Árvores AVL

Estrutura de Dados II

Jairo Francisco de Souza

Introdução

As árvores binárias de pesquisa são projetadas para um acesso rápido à informação. Idealmente a árvore deve ser razoavelmente equilibrada e a sua altura será dada (no caso de estar completa) por $h=\log_2(n+1)$

O tempo de pesquisa tende a $O(\log_2 N)$

Porém, com sucessivas inserções de dados principalmente ordenados, ela pode se degenerar para O(n)

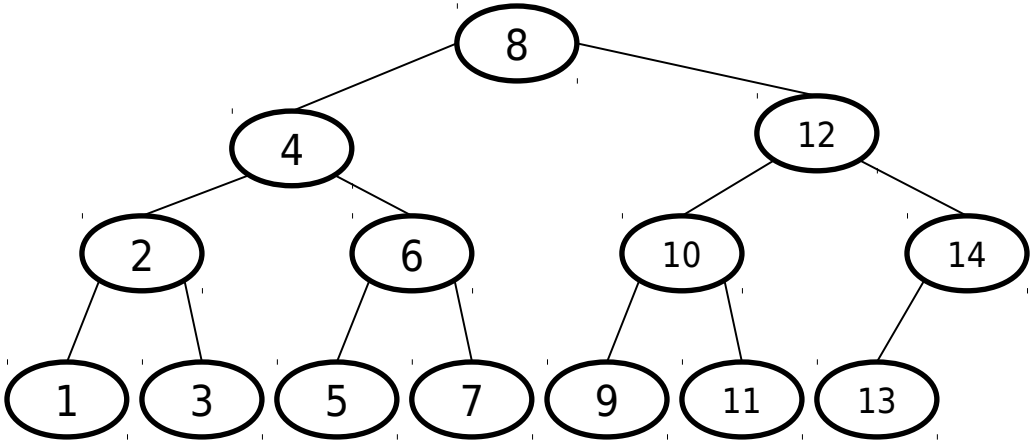
Conceito de balanceamento

Árvores completas são aquelas que minimizam o número de comparações efetuadas no pior caso para uma busca com chaves de probabilidades de ocorrências idênticas.

Contudo, para garantir essa propriedade em aplicações dinâmicas, é preciso reconstruir a árvore para seu estado ideal a cada operação sobre seus nós (inclusão ou exclusão).

Conceito de balanceamento

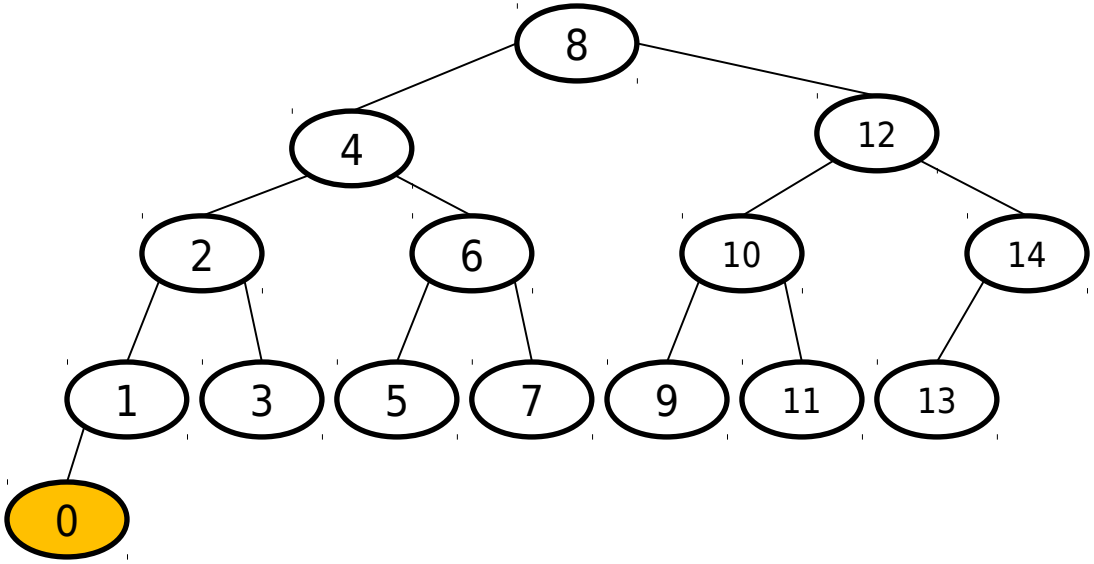
Exemplo:



Suponha a inclusão da chave 0 (zero).

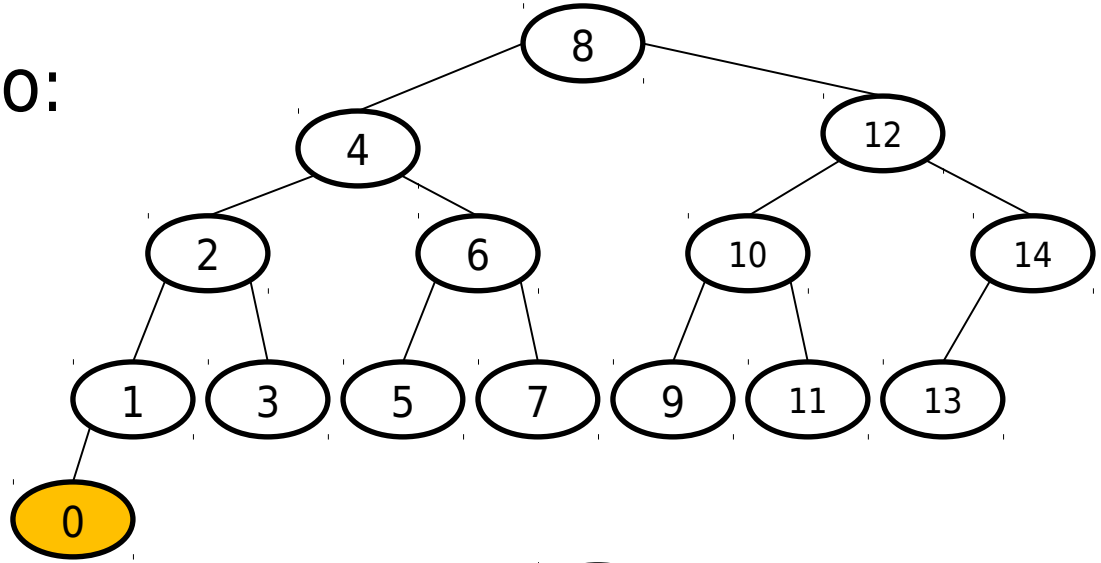
Conceito de balanceamento

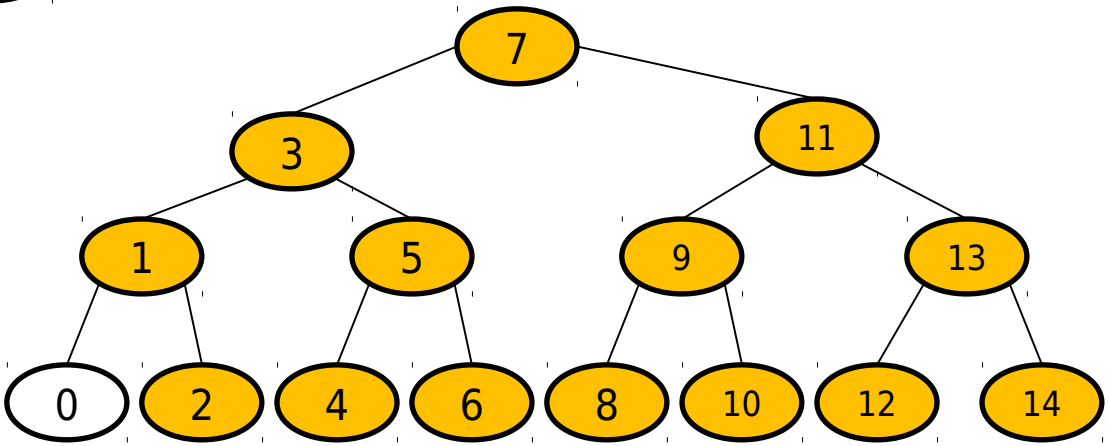
Exemplo:



Conceito de balanceamento

Exemplo:





Conceito de balanceamento

Para reorganizar a árvore anterior, foi utilizada uma abordagem O(n), no pior caso.

Naturalmente, essa é uma péssima solução, uma vez que operações como inserção e remoção geralmente são efetuados em O(logn) passos.

Por esse motivo, árvores completas não são recomendadas para aplicações que requeiram estruturas dinâmicas.

Conceito de balanceamento

Alternativa: utilizar um determinado tipo de árvore binária cujo pior caso para a busca não seja necessariamente tão pequeno quanto o mínimo 1 + lower_bound(logn) passos pela árvore completa.

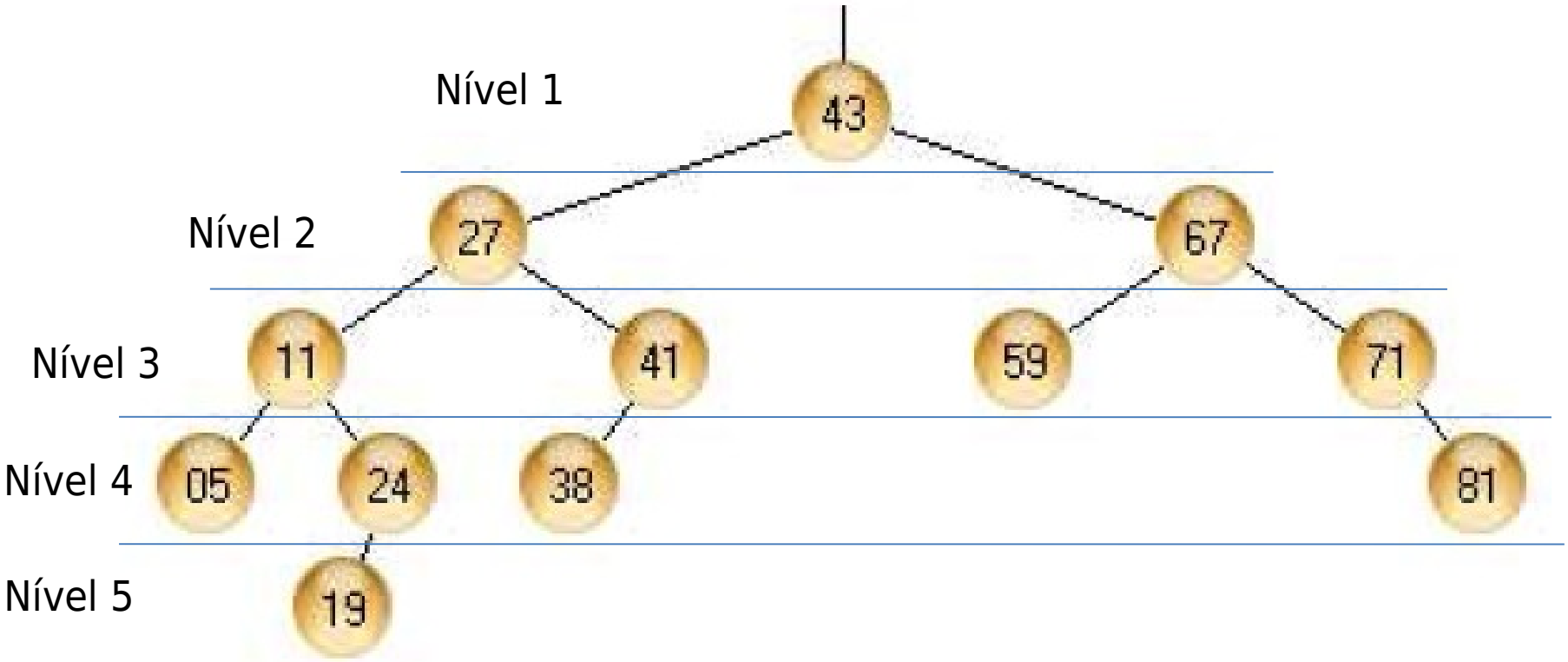
Contudo, a altura dessa árvore deve ser da mesma ordem de grandeza que a altura de uma árvore completa com o mesmo número de nós.

Ou seja, deve possuir altura O(logn) para todas as suas subárvores.

Árvore AVL

- A AVL (Adelson-Velskii e Landis – 1962) é uma árvore altamente balanceada, isto é, nas inserções e exclusões, procura-se executar uma rotina de balanceamento tal que as alturas das sub-árvores esquerda e sub-árvores direita tenham alturas **bem próximas**
- Definição
  - Uma árvore AVL é uma árvore na qual as alturas das subárvores esquerda e direita de cada nó diferem no máximo por uma unidade.
  - Fator de balanceamento
  - Altura da subárvore direita - altura da subárvore esquerda

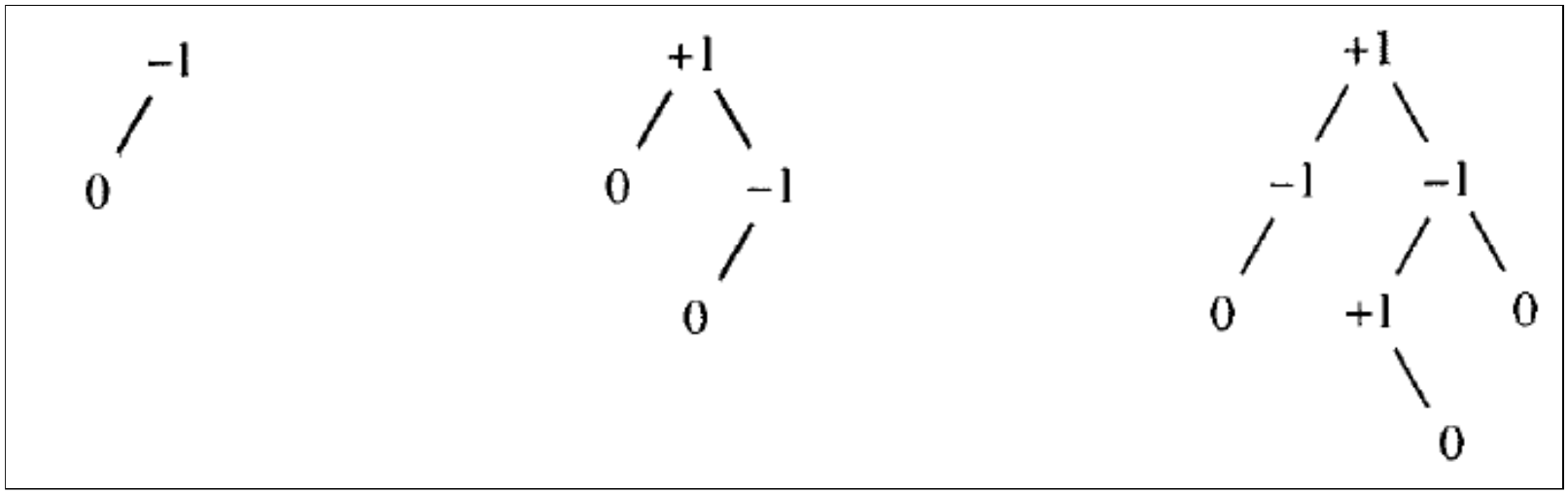
Exemplo


Nível 1
Nível 2
Nível 3
Nível 4
Nível 5
43
27
67
11
41
59
71
05
24
38
81
19

Árvore AVL

Em uma árvore AVL, para todo nó, seja hd a altura de uma subárvore direita e he a altura de uma subárvore esquerda de um nó:

`hd – he ϵ {0, 1, -1}`



`Se o fator de balanceamento de qualquer nó ficar menor do que -1 ou maior do que 1 então a árvore tem que ser balanceada.`

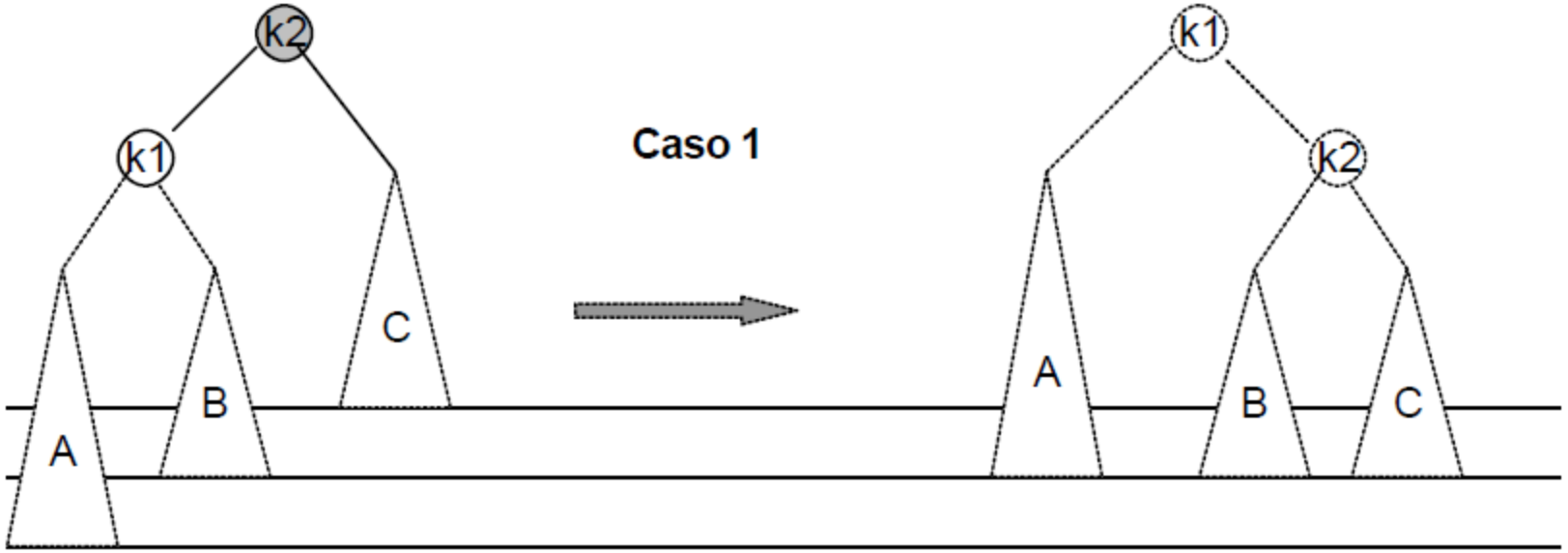
Rotações em AVL

Na inserção utiliza-se um processo de balanceamento que pode ser de 2 tipos gerais:

Rotação simples

Rotação dupla

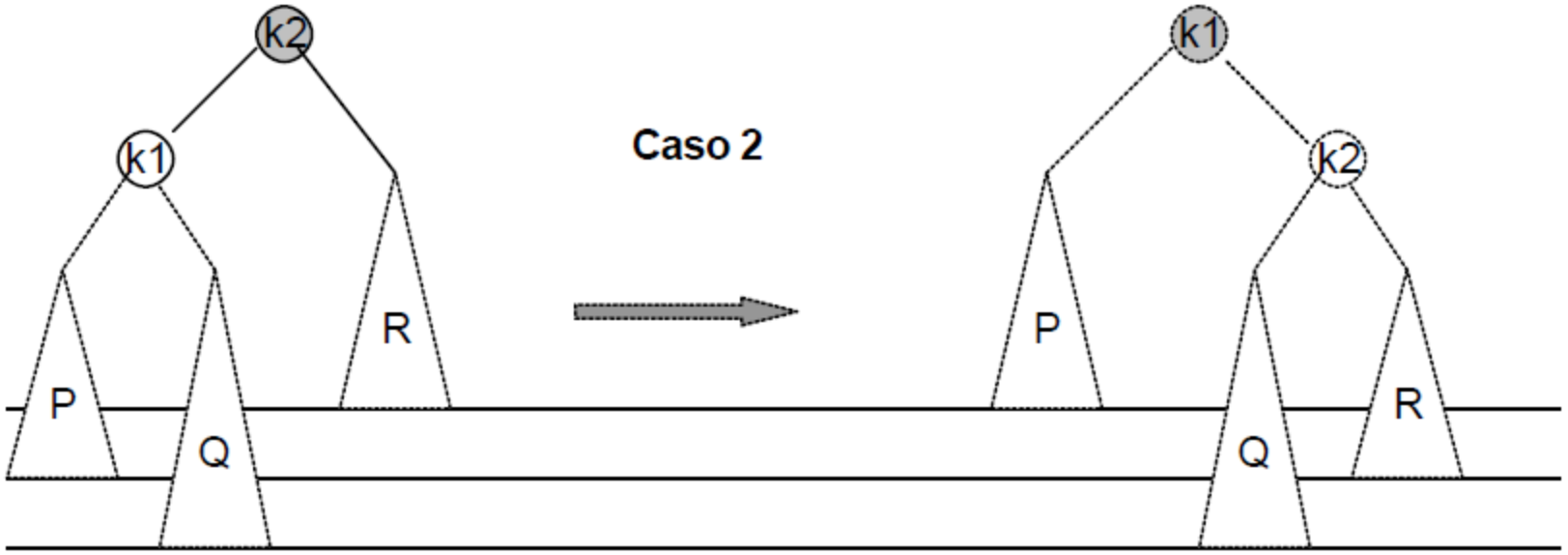
Rotações simples



k2 é nó mais profundo onde falha o equilíbrio

sub-árvore esquerda está 2 níveis abaixo da direita

Rotações dupla

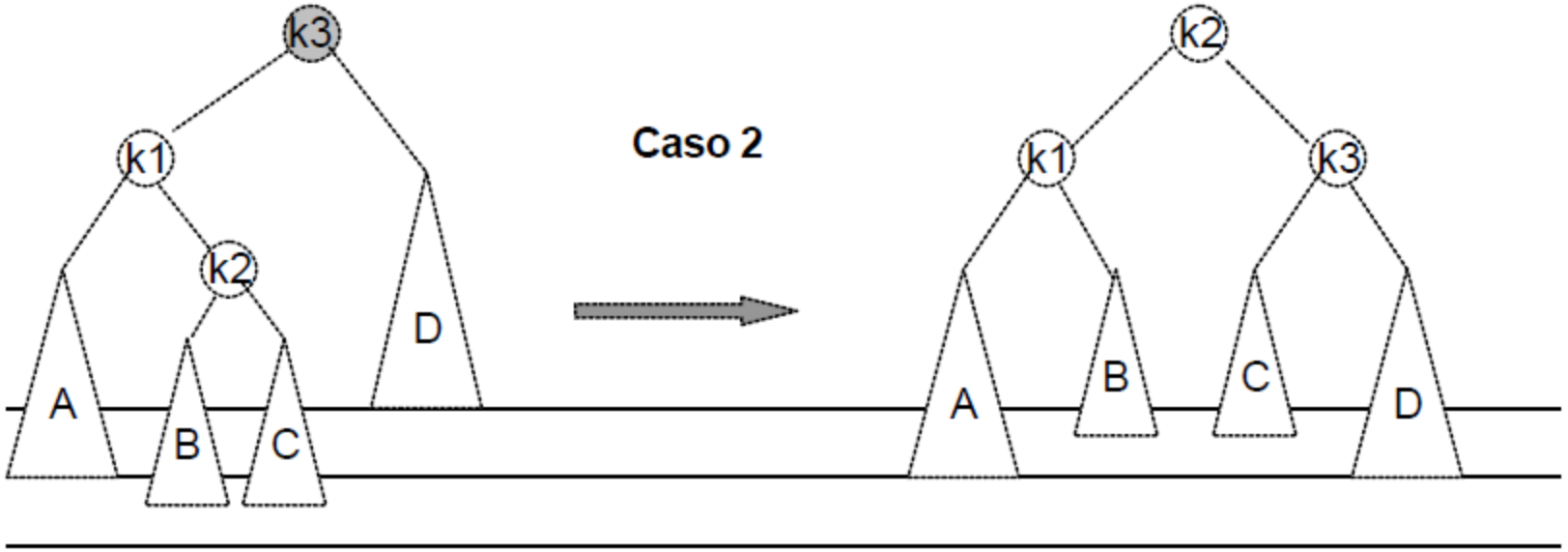


Rotação simples não resolve o desequilíbrio!

sub-árvore Q está a 2 níveis de diferença de R

sub-árvore Q passa a estar a 2 níveis de diferença de P

Rotações dupla



Uma das subárvores B ou C está 2 níveis abaixo de D

k2, a chave intermédia, fica na raiz

posições de k1, k3 e subárvores completamente determinadas pela ordenação

Rotações em AVL

Na inserção utiliza-se um processo de balanceamento que pode ser de 4 tipos específicos:

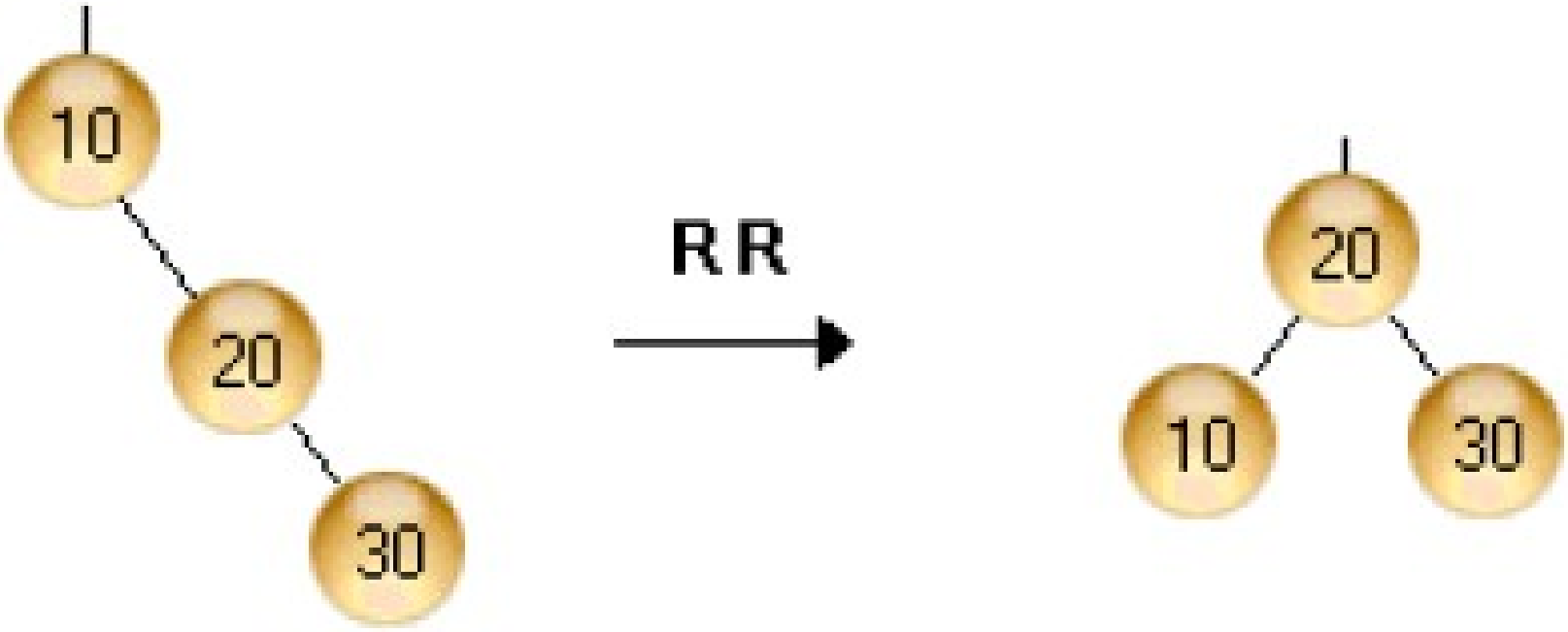
RR → caso Right-Right (rotação a esquerda)

LL → caso Left-Left (rotação a direita)

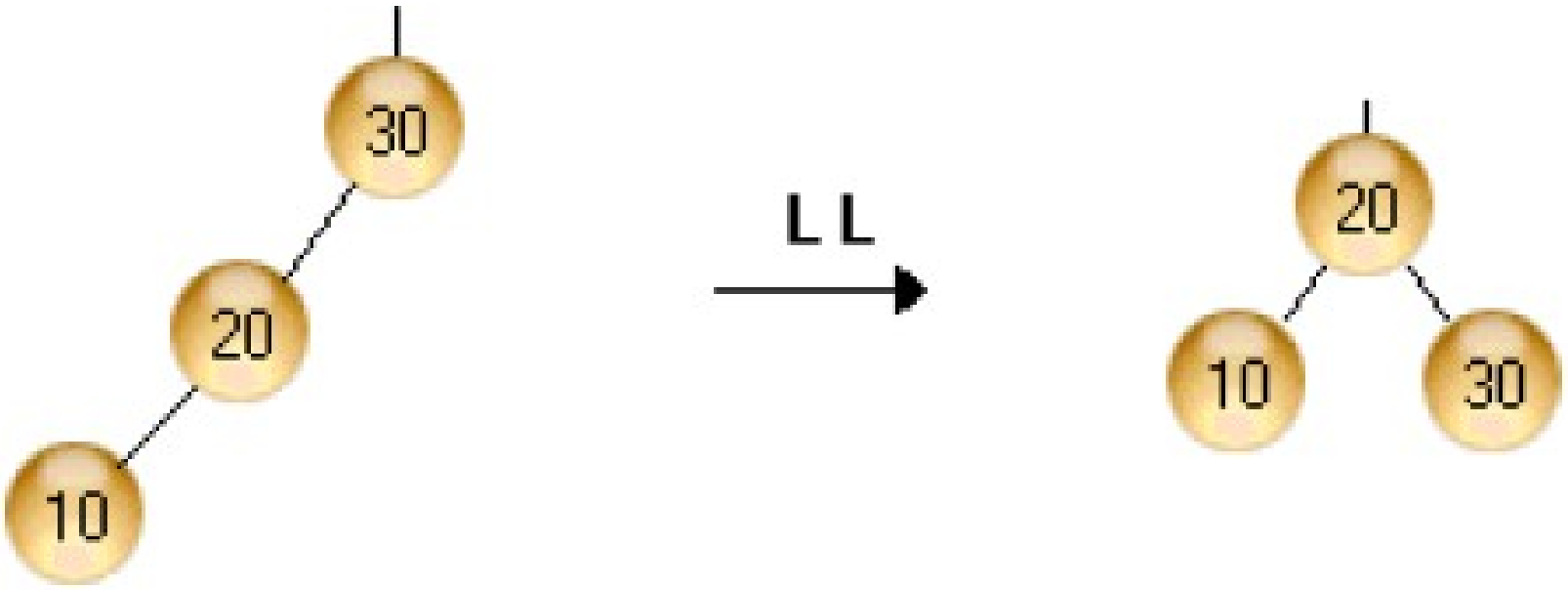
LR → caso Left-Right (rotação esquerda-direita)

RL → caso Right-Left (rotação direita-esquerda)

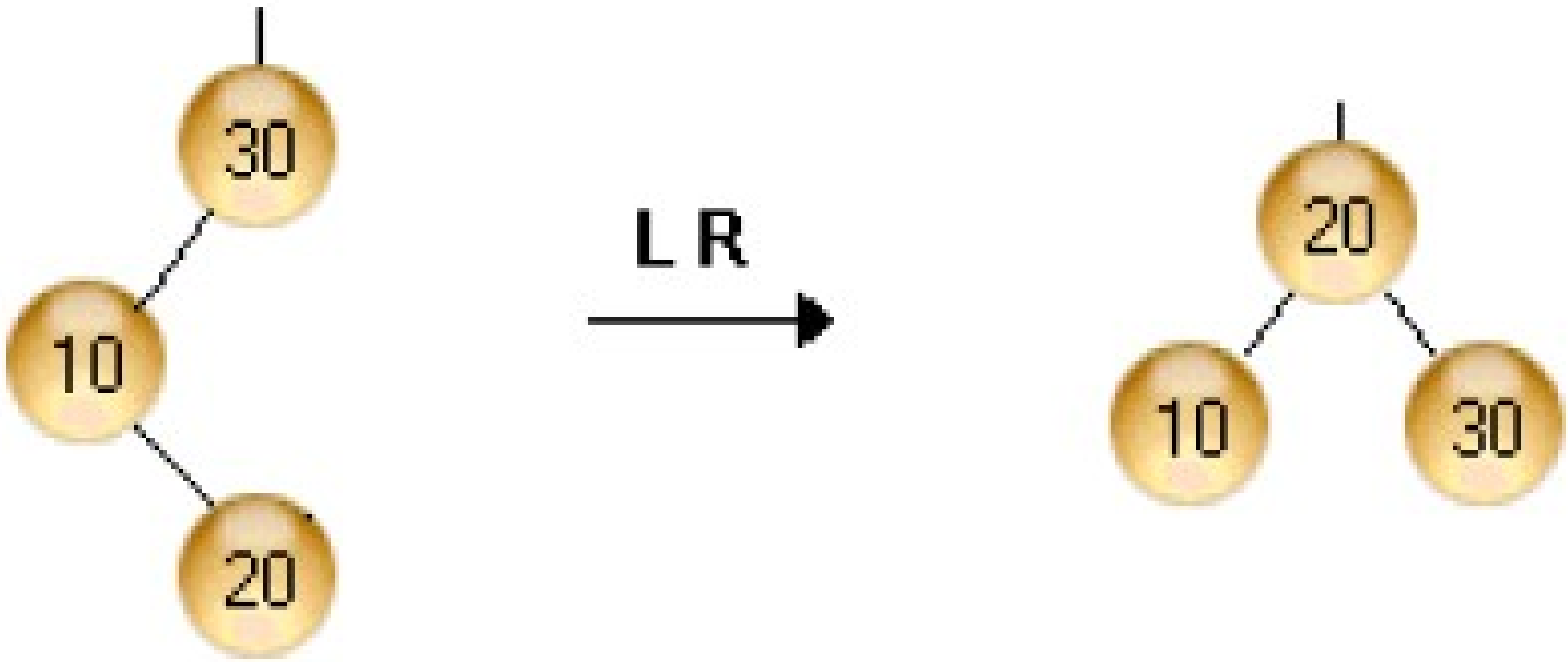
Caso Right-Right
(rotação a esquerda)



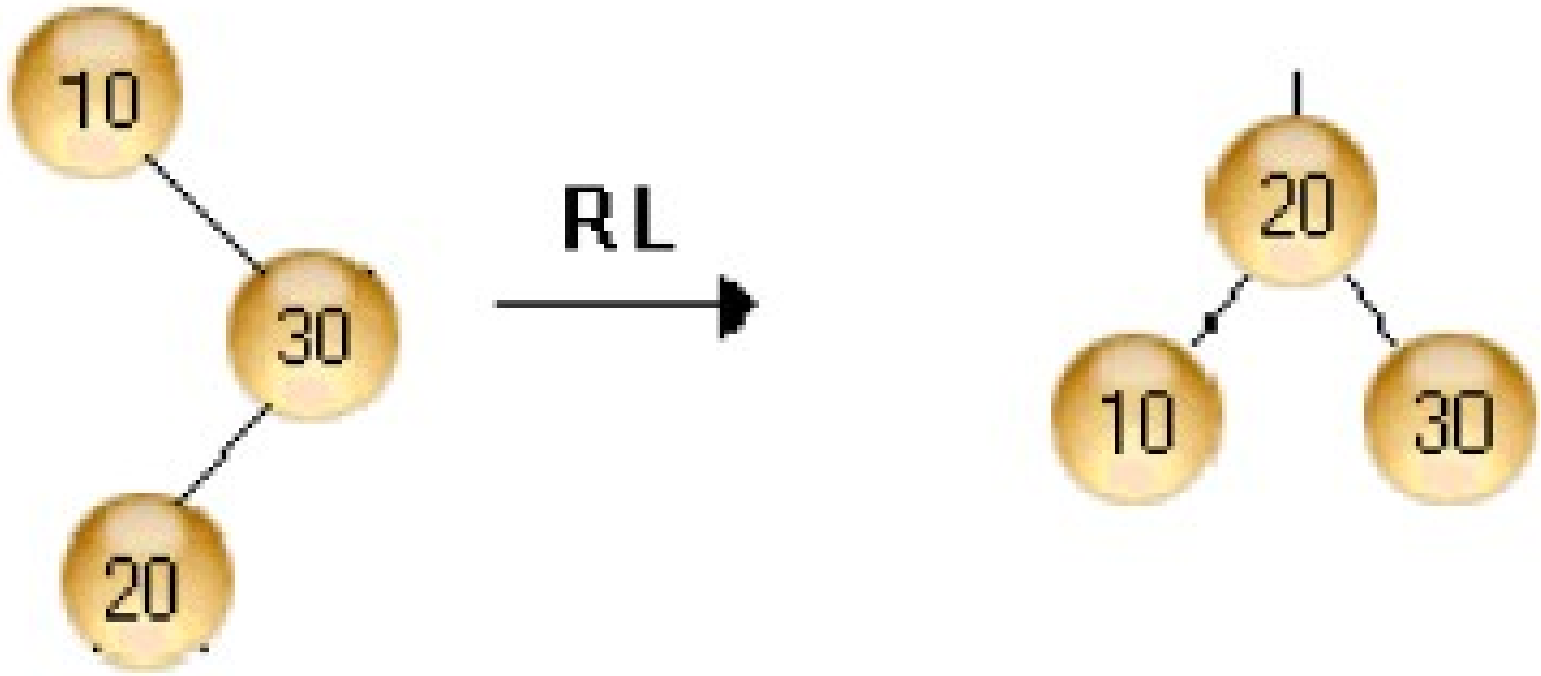
Caso Left-Left
(rotação a direita)



Left-Right
(rotação esquerda-direita)



Caso Right-Left
(rotação direita-esquerda)



Fator de Balanceamento

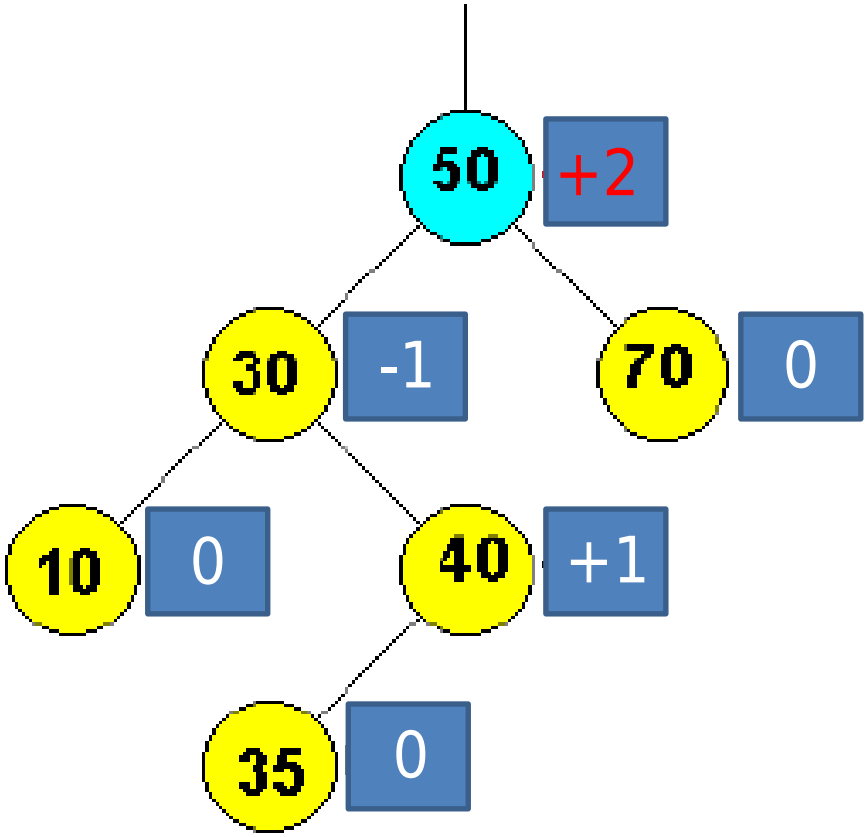
Coeficiente que serve como referência para verificar se uma árvore AVL está ou não balanceada

O fator é calculado nó a nó e leva em consideração a diferença das alturas das sub-árvores da direita e da esquerda

Genericamente

FB = he - hd

Exemplo – FB de cada nó

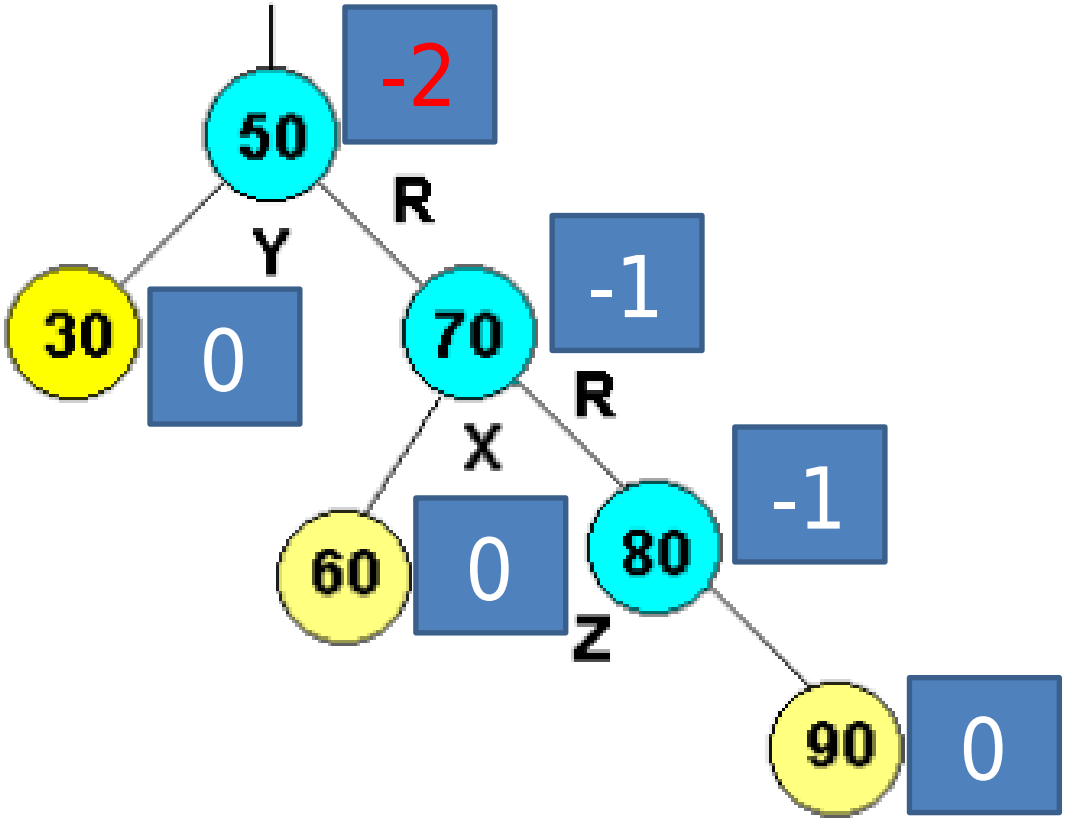


Quando balancear?

- Sempre que existir um fator de balanceamento superior a +1 ou inferior a -1
- Caso exista mais de um nó que se encaixe neste perfil deve-se sempre balancear o nó com o nível mais alto
- Como balancear? Utilizando os processos:
  - Right-Right
  - Left-Left
  - Left-Right
  - Right-Left

Tipos de Balanceamento - RR

Suponha na figura que a última célula a ser inserida foi a célula de chave 90



Tipos de Balanceamento - RR

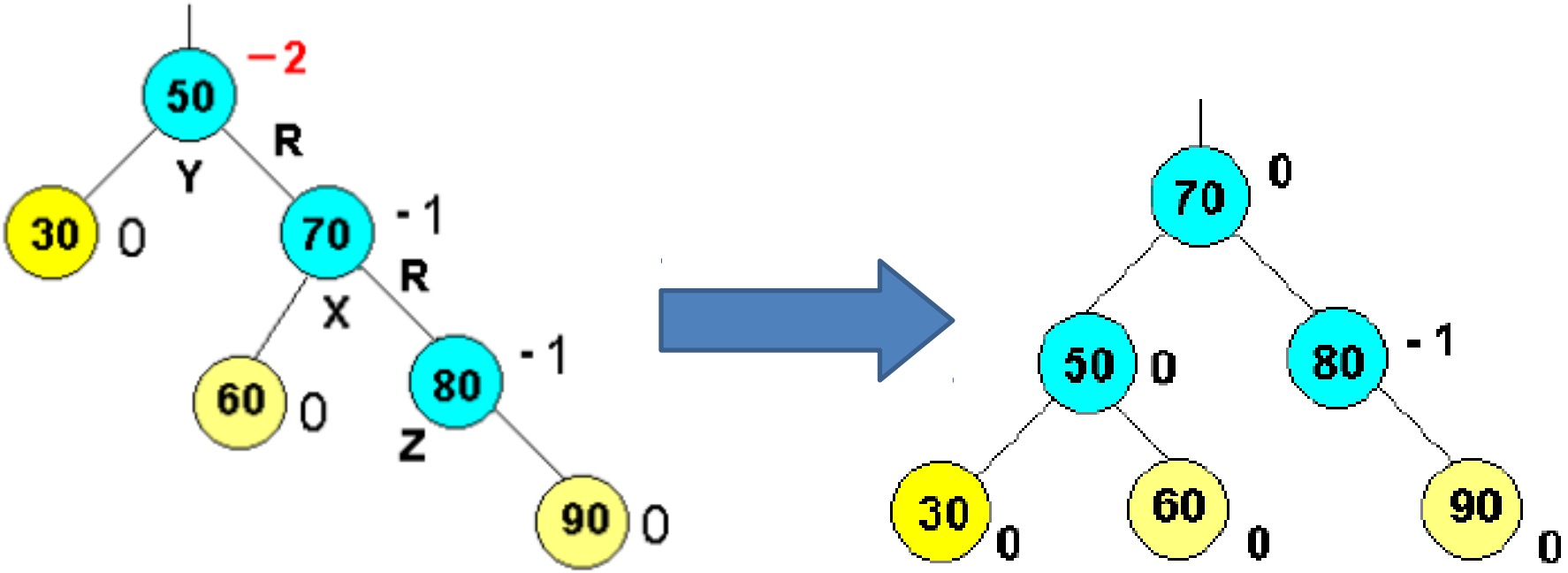
O nó X que está no nível do meio dos três envolvidos toma o lugar do nó com FB=-2

A sub-árvore direita do nó X permanece

A sub-árvore esquerda do nó X será colocada como sub-árvore direita do nó Y

O filho esquerdo do nó X aponta para o nó Y

Tipos de Balanceamento - RR

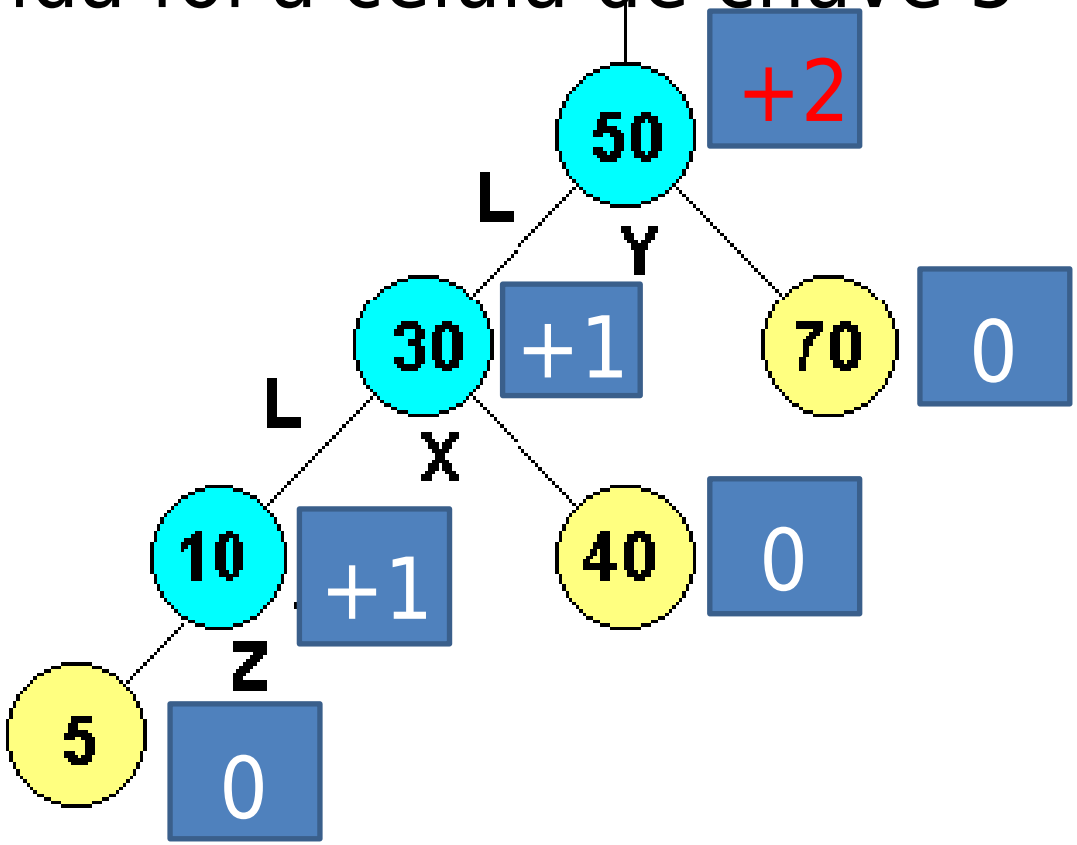


Tipos de Balanceamento - RR

```
static Node rotacaoRR(Node
y)
    {
        Node x = y.right;
        y.right = x.left;
        x.left = y;
        return x;
    }
```

Tipos de Balanceamento - LL

Suponha na figura que a última célula a ser inserida foi a célula de chave 5



Tipos de Balanceamento - LL

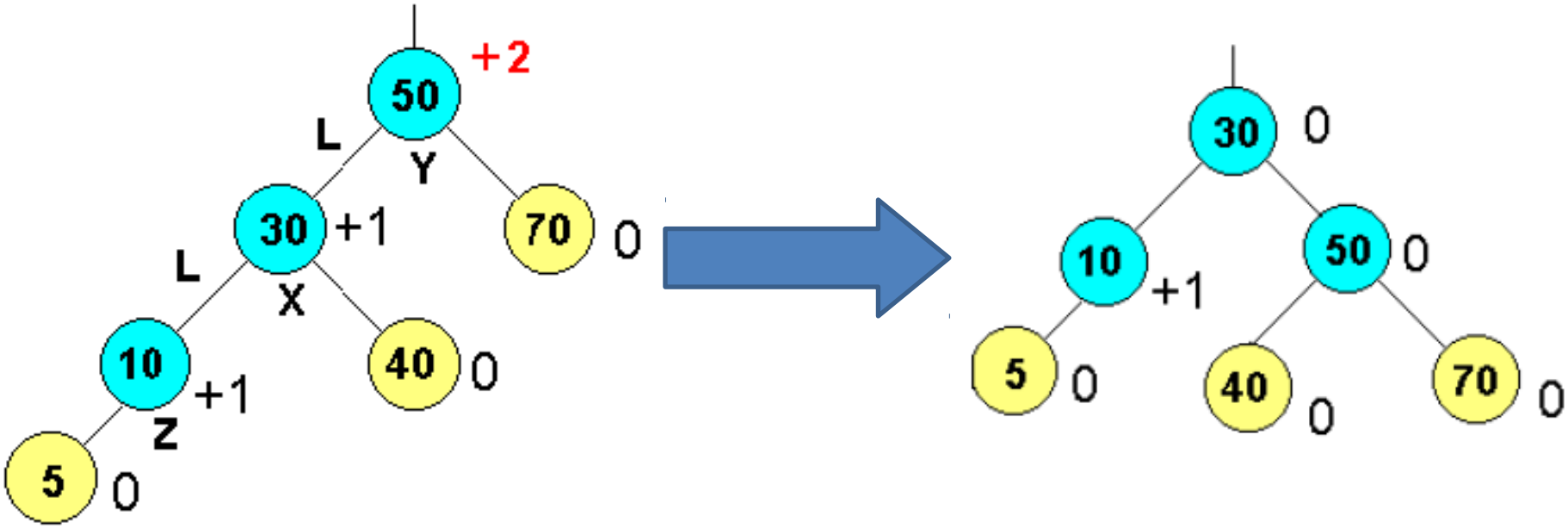
O nó X que está no nível do meio dos três envolvidos toma o lugar do nó com FB=-2

A sub-árvore esquerda do nó X permanece

A sub-árvore direita do nó X será colocada como sub-árvore esquerda do nó Y

O filho direito do nó X aponta para o nó Y

Tipos de Balanceamento - LL

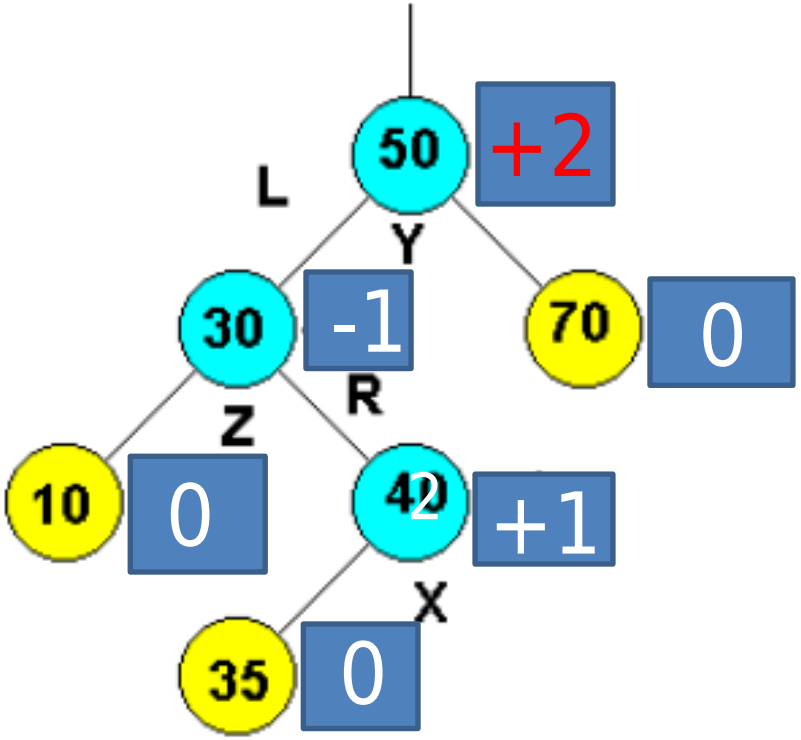


Tipos de Balanceamento - LL

```
static Node rotacaoLL(Node
y)
    {
        Node x = y.left;
        y.left = x.right;
        x.right = y;
        return x;
    }
```

Tipos de Balanceamento - LR

Suponha na figura que a última célula a ser inserida foi a célula de chave 35



Tipos de Balanceamento - LR

O nó que está no nível mais alto das três envolvidas (nó X) toma o lugar da célula cujo fator de balanceamento é +2 (nó Y)

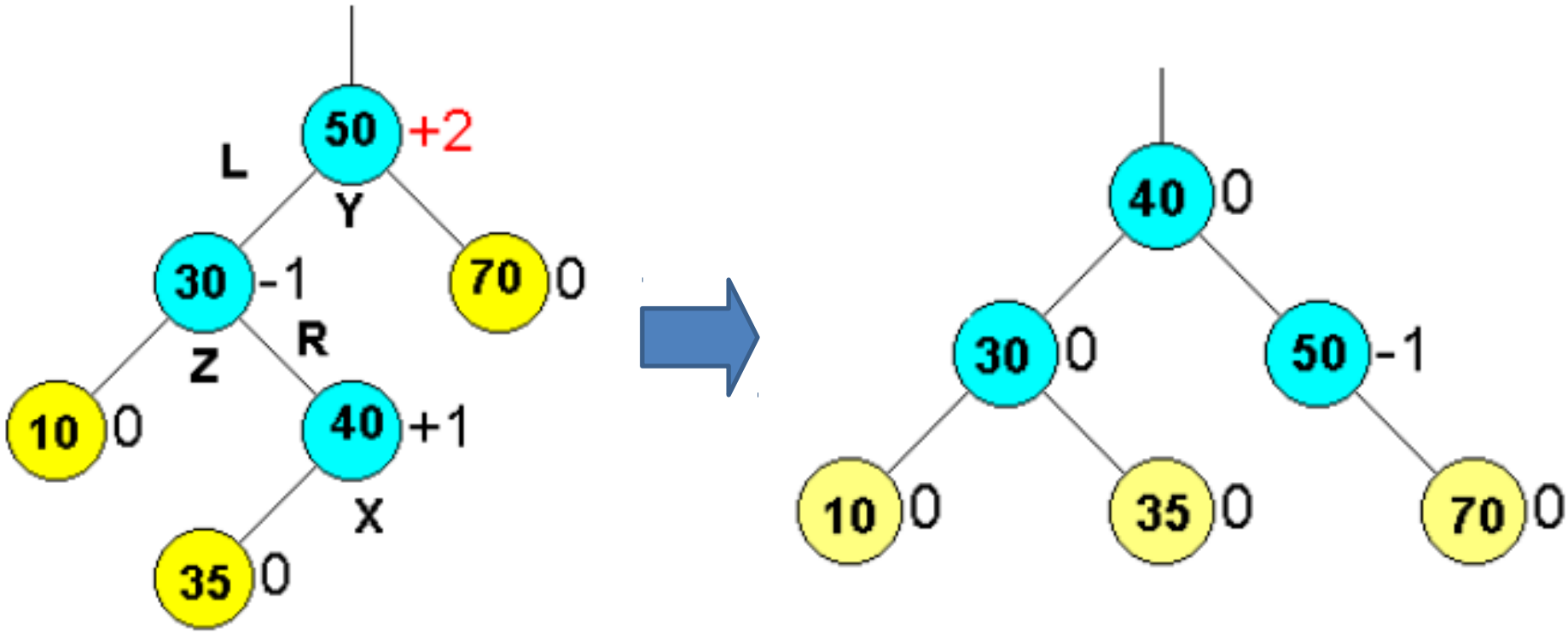
A sub-árvore direita do nó X será colocada como sub-árvore esquerda do nó Y

A sub-árvore esquerda do nó X será colocada como sub-árvore direita do nó Z

O filho direito do nó X aponta para o nó Y

O filho esquerdo do nó X aponta para o nó Z

Tipos de Balanceamento - LR

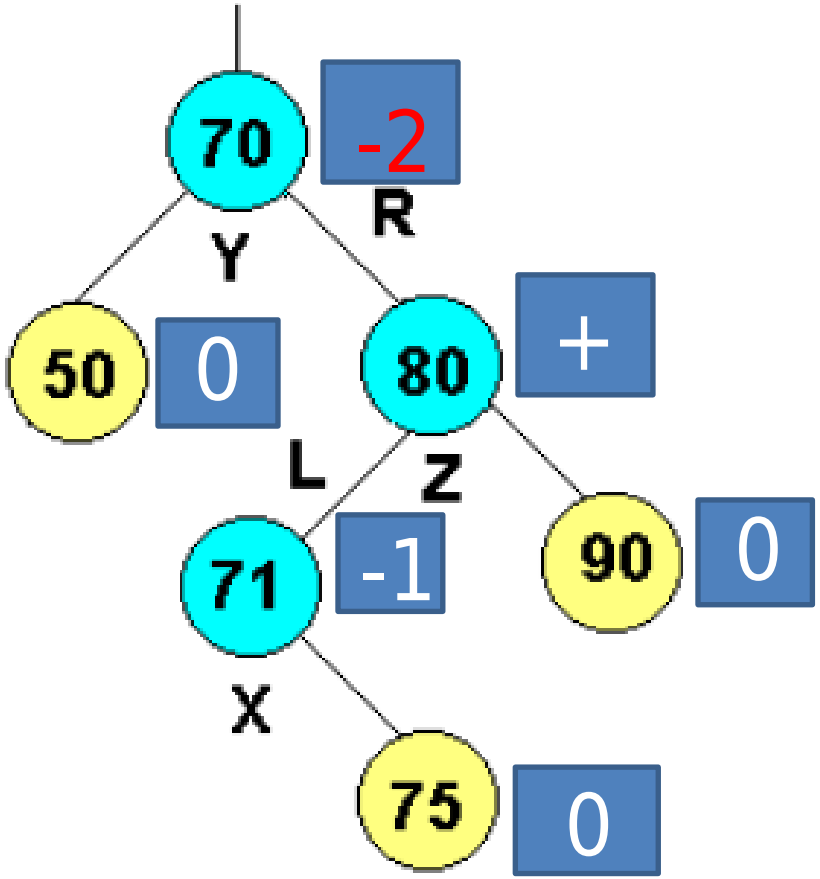


Tipos de Balanceamento - LR

```
static Node rotacaoLR(Node y)
    {
        y.left = rotacaoRR(y.left);
        return rotacaoLL(y);
    }
```

Tipos de Balanceamento - RL

Suponha na figura que a última célula a ser inserida foi a célula de chave 75



Tipos de Balanceamento - RL

O nó que está no nível mais alto das três envolvidas (nó X) toma o lugar da célula cujo fator de balanceamento é -2 (nó Y)

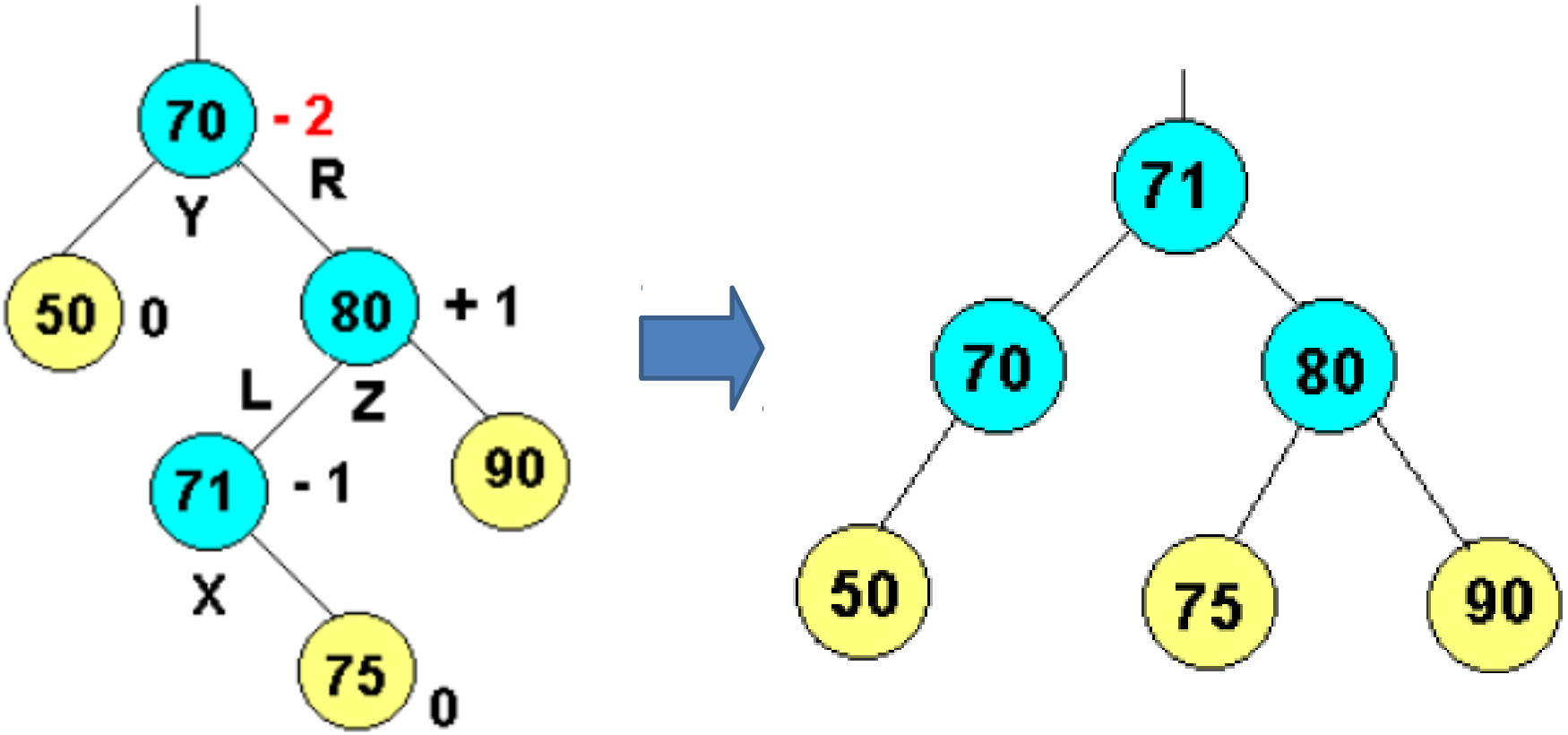
A sub-árvore direita do nó X será colocada como sub-árvore esquerda do nó Z

A sub-árvore esquerda do nó X será colocada como sub-árvore direita do nó Y

O filho direito do nó X aponta para o nó Y

O filho esquerdo do nó X aponta para o nó Z

Tipos de Balanceamento - RL



Tipos de Balanceamento - RL

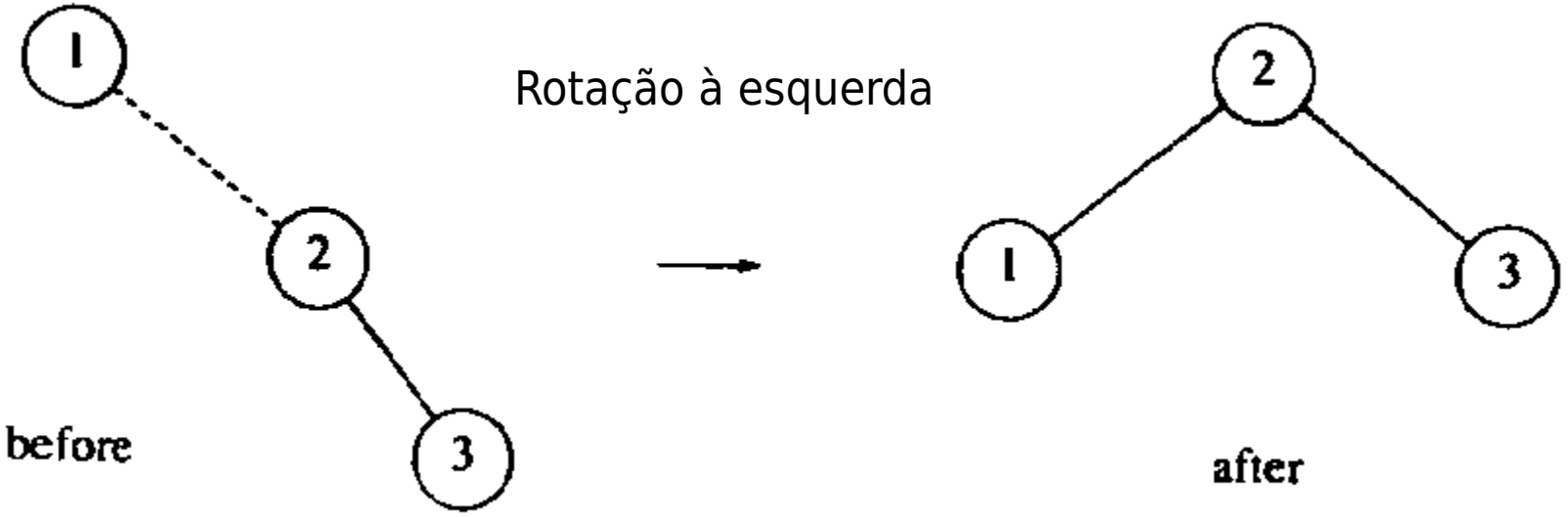
```
static Node rotacaoRL(Node y)
    {
        y.right = rotacaoLL(y.right);
        return rotacaoRR(y);
    }
```

Árvore AVL

Mais exemplos...

Inserção de 1, 2 e 3.

Ao inserir 3, o nó raiz fica desbalanceado (+2)

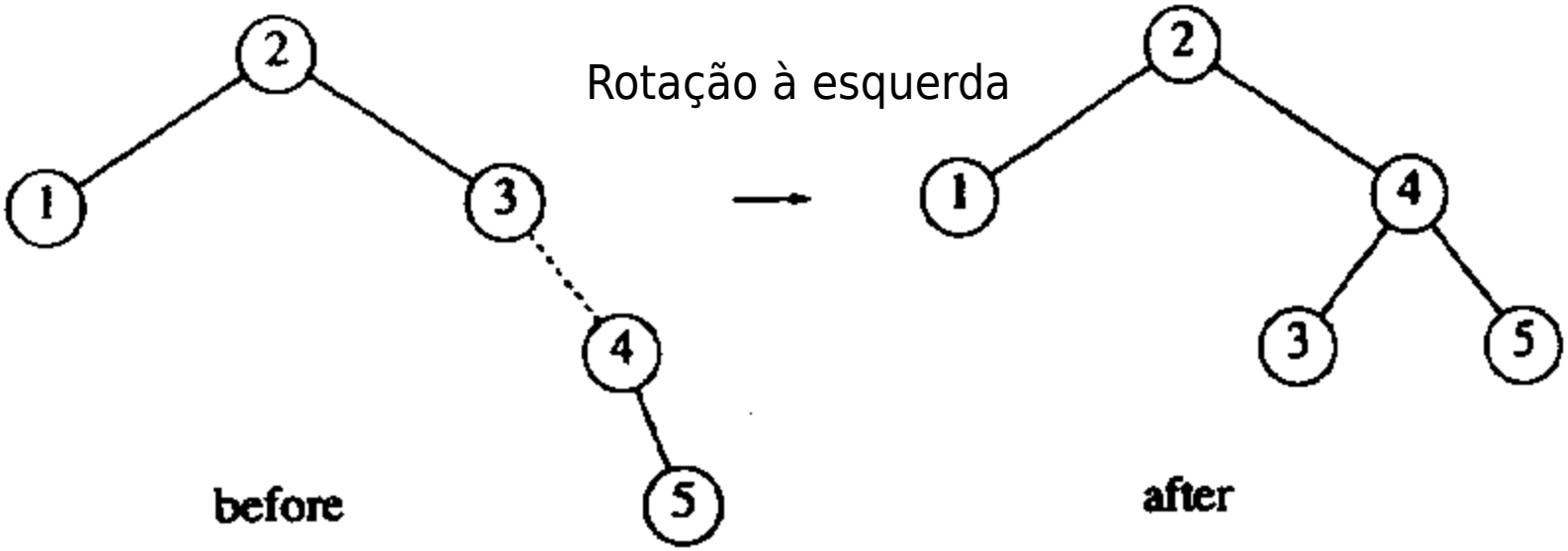


Árvore AVL

Inserção do 4 e 5.

4: sem problemas

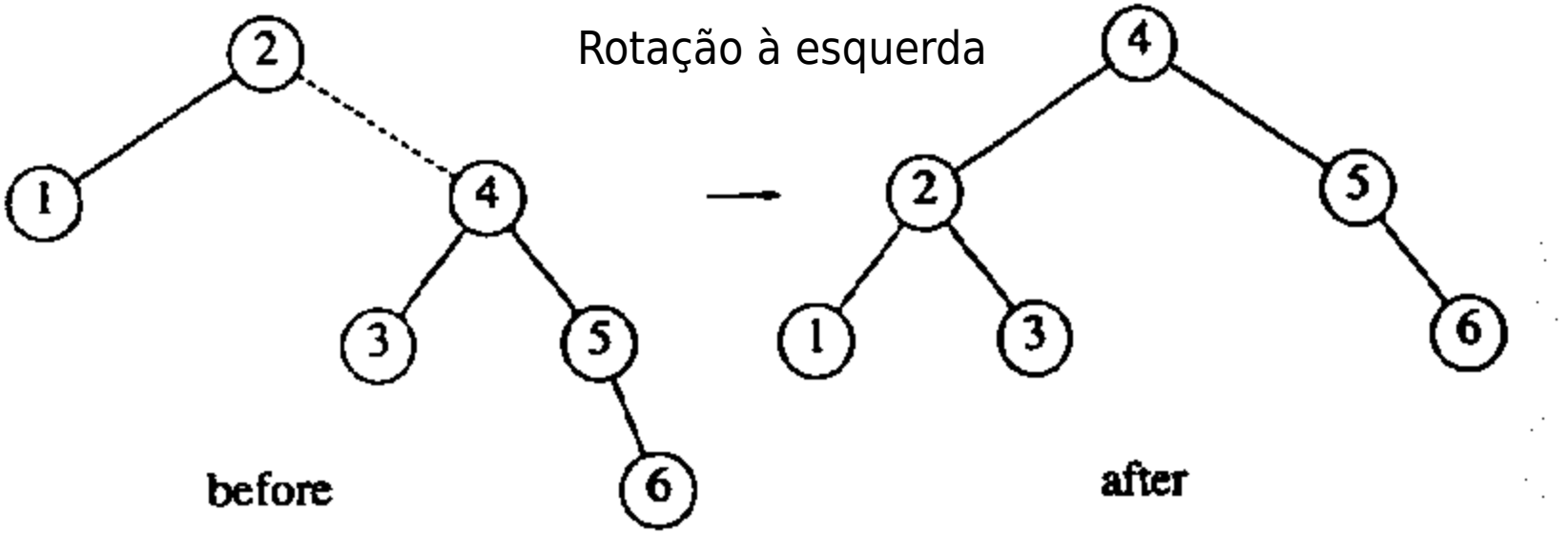
5: desbalanceamento do nó 3 (+2)



Árvore AVL

Inserção do 6

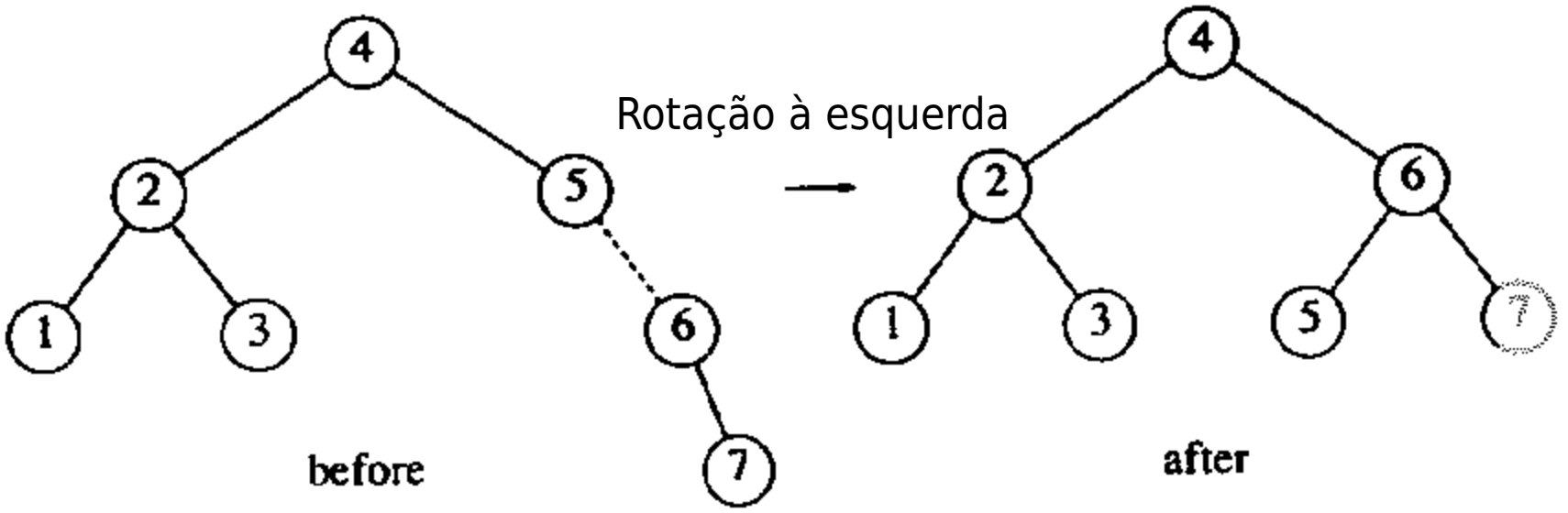
Nó 2 fica desbalanceado (+2)



Árvore AVL

Inserção do nó 7

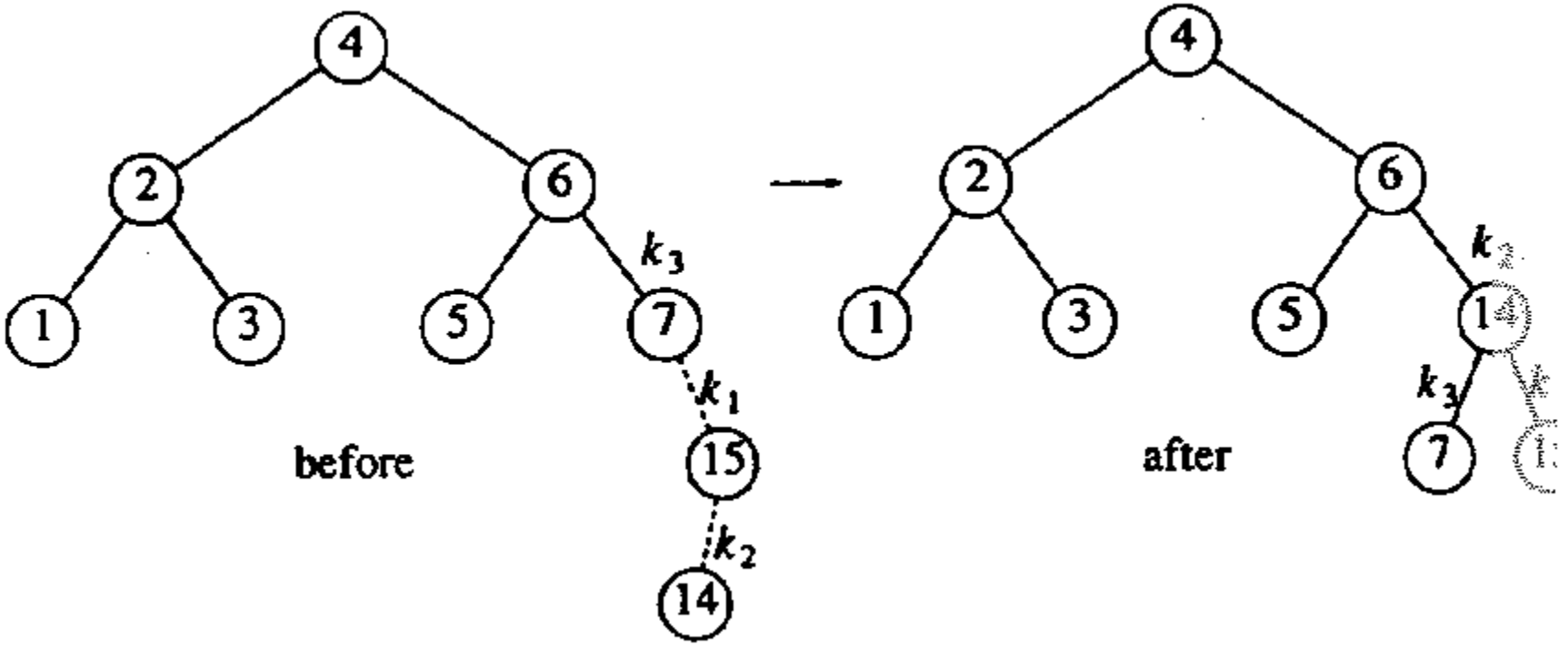
Nó 5 fica desbalanceado (+2)



Árvore AVL

Inserção de 15 e 14

Rotação dupla: 14 e 15 à direita e depois 7 e 14 à esquerda.

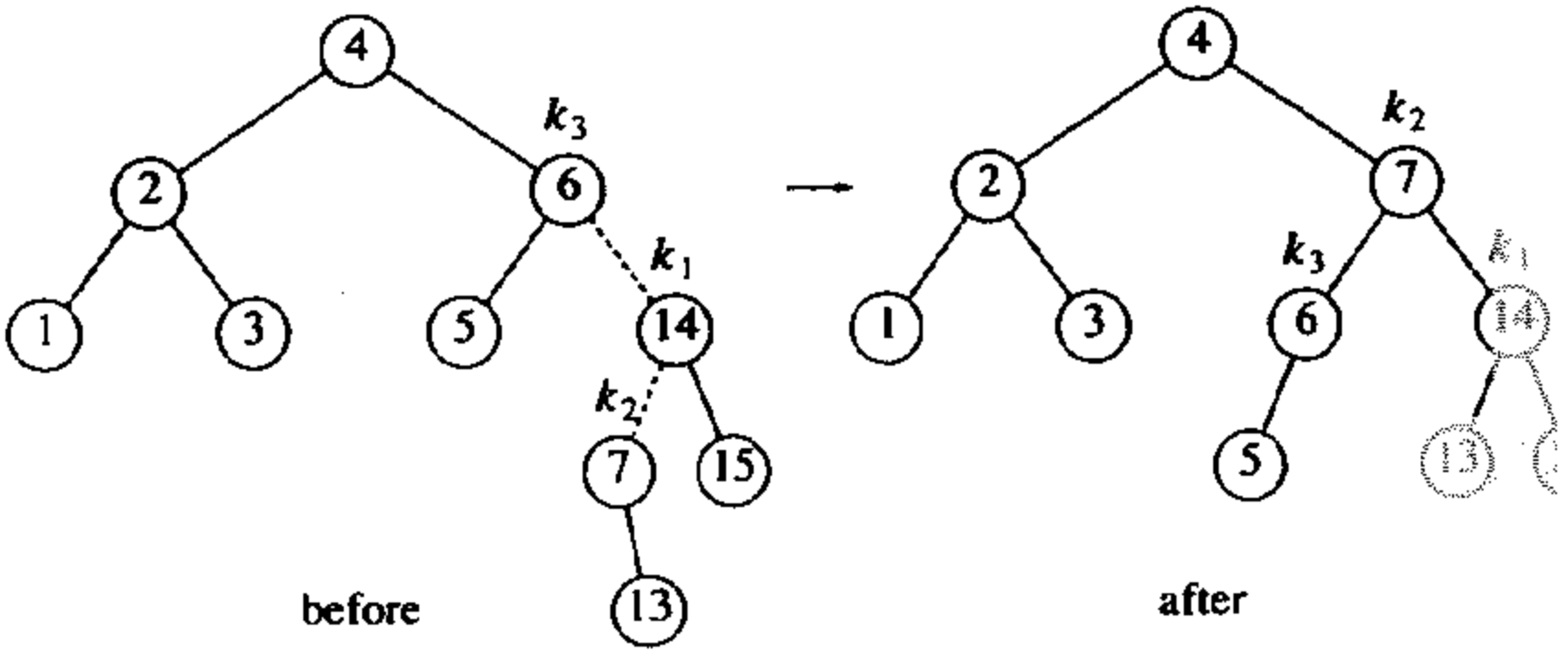


Árvore AVL

Inserção do 13

Rotação do 7 e 14 à direita

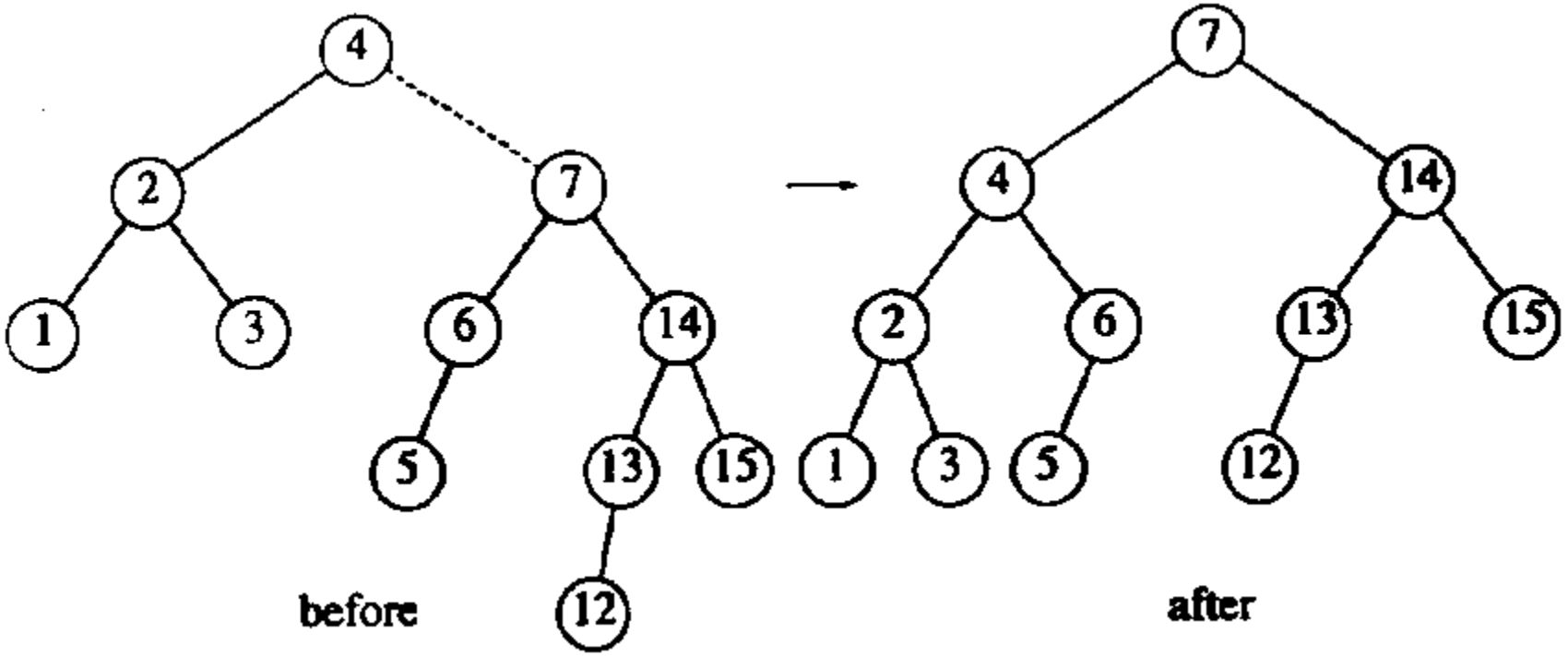
Rotação de 6 e 7 à esquerda



Árvore AVL

Inserção do 12

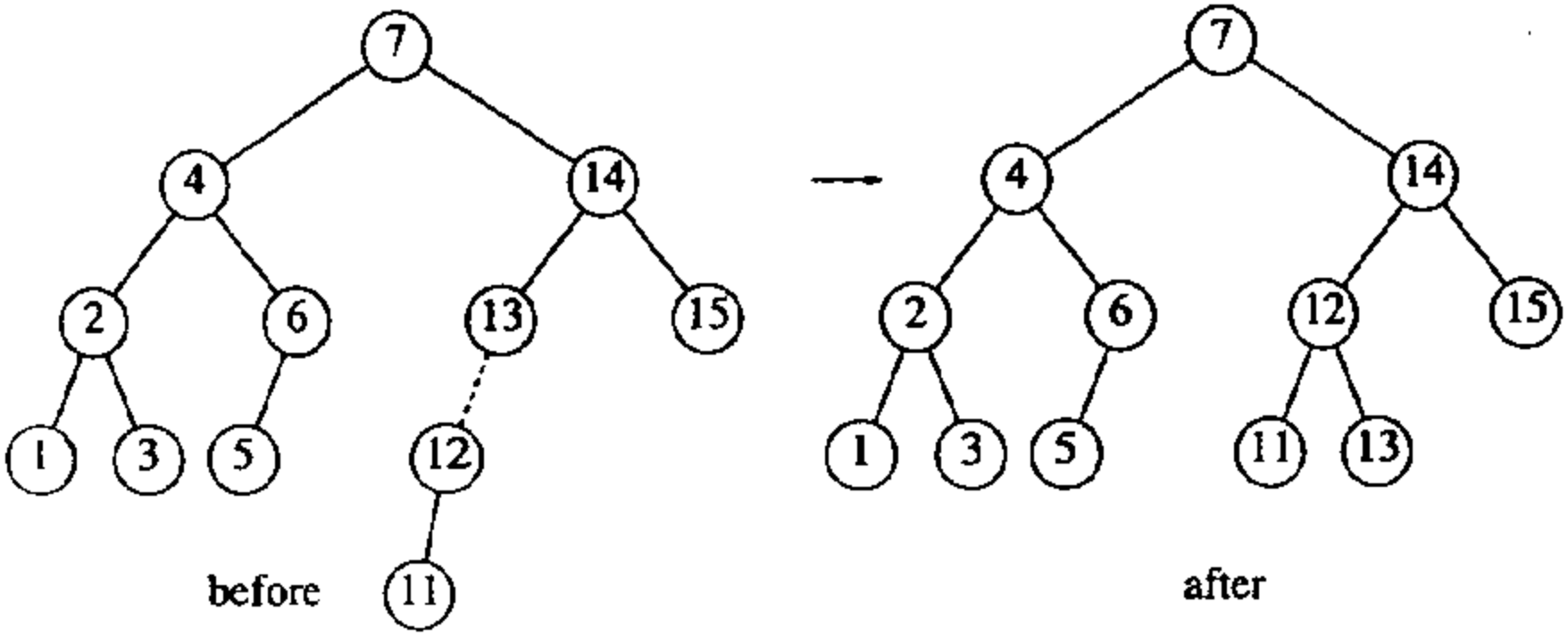
Rotação da raiz à esquerda



Árvore AVL

Inserção do 11

Rotação de 12 e 13 à direita



Algoritmo

- Algoritmo recursivo
  - Inserir nó com chave X numa árvore A
    - recursivamente, inserir na subárvore conveniente de A, SA
    - se a altura de SA não se modifica: terminar
    - se a altura de SA é modificada: se ocorre desequilíbrio em A, fazer as rotações necessárias para reequilibrar
  - Comparação de alturas
    - para evitar o cálculo repetido de alturas de sub-árvores, pode-se manter em cada nó o resultado da comparação das alturas das sub-árvores
- Algoritmo iterativo
  - Especificar parada logo que uma rotação é realizada
- Na prática, são usadas outras árvores binárias equilibradas (como as árvores vermelho-preto) em que a inserção ou remoção e a correspondente reposição do equilíbrio pode ser feito mais eficientemente

Importante

- Na inserção, caso a árvore AVL esteja desbalanceada, basta 1 operação de rotação para rebalanceá-la.
- Na remoção, caso a árvore esteja desbalanceada, pode ser necessário até log(n) operações de rotação.

Como descobrir qual rotação deve ser realizada?

- Processo simples (supondo que cada nó guarda seu fator de balanceamento):

```
void AVLTree<tipo>::balanceia(Celula *&x) {
        if (x->b == 2) {
           if (x->esq->b == -1)
               rotateLeft(x->esq);
           rotateRight(x);
        }
        else if (x->b == -2) {
             if (x->dir->b == 1)
                 rotateRight(x->dir);
             rotateLeft(x);
        }
    }
```

Exercícios

Construir uma AVL com as chaves:

(10, 20, 30, 5, 3, 50, 40, 70, 60, 90)

Construir uma AVL com as chaves:

(PSC, INF, ENG, QUI, MAT, LET, MED, ECO, ADM)