Árvores B

Prof. Márcio Bueno
ed2tarde@marciobueno.com / ed2noite@marciobueno.com

Fonte: Material da Profª Ana Eliza Lopes Moura

Situação Problema

- Memória Principal
  - Volátil e limitada
- Aplicações
  - Grandes quantidades de informação
  - Armazenamento permanente de informações
- Chaves mantidas em memória secundária

Situação Problema

- Árvores de Busca Binária
  - Apropriada para memória principal
  - Ineficiente em memória secundária
    - Acesso: cerca de $\log_2 n$ passos
    - Grande quantidade de acessos a disco
      - Acesso feito em blocos

Situação Problema

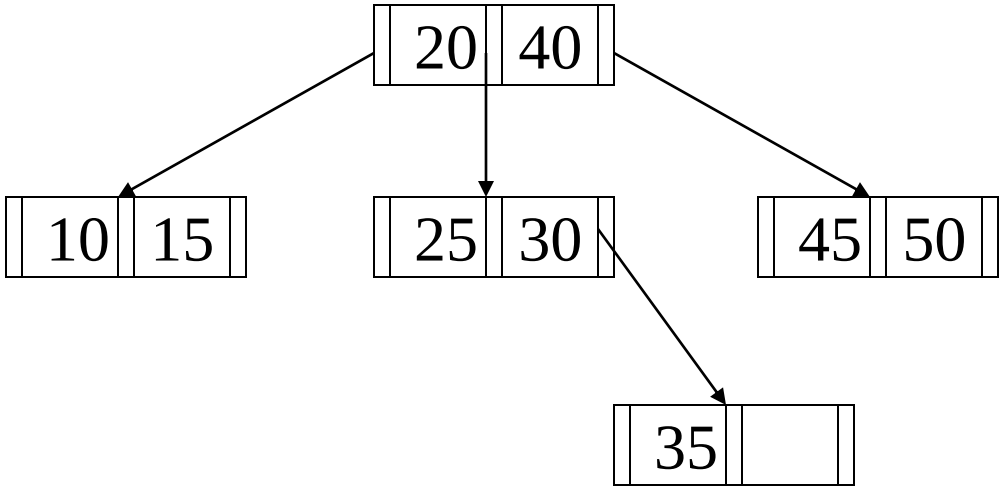
- Necessidade
  - Reduzir o número de acessos a disco
- Solução
  - Agrupar várias chaves dentro de um nó
    - Obter com o mesmo acesso várias chaves
    - Reduzir o número de acessos
    - Diminuir o tempo necessário para inserções, remoções e pesquisas.

Árvores Multivias ou M-Vias

- Definição
  - Uma árvore de busca multivias de ordem **M** é uma árvore n-ária na qual todos os nós têm grau menor ou igual a **M**.
  - Um nó com **M** descendentes contém **M-1** valores de chave.

Árvores Multivias ou M-Vias

- Exemplo: M = 3



Árvores Multivias ou M-Vias

- Desempenho da Busca
  - Árvores multivias de **N** chaves e fator de ramificação **S**.
  - Caminho médio de busca: $O(\log_S N)$
  - Se $N = 10^6$ e $S = 100$, então uma busca requer, em média, $\log_{100} 10^6 = 3$ passos.
  - Árvore de busca binária: $\log_2 10^6 = 20$ passos.

Árvores Multivias ou M-Vias

- Problema
  - Inserções aleatórias de maneira irrestrita
  - Aumento do caminho de busca
- Solução
  - Balanceamento

Árvores B

- Definição
  - Bayer e McCreight em 1970.
  - Uma árvore B de ordem **M** é uma árvore de busca multivias **balanceada**.
  - Uma árvore B ou está vazia ou possui nós com **K** apontadores e **K-1** chaves.

  - OBS: Um nó de uma árvore B é chamado de **página**.

Árvores B

- Utilização
  - Árvores B são utilizadas como forma de armazenamento em diversos sistemas de BD comerciais.

Árvores B

- Características Estruturais:
  - Na raiz, **K** deve ser, no mínimo, 2.
    - Ou seja, a raiz possui no mínimo **dois** filhos e **uma** chave.
  - Nos demais nós, K deve ser, no mínimo, **M/2**.
    - Ou seja, os demais nós possuem, no mínimo, **M/2** filhos e **M/2-1** chaves;
    - Exceção: Folhas não têm filhos.

Árvores B

- Características Estruturais (cont.):
  - O valor máximo de K é M
    - Ou seja, todos os nós têm, no máximo, **M-1** chaves e **M** filhos;
  - Todas as folhas estão no mesmo nível (balanceamento).
  - **OBS**: **M** deve ser escolhido de forma que o número máximo de chaves nos nós da árvore seja uma potência de 2

Árvores B

- Características Estruturais (cont.)
  - Formato do nó:

    N, A_0, (C_1A_1), (C_2A_2), ..., ($C_{M-1}A_{M-1}$) onde:
    - N, **M/2 ≤ N ≤ M**, é o número de entrada ativas (ocupadas) de um nó em um dado momento;
    - A_i, $0 \leq i \leq M-1$, é um apontador para uma subárvore;
    - C_i, $1 \leq i \leq M-1$, é um valor de chave e $C_i < C_{i+1}$;
    - O par (C_iA_i) é chamado de entrada;
    - O apontador A_0 também é definido como entrada.

Árvores B

- Características Estruturais (cont.)

  - Definição do nó:

```
typedef char Tipo;

struct no {
  int n;
  Tipo chv[M-1];
  no* pont[M];
};
```

Árvores B

- Características (cont.):
  - Seja uma página com **D** chaves:
    - Para qualquer chave **y**, pertencente à página apontada por A_0, $y < C_1$;
    - Para qualquer chave **y**, pertencente à página apontada por A_i, $1 \leq i \leq D-1$, $C_i < y < C_{i+1}$;
    - Para qualquer chave **y**, pertencente à página apontada por A_D, $y > C_D$.

Árvores B

- Comparação em termos de nós e chaves por nível entre uma árvore binária e uma árvore B de ordem **M** de mesma altura.

Nível	Binária	Árvore B
0	1 nó	1 nó x M-1 chaves
1	2 nós	M nós x (M-1) chaves
2	4 nós	M x M nós x (M-1) chaves
3	8 nós	M x M x M nós x (M-1) chaves
...	...	...
n	2^n nós	M^n nós x (M-1) chaves

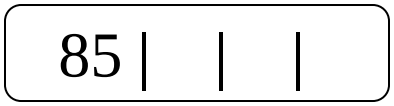
Árvores B

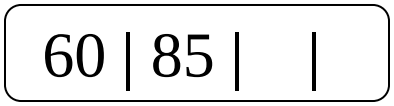
- Observações:
  - A ordem **M** determina as quantidades máximas e mínimas de chaves dentro de cada nó.
  - O número mínimo de chaves é estabelecido para determinar o percentual mínimo de ocupação dentro de um nó. Na árvore B esse percentual é de 50% (não considerando a raiz).

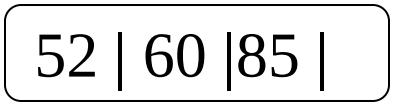
Árvore B

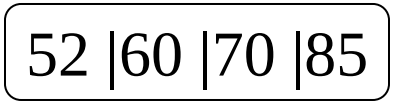
- Inserção
  - Em uma árvore B, a inserção de uma nova chave ocorre sempre em um nó folha.
  - Passos:
    - Localizar a folha dentro da qual a chave deve ser inserida;
    - Se a folha não estiver completa, inserir chave na ordem correta;
    - Se a folha estiver completa, realizar a **cisão da página**.

Árvore B

- Inserção: Exemplo - M = 5
  - Inserir chave 85 

  - Inserir chave 60 

  - Inserir chave 52 

  - Inserir chave 70 

  - Inserir chave 58 ⇦ ***Realizar cisão***

Árvore B

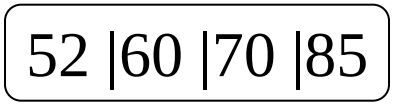
- Inserção
  - Cisão de Página
    - O processo de cisão consiste em separar a folha completa em duas: folha esquerda e folha direita.

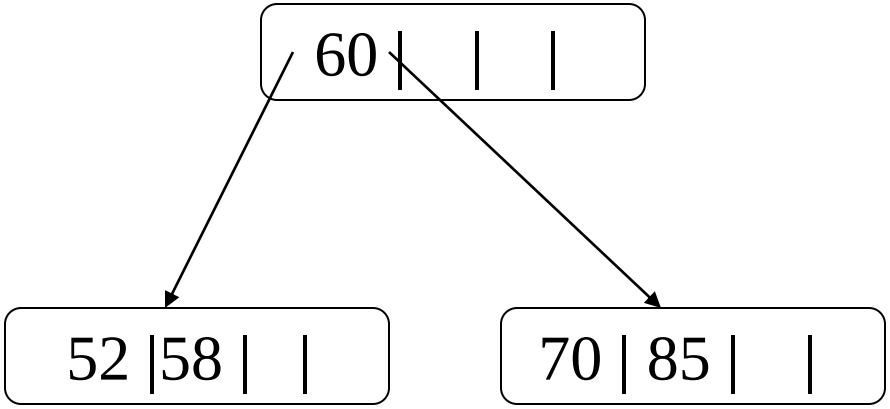
Árvore B

- Inserção -> Cisão de Página
  - As **M** chaves serão divididas em três grupos:
    - As (**M / 2**) chaves menores ficam na folha esquerda;
    - As (**M / 2**) chaves maiores ficam na folha direita;
    - A chave do meio é colocada no nó pai, se possível.
  - Obs.: A divisão é inteira

Árvore B

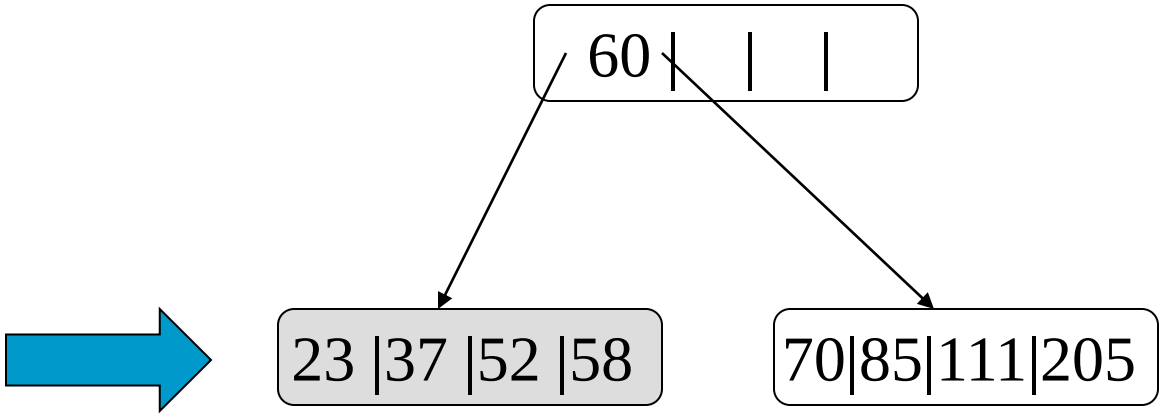
- Inserção (Exemplo - cont.)
  - Inserir chave 58 (antes)
  - Inserir chave 58 (depois)





Árvore B

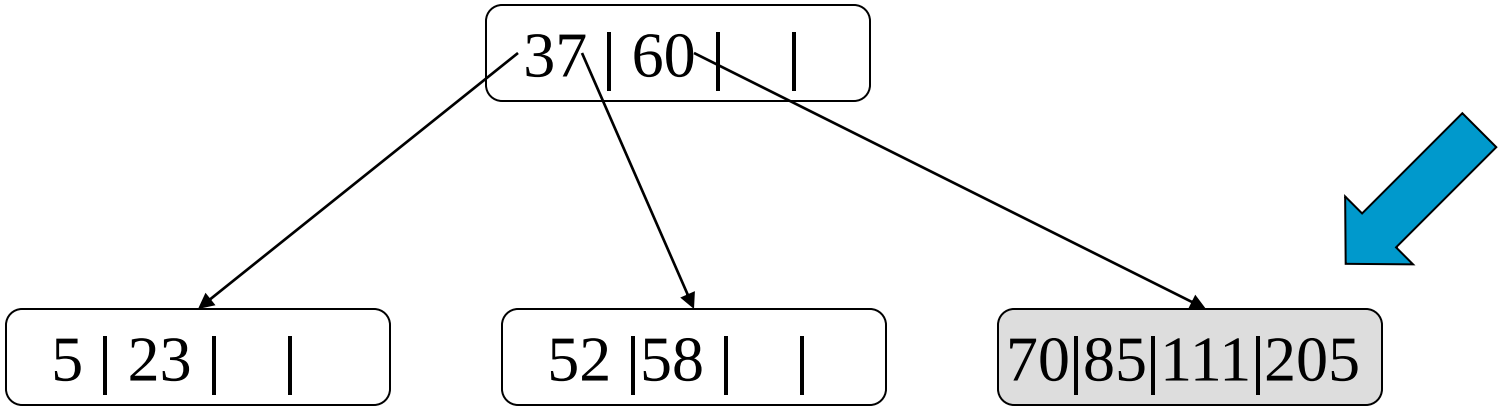
- Inserção (Exemplo - cont.)
  - Inserir chaves 37, 111, 23, 205



  - Inserir chave 5 ⇦ ***Realizar cisão***

Árvore B

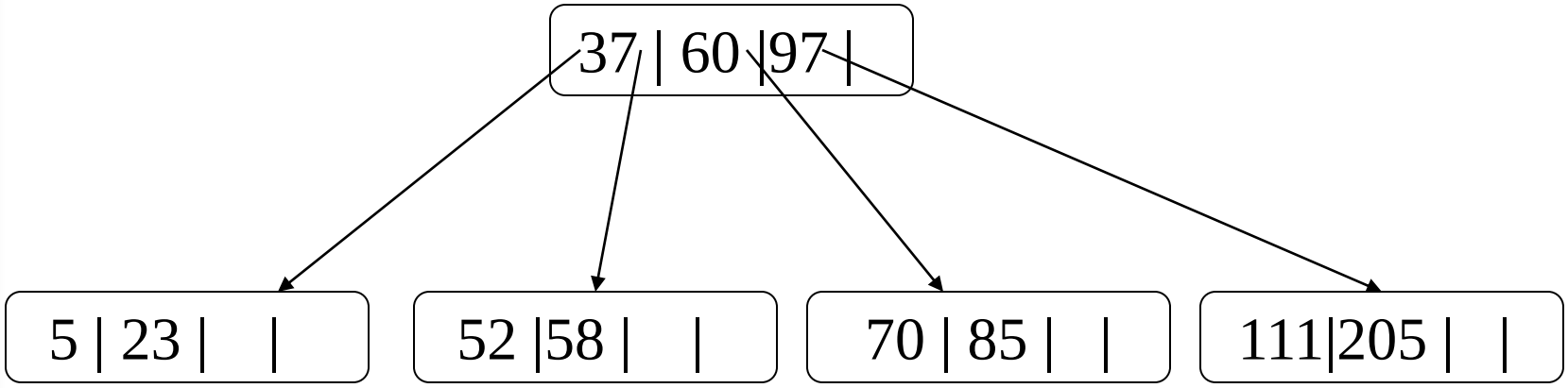
- Inserção (Exemplo - cont.)
  - Inserir chave 5 (depois)



  - Inserir chave 97 ⇦ ***Realizar cisão***

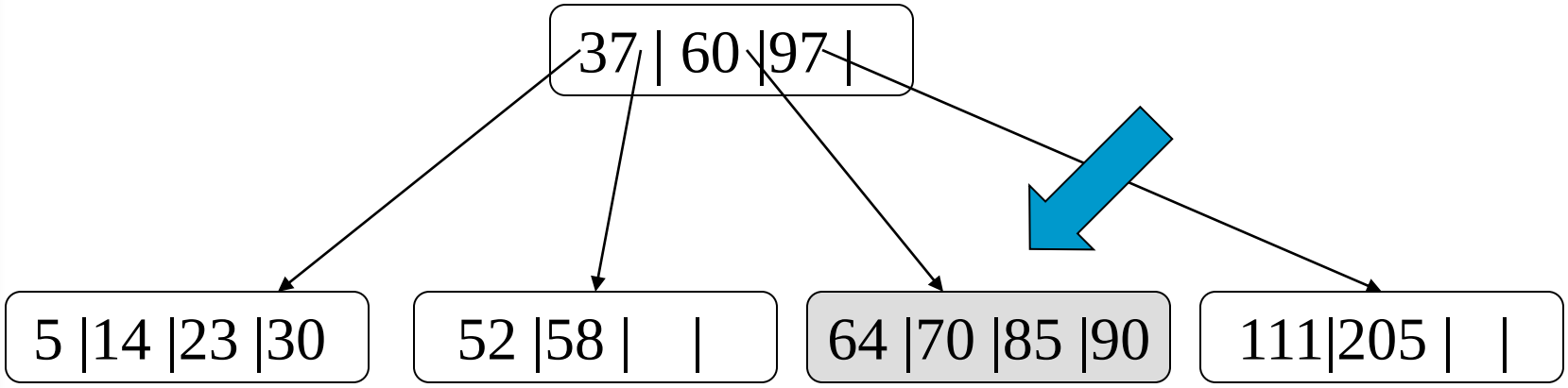
Árvore B

- Inserção (Exemplo - cont.)
  - Inserir chave 97 (depois)



Árvore B

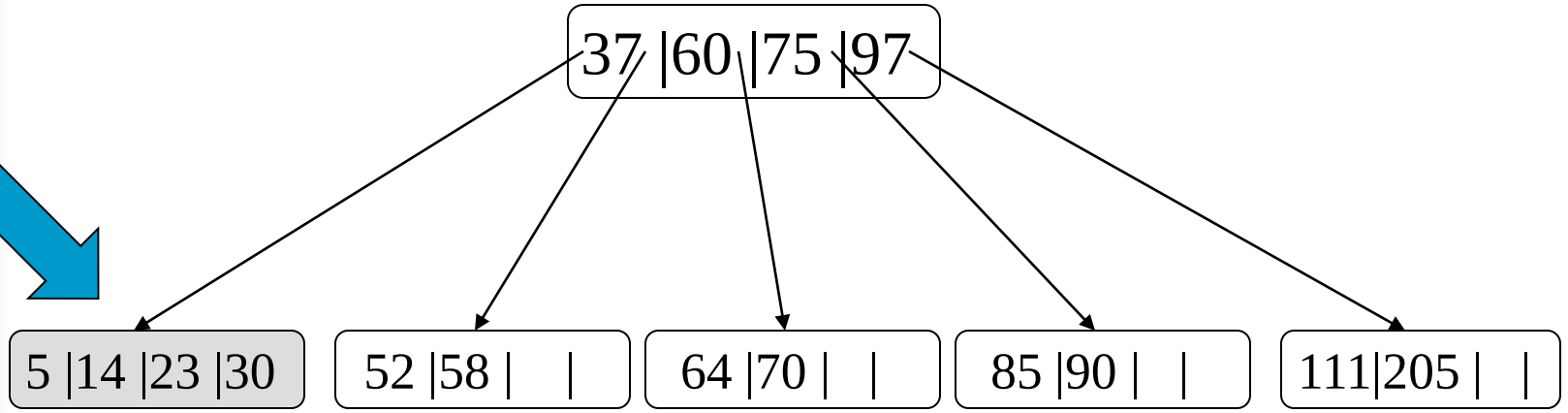
- Inserção (Exemplo - cont.)
  - Inserir chaves 64,14, 90, 30



  - Inserir chave 75 ⇦ ***Realizar cisão***

Árvore B

- Inserção (Exemplo - cont.)
  - Inserir chave 75 (depois)



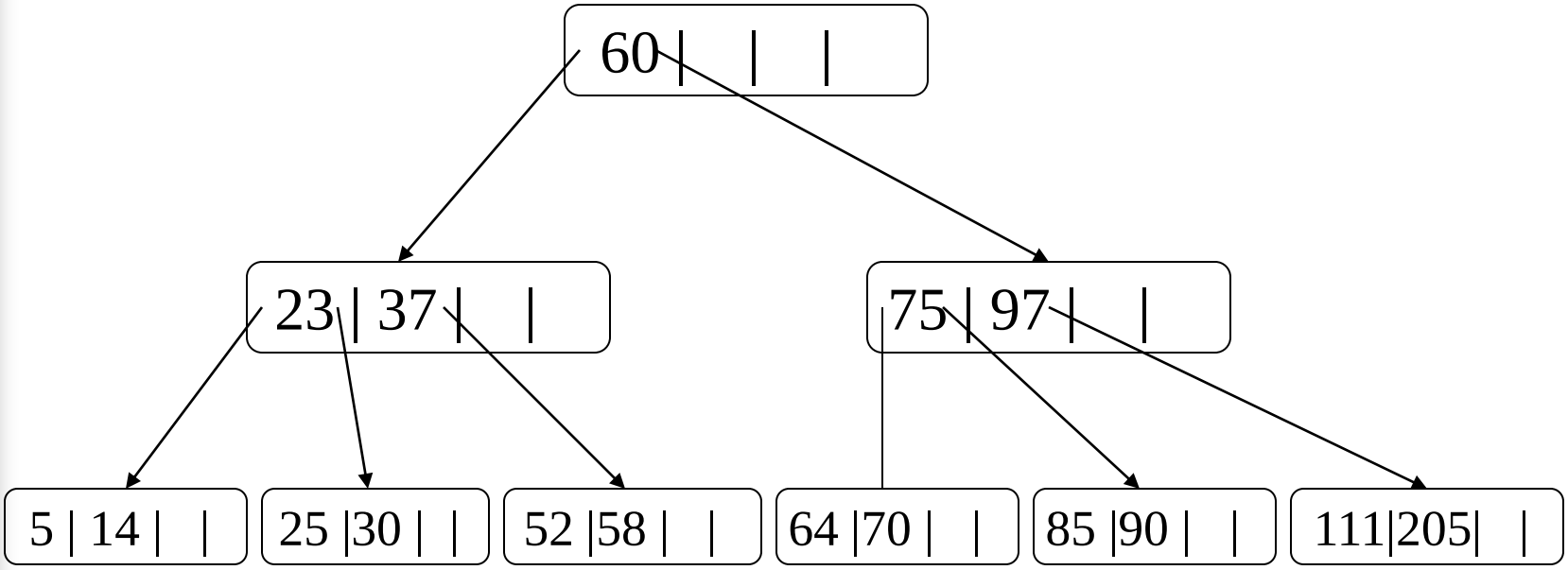
  - Inserir chave 25 ⇦ ***Realizar cisão***

Árvore B

- Inserção
  - A inserção da nova entrada no nó pai pode acarretar a necessidade de uma nova cisão;
  - A cisão de páginas é propagável, podendo atingir até mesmo a raiz da árvore.
  - Neste caso, surge uma nova raiz, o que implica em alteração da altura da árvore.
  - Após o processo de inserção, a árvore permanece balanceada.

Árvore B

- Inserção (Exemplo - cont.)
  - Inserir chave 25 (depois)

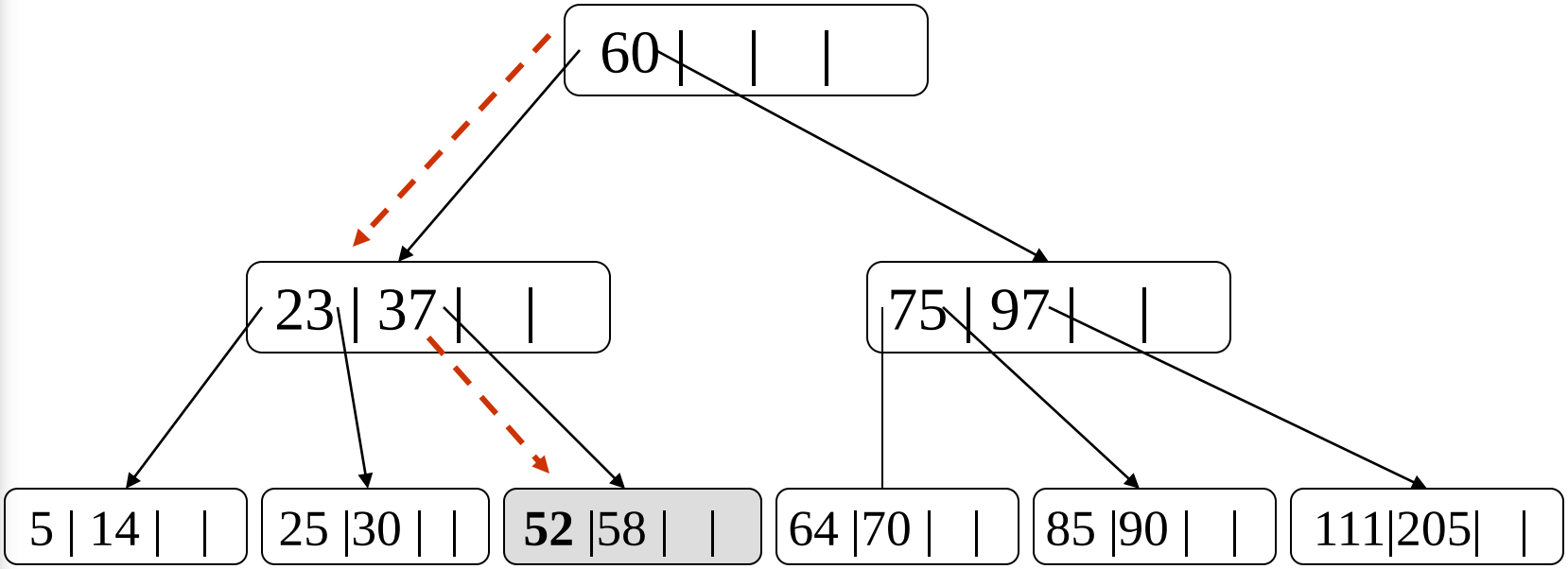


Árvores B

- Consulta
  - Verifica se a chave procurada está na raiz;
  - Caso não esteja, se a chave for menor que a chave C_i, $1 \leq i \leq N-1$, então repetir a pesquisa na subárvore A_{i-1};
  - A pesquisa termina quando encontramos a chave ou um apontador A_i igual a nulo.

Árvore B

- Consulta – Exemplo:
  - Procurar chave 52



Árvores B

- Consulta - Algoritmo:

```
void BuscaB(Tipo x, no *raiz, no *&pt, bool &f, int &g) {
  no *p = raiz; pt = null; f = false;
  while (p != null) {
    int qtd = p->n, i; i = g = 0; pt = p;
    while (i < qtd)
      if (x > p->chv[i]) {
        i = g = i + 1;
       } else if (x == p->chv[i]) {
         f = true; return;
       } else {
         p = p->pont[i]; i = qtd + 1;
       }
    if (i == qtd)
       p = p->pont[qtd];
  }
}
```

Árvore B

- Consulta - Algoritmo:
  - Os parâmetros **pt**, **f** e **g** fornecem o resultado da busca.
  - Se a chave for encontrada na tabela, **f** é verdadeiro, **pt** contém o endereço da página que contém a chave e **g** contém a posição da chave dentro da página.
  - Se a chave não for encontrada, **f** continua falso, **pt** aponta para a última página examinada e **g** informa a posição, nesta página, onde a chave seria incluída.

Árvore B

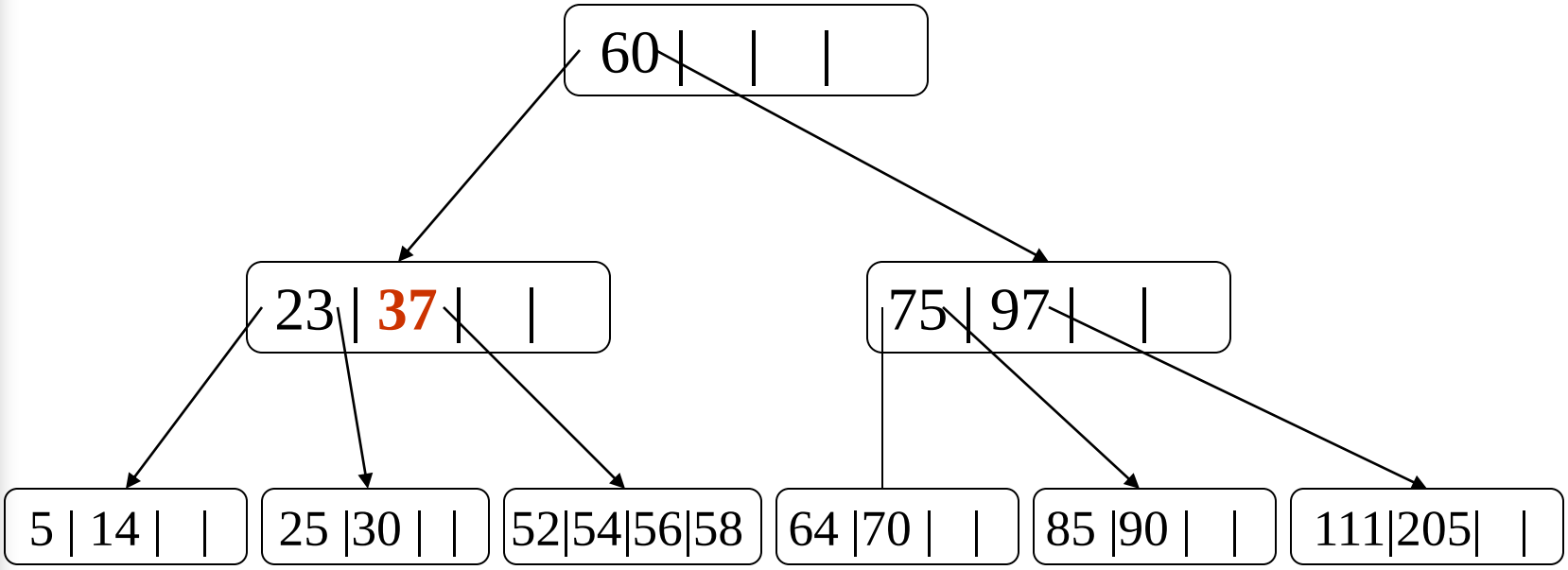
- Consulta – Algoritmo:
  - A pesquisa dentro de um nó é seqüencial.
  - Se a ordem da árvore for maior que 10, devemos considerar a utilização de pesquisa binária.

Árvore B

- Remoção de uma chave X
  - **Caso 1**: A chave **X** não se encontra em uma folha
    - **X** é substituída pela chave **Y**, imediatamente maior;
    - **Y** necessariamente pertence a uma folha.

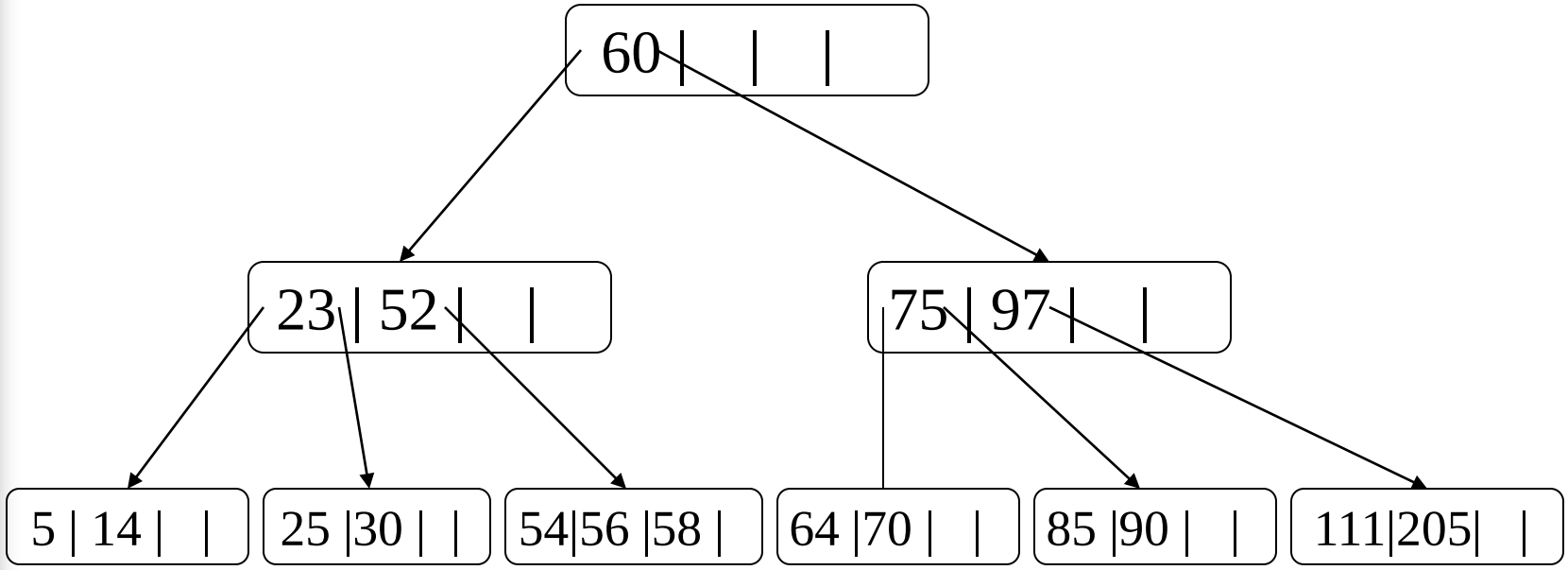
Árvore B

- Remoção (Exemplo)
  - **Caso 1**: Remover a chave 37 (antes)



Árvore B

- Remoção (Exemplo)
  - **Caso 1**: Remover a chave 37 (depois)

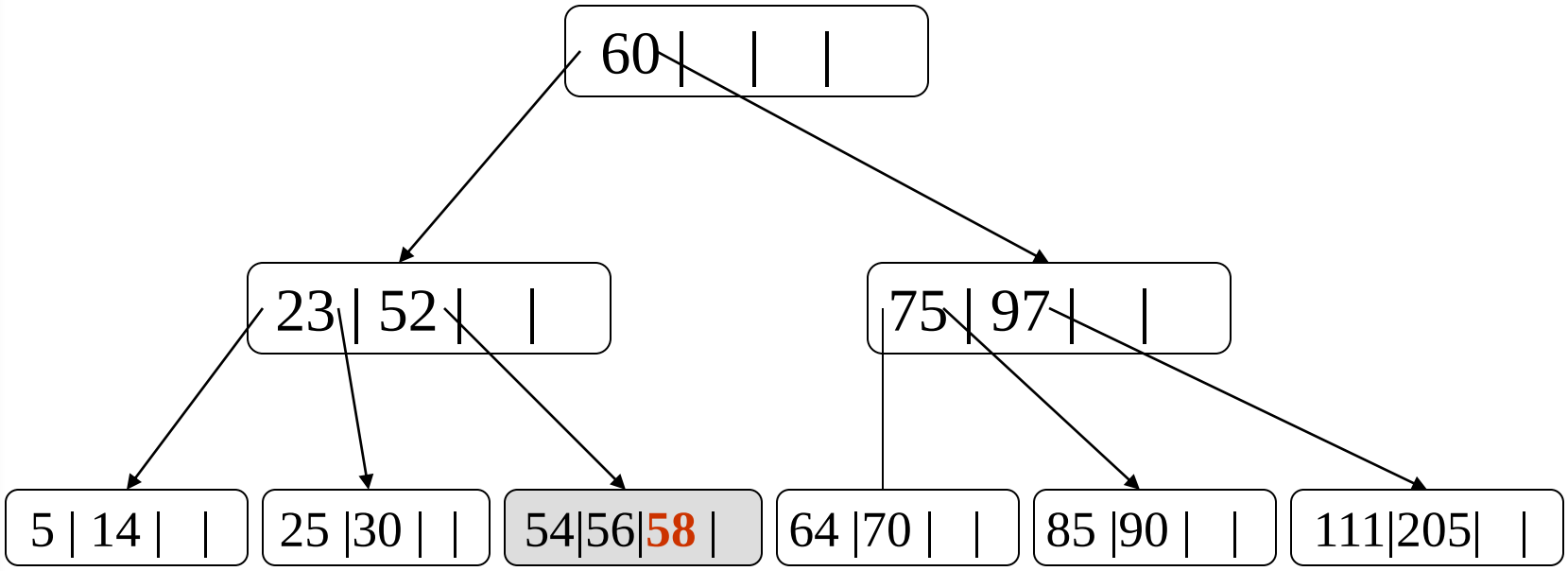


Árvore B

- Remoção
  - **Caso 2**: A chave X se encontra em uma folha
    - A chave é simplesmente removida.

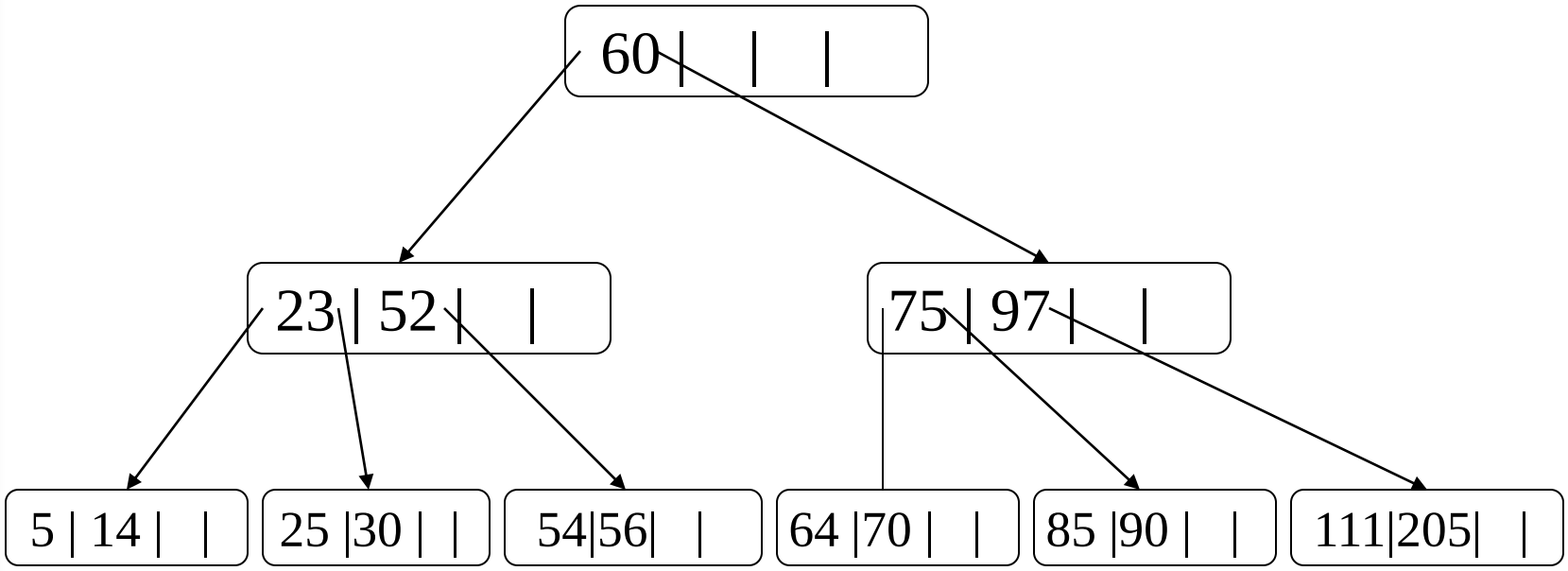
Árvore B

- Remoção (Exemplo)
  - **Caso 2**: Remover a chave 58 (antes)



Árvore B

- Remoção (Exemplo)
  - **Caso 2**: Remover a chave 58 (depois)



Árvore B

- Remoção
  - Quando uma chave é retirada de um nó folha, o número de chaves restantes pode ser menor que **(M-1)/2**.
  - Tratamentos:
    - **Concatenação**
    - **Redistribuição**

Árvore B

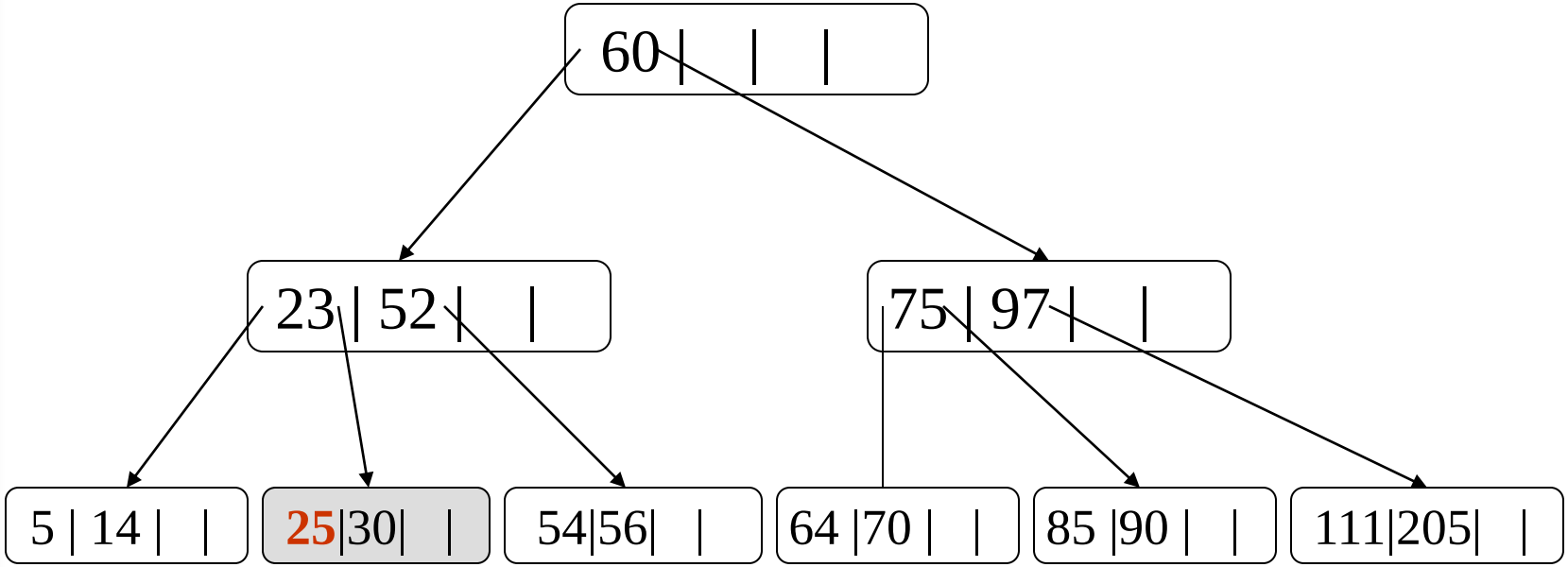
- Remoção com Concatenação
  - Duas páginas **P** e **Q** são chamada **irmãos adjacentes** se têm o mesmo pai **W** e são apontadas por ponteiros adjacentes em **W**.
  - **P** e **Q** podem ser concatenadas se são irmãos adjacentes e juntas possuem menos de **M-1** chaves.

Árvore B

- Remoção com Concatenação
  - A concatenação agrupa as entradas de duas páginas em uma só;
  - No nó pai deixa de existir uma entrada: aquela da chave que se encontra entre os ponteiros para **P** e **Q**.
  - Essa chave passa a fazer parte do nó concatenado e seu ponteiro desaparece.

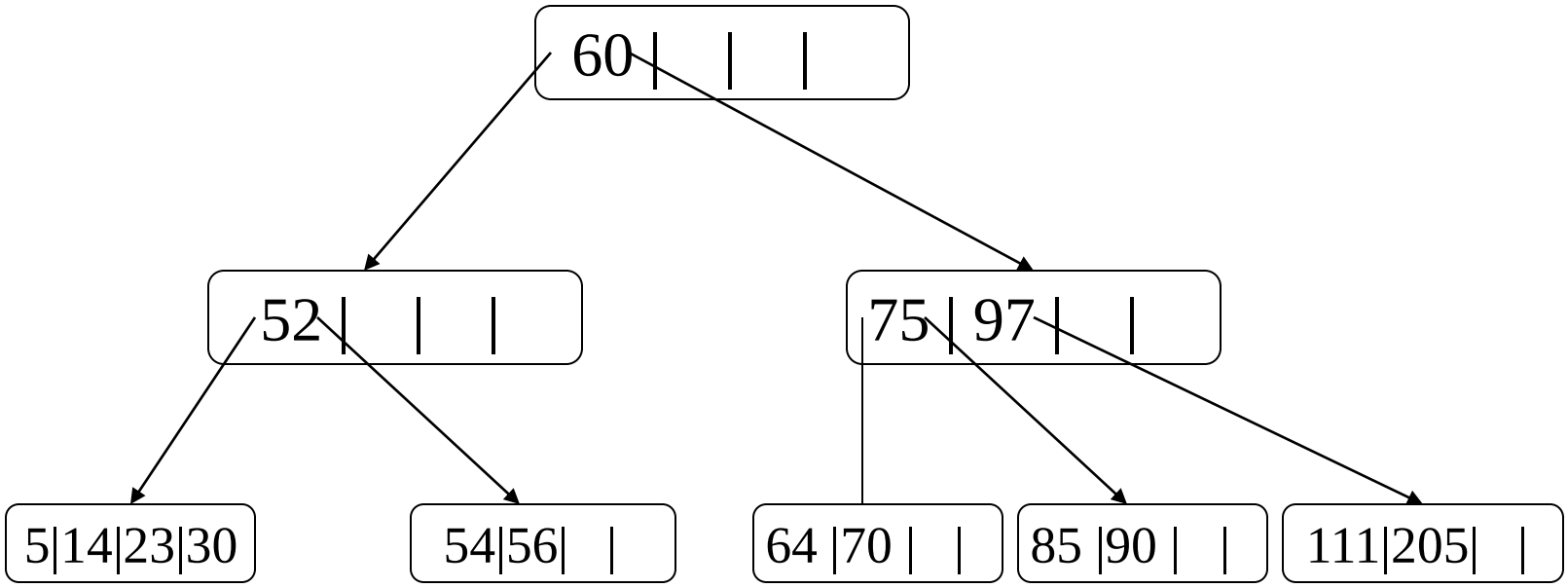
Árvore B

- Remoção com Concatenação
  - Exemplo: Remover a chave 25 (antes)



Árvore B

- Remoção com Concatenação
  - Exemplo: Remover a chave 25 (depois)



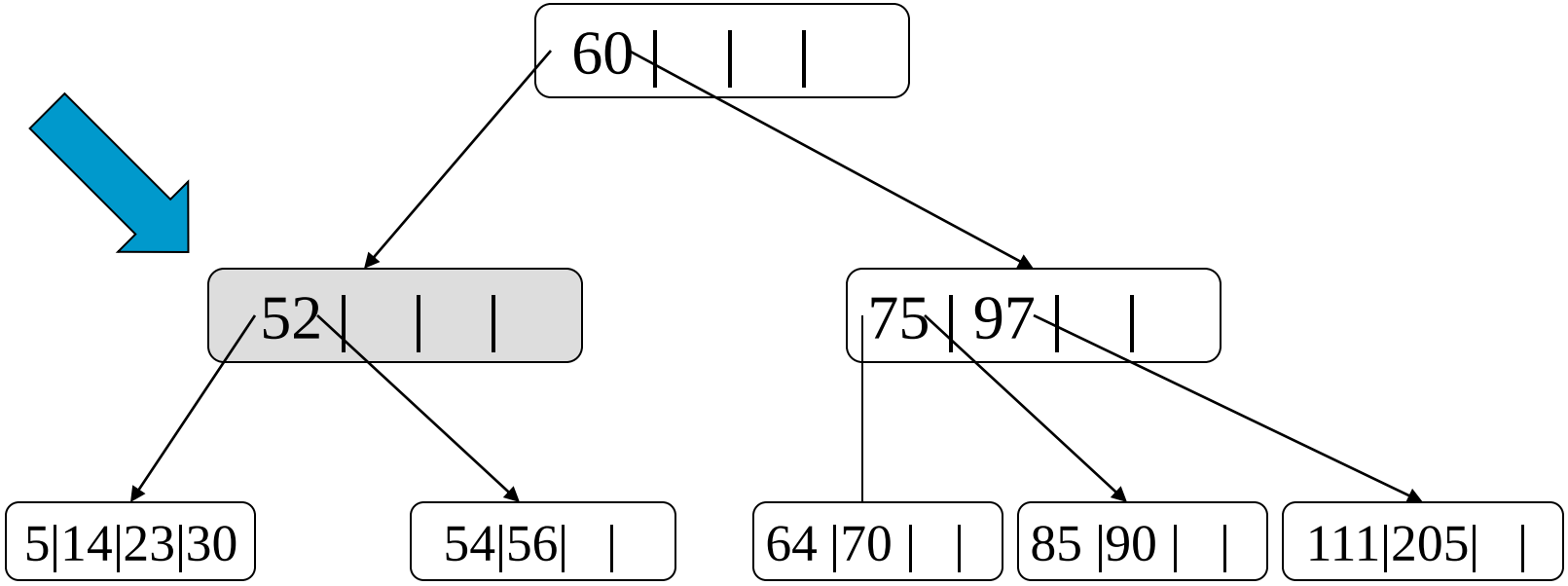
Árvore B

- Remoção com Concatenação
  - Como foi retirada uma chave do nó **W**, caso ele passe a ter menos de **(M-1)/2** chaves, o processo se repete;
  - Ou seja, a concatenação é um processo propagável;
  - Se a propagação atingir a raiz, a árvore diminuirá de altura.

Árvore B

- Remoção com Concatenação
  - Exemplo: Remover a chave 25 (cont.)

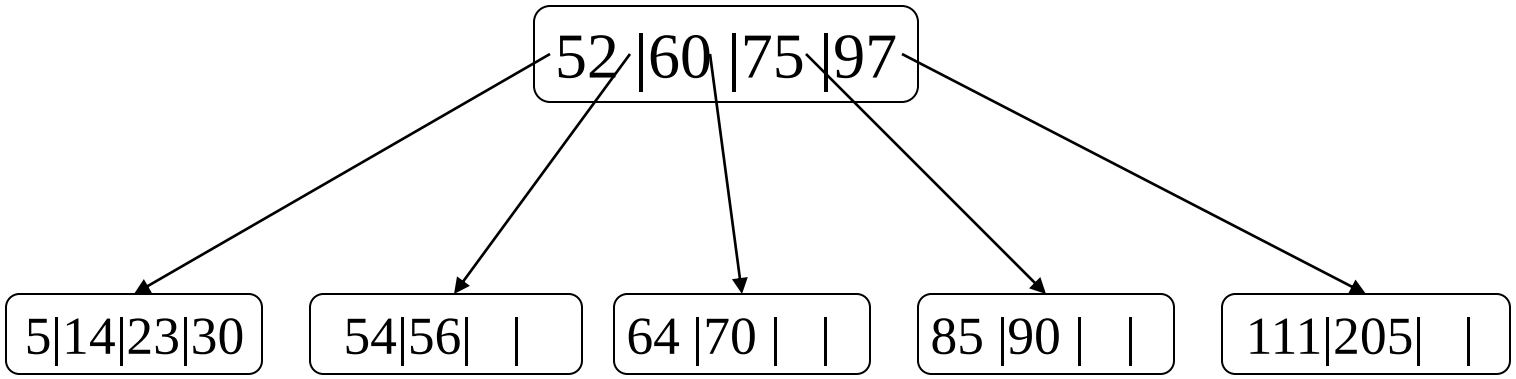
⇨ **Propagação**



Árvore B

- Remoção com Concatenação
  - Exemplo: Remover a chave 25 (cont.)

⇨ **Propagação**

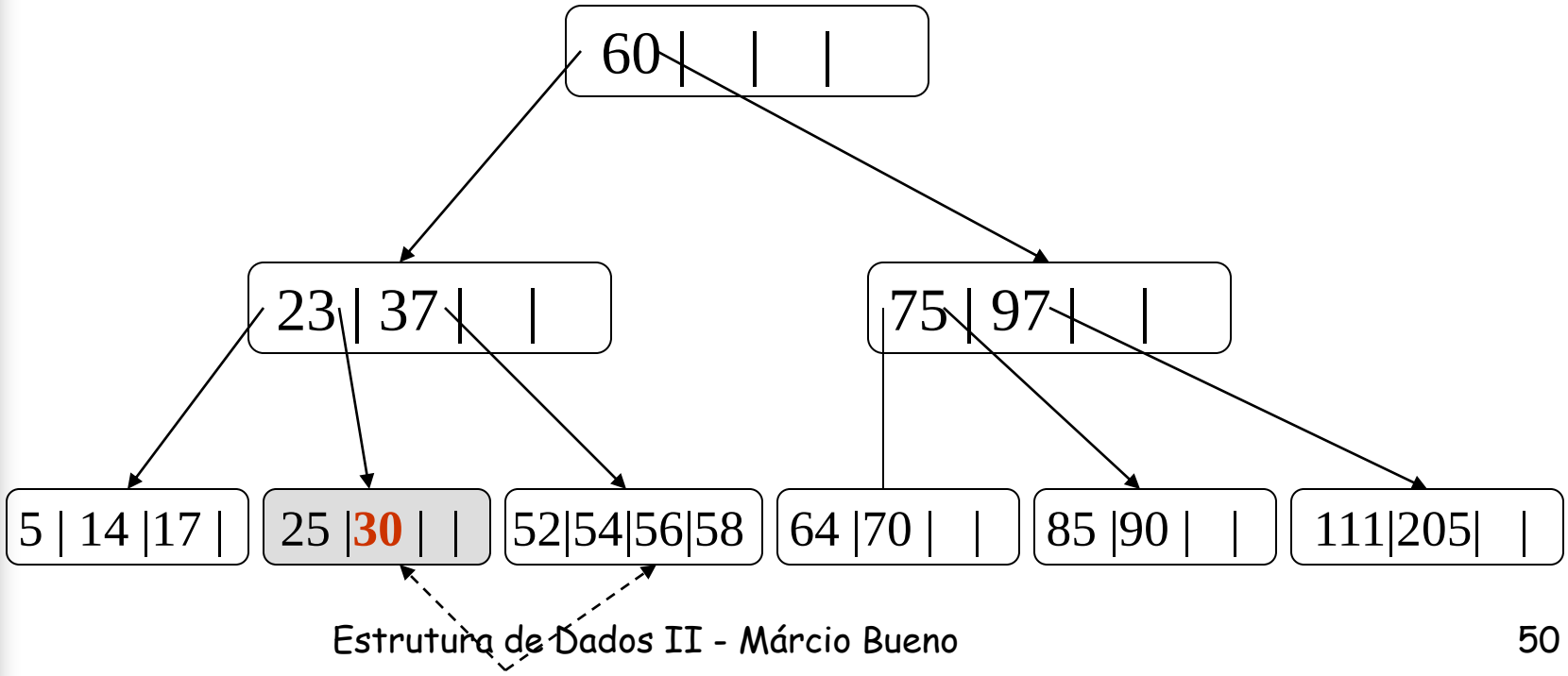


Árvore B

- Remoção com Redistribuição
  - Se a página **P** e seu irmão adjacente **Q** possuem em conjunto **M-1** ou mais chaves, estas podem ser equilibradamente distribuídas:
    - Concatena-se **P** e **Q**;
    - Efetua-se a cisão da página resultante.

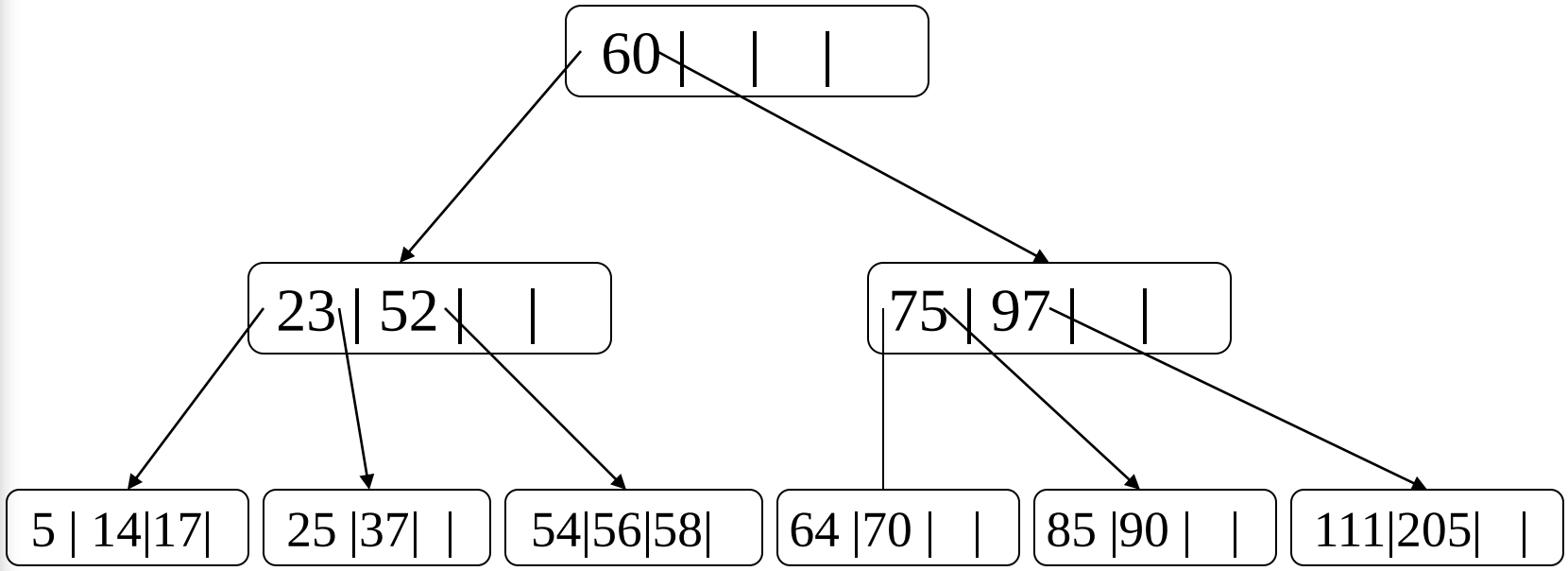
Árvore B

- Remoção com Redistribuição
  - Exemplo: Remoção da chave 30 (antes)



Árvore B

- Remoção com Redistribuição
  - Exemplo: Remoção da chave 30 (depois)



Árvore B

- Remoção com Redistribuição
  - A redistribuição não é propagável;
  - A página **W**, pai de **P** e **Q**, é modificada, mas seu número de chaves permanece o mesmo.