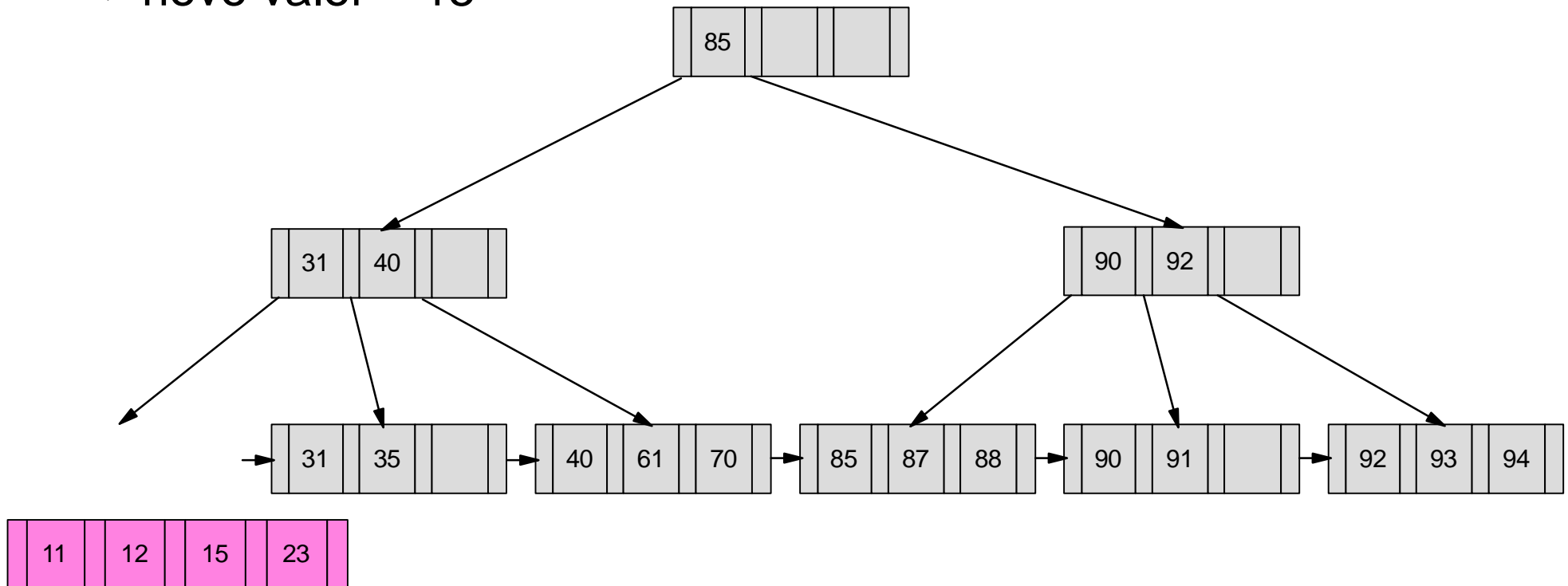


► $[31, 85)$ é uma "assinatura" para o conjunto $\{31, 35, 40, 61, 70\}$

Árvores-B⁺

■ Inserções

► novo valor = 15

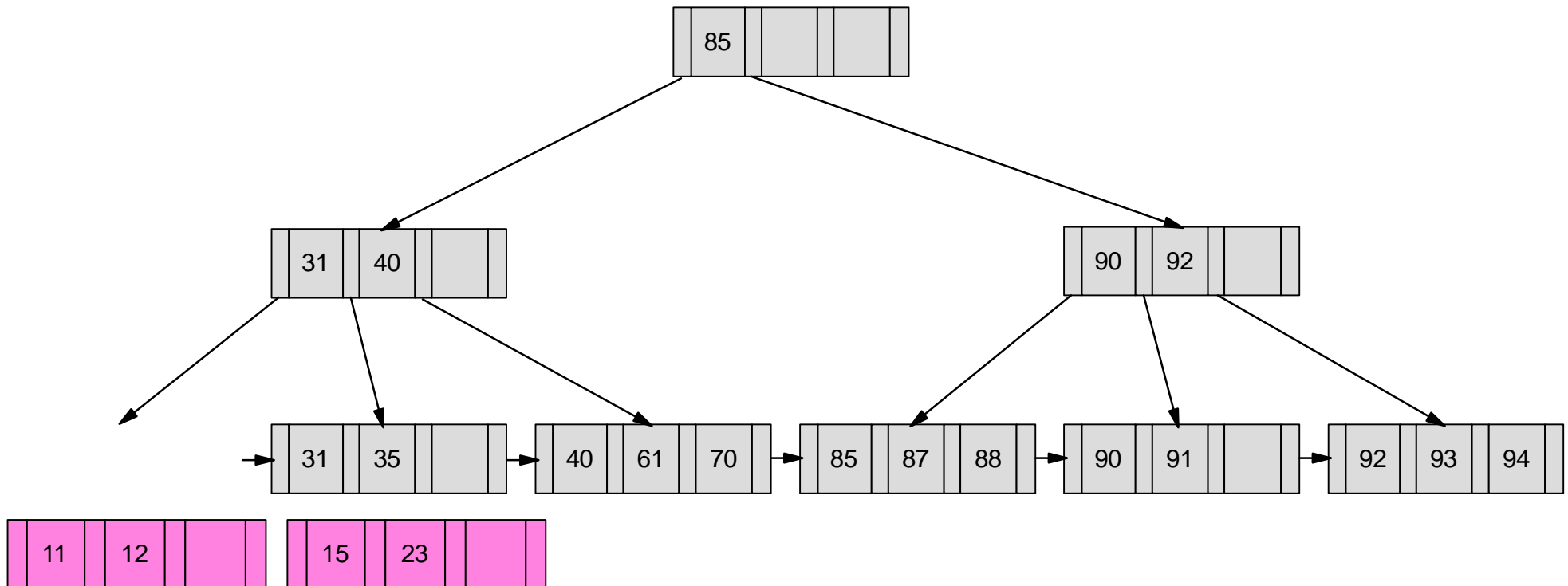


► insere-se o novo valor 15 na folha correta, que neste caso ficará grande demais

Árvores-B⁺

■ Inserções

► novo valor = 15

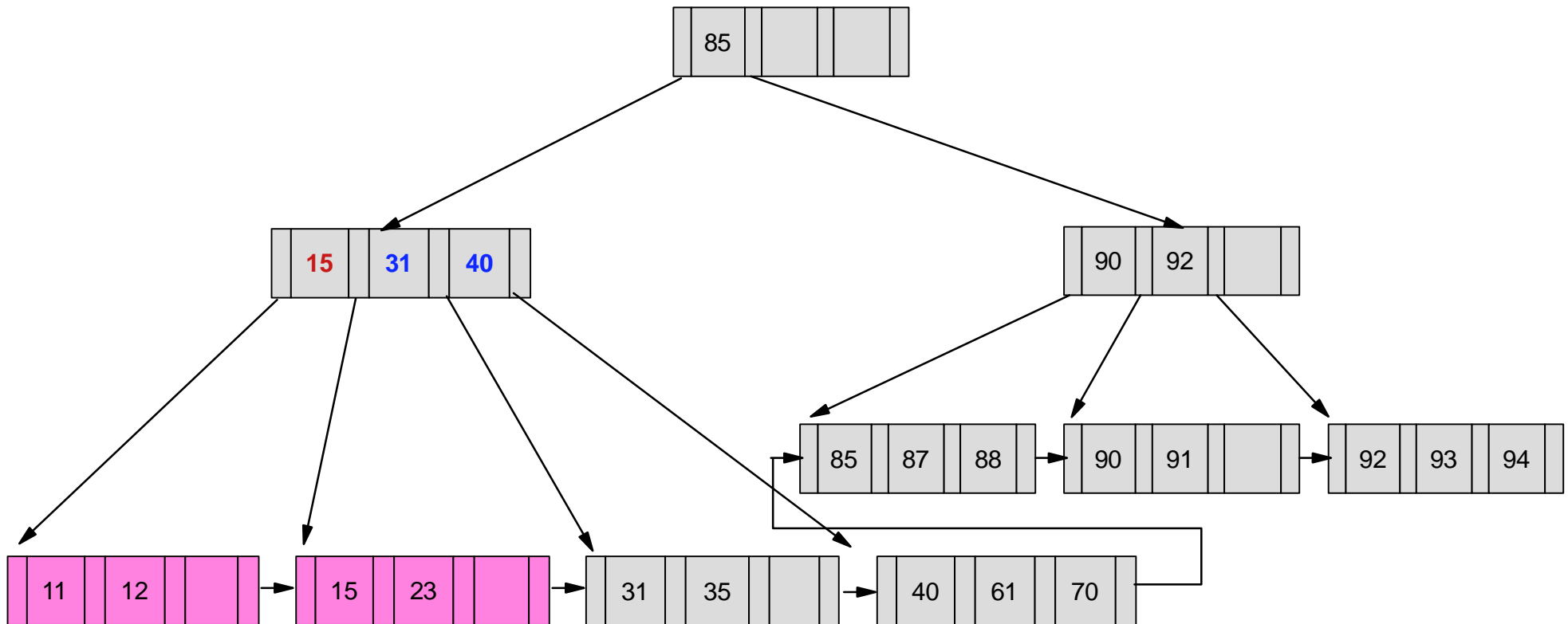


► a folha é então quebrada em duas

Árvores-B⁺

■ Inserção

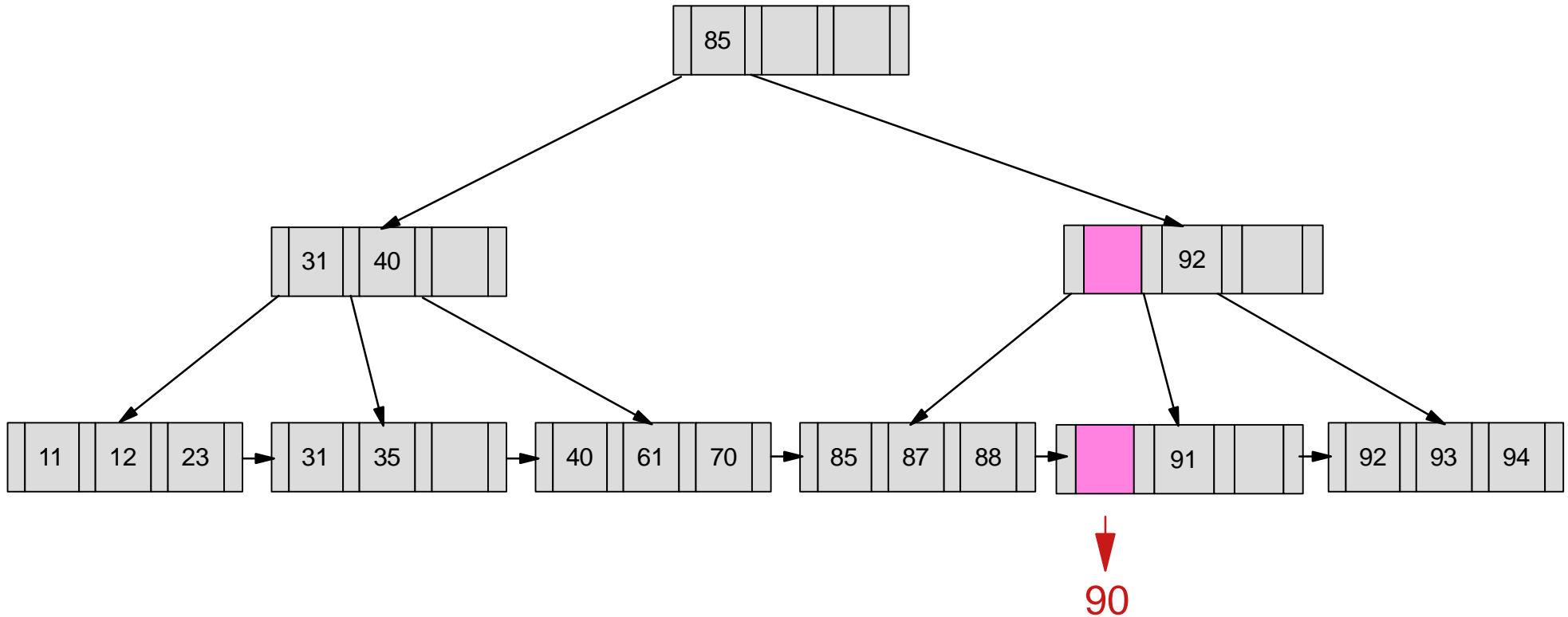
► novo valor = 15



► o primeira valor da folha da direita, que por acaso é 15, é inserido no pai, que passa a ter 1 novo filho

Árvores-B⁺

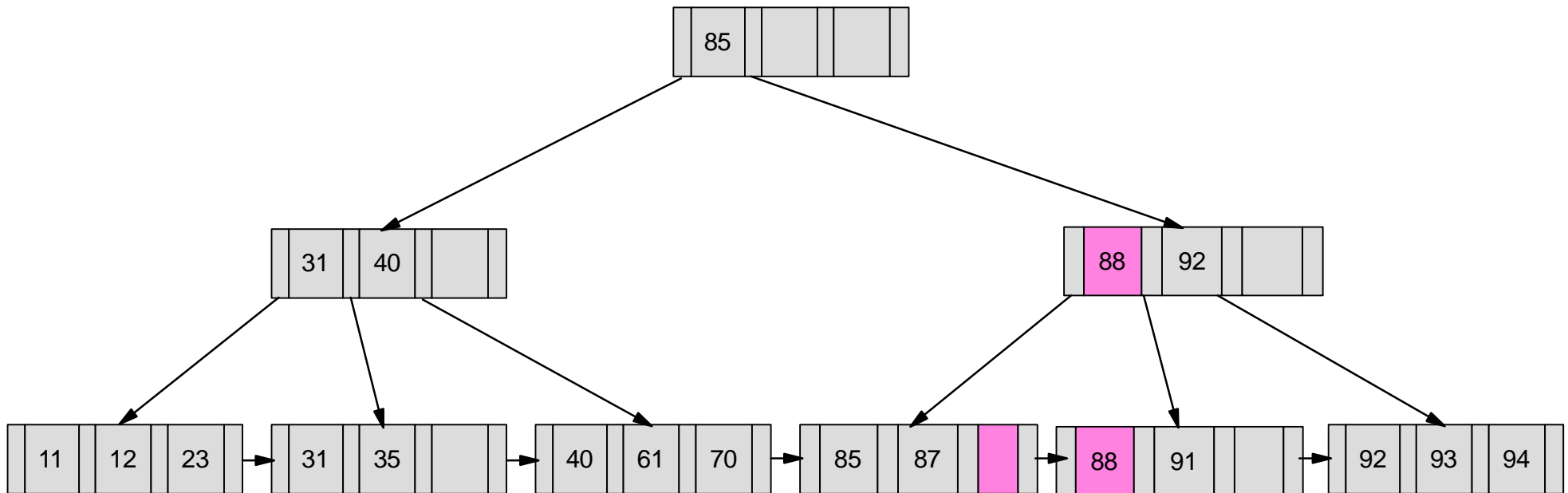
- Remoção:
 - valor removido = 90



- remove-se o valor 90 da folha correta,
que neste caso ficará pequena demais

Árvores-B⁺

- Remoção:
 - ▶ valor removido = 90

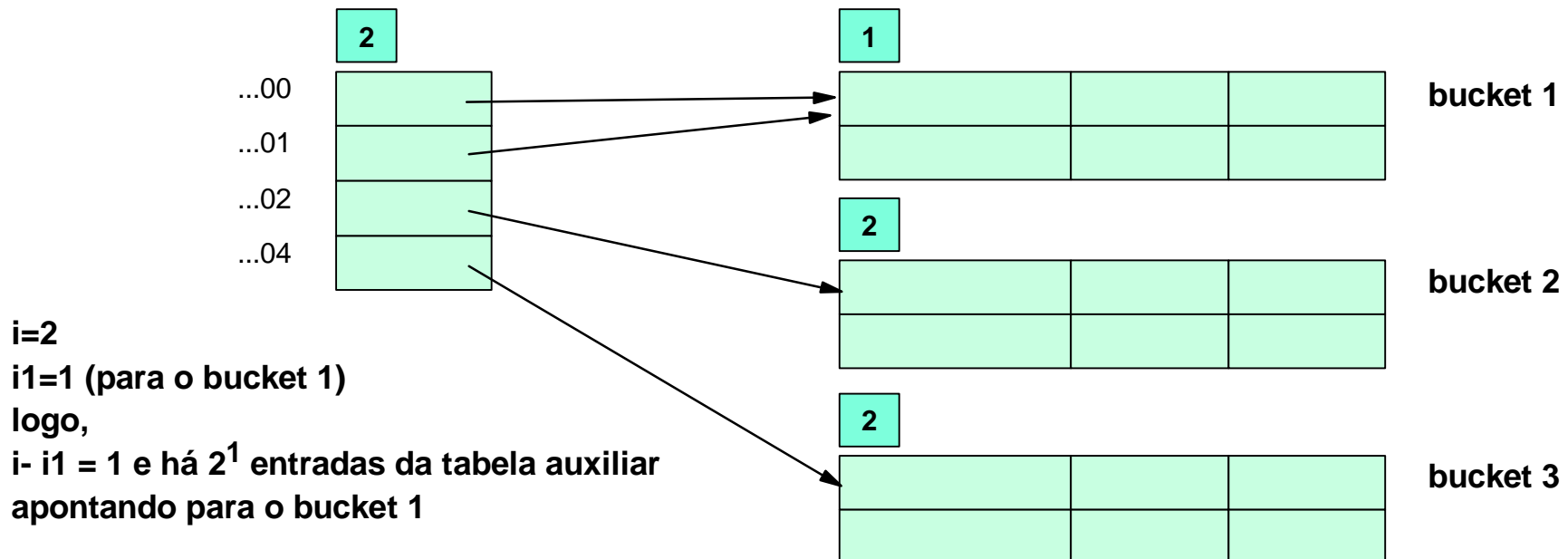


- ▶ o último valor da folha da esquerda, que neste caso é 88, é inserido na folha da direita, que passa a ter o tamanho correto

Hashing Expansível

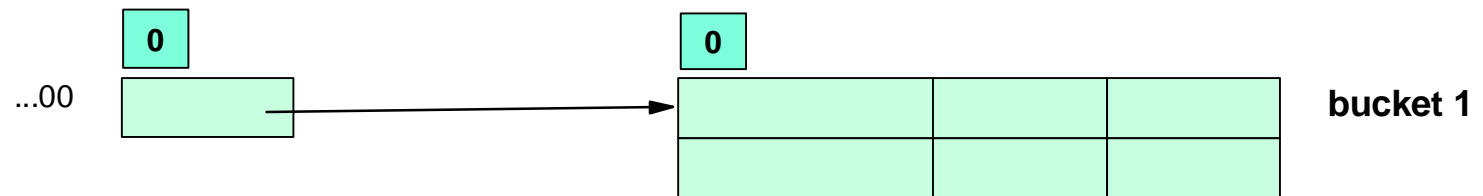
■ Definição

- ▶ função de hash gera inteiros binários com b bits
- ▶ i bits do início do valor do hash são usados como índice para uma tabela auxiliar para endereçamento dos buckets
- ▶ o valor de i cresce ou decresce com o BD
- ▶ cada bucket j possui um inteiro i_j associado que define quantas entradas da tabela auxiliar apontam para ele

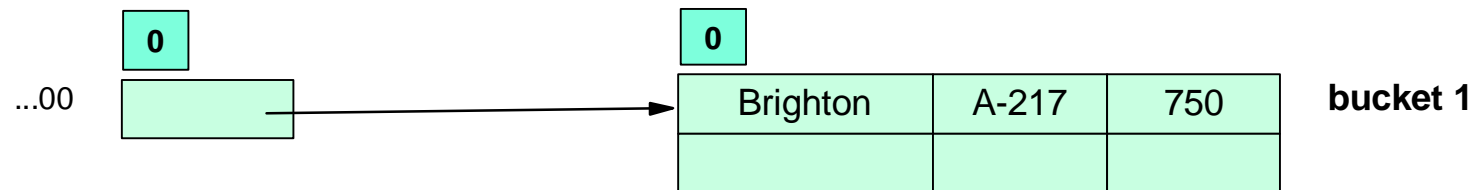


Hashing Expansível

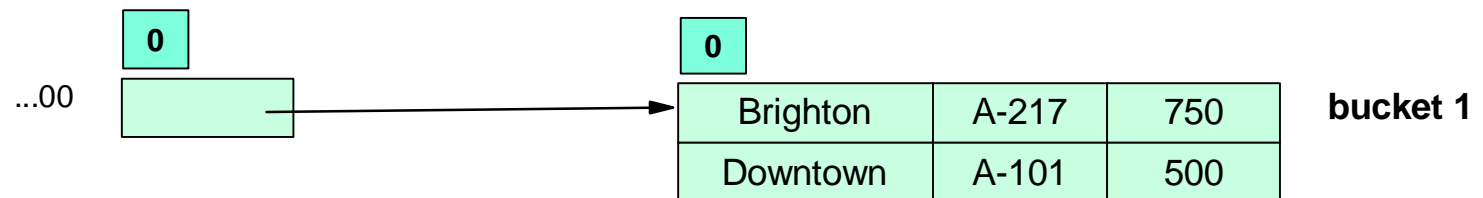
■ Inserção



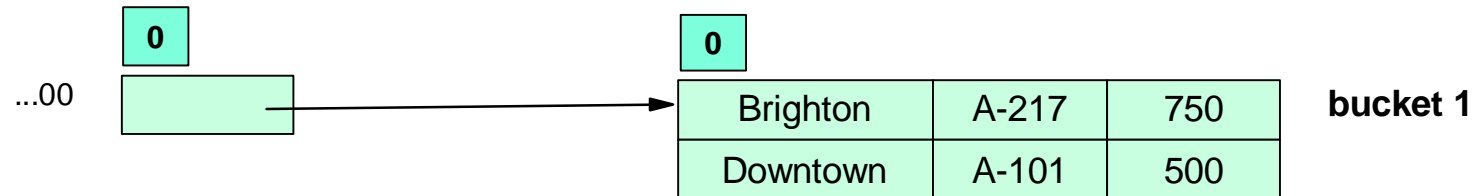
->(Brighton,A-217,750) $h(\text{Brighton})=0010$



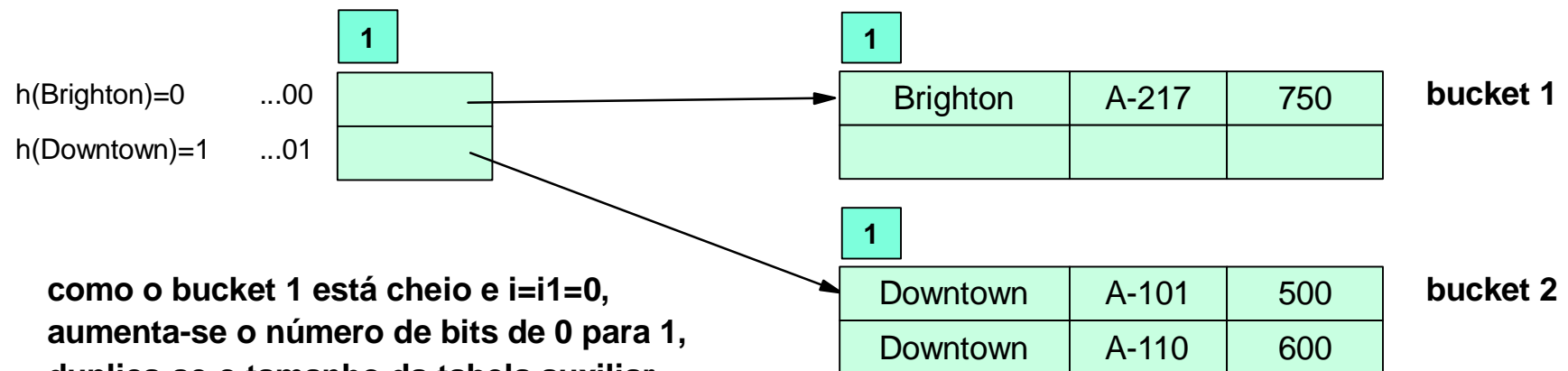
->(Downtown,A-101,500) $h(\text{Downtown})=1010$



Hashing Expansível

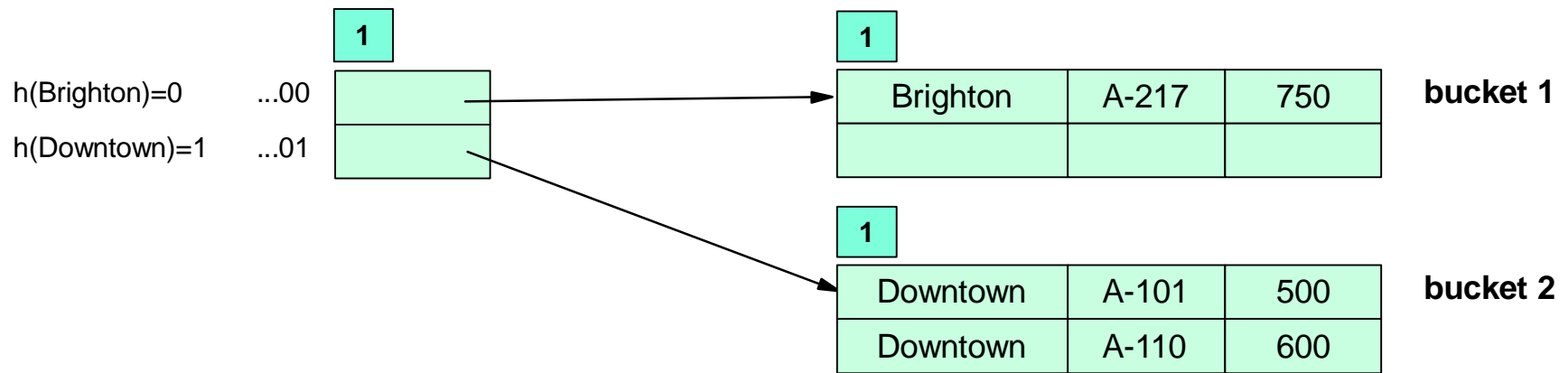


->(Downtown,A-110,600) $h(\text{Downtown})=1010$

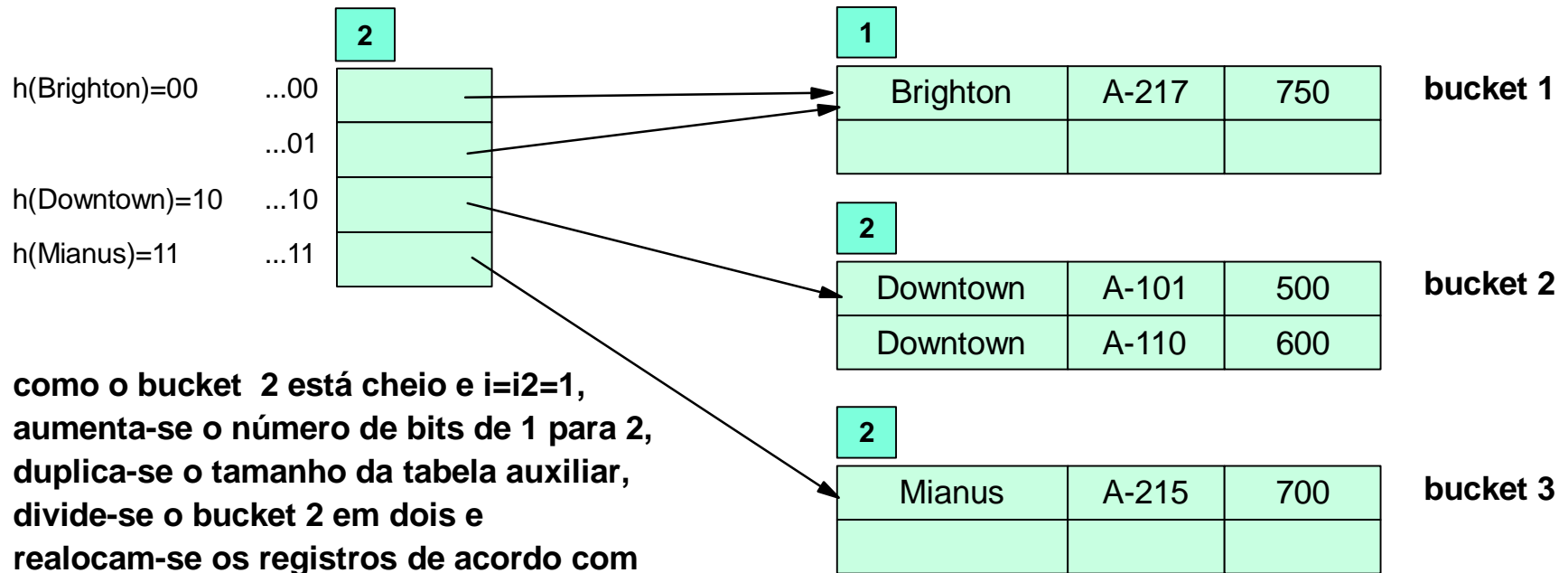


como o bucket 1 está cheio e $i=i1=0$,
aumenta-se o número de bits de 0 para 1,
duplica-se o tamanho da tabela auxiliar,
divide-se o bucket 1 em dois e
realocam-se os registros de acordo com
o primeiro bit da função de hash

Hashing Expansível



->(Mianus,A-215,700) h(Mianus)=1100



como o bucket 2 está cheio e $i=i_2=1$,
aumenta-se o número de bits de 1 para 2,
duplica-se o tamanho da tabela auxiliar,
divide-se o bucket 2 em dois e
realocam-se os registros de acordo com
os 2 primeiros bits da função de hash

Hashing Expansível

->(Perryridge,A102,400) $h(\text{Perryridge})=1111$

->(Perryridge,A201,900)

->(Perryridge,A218,700)

