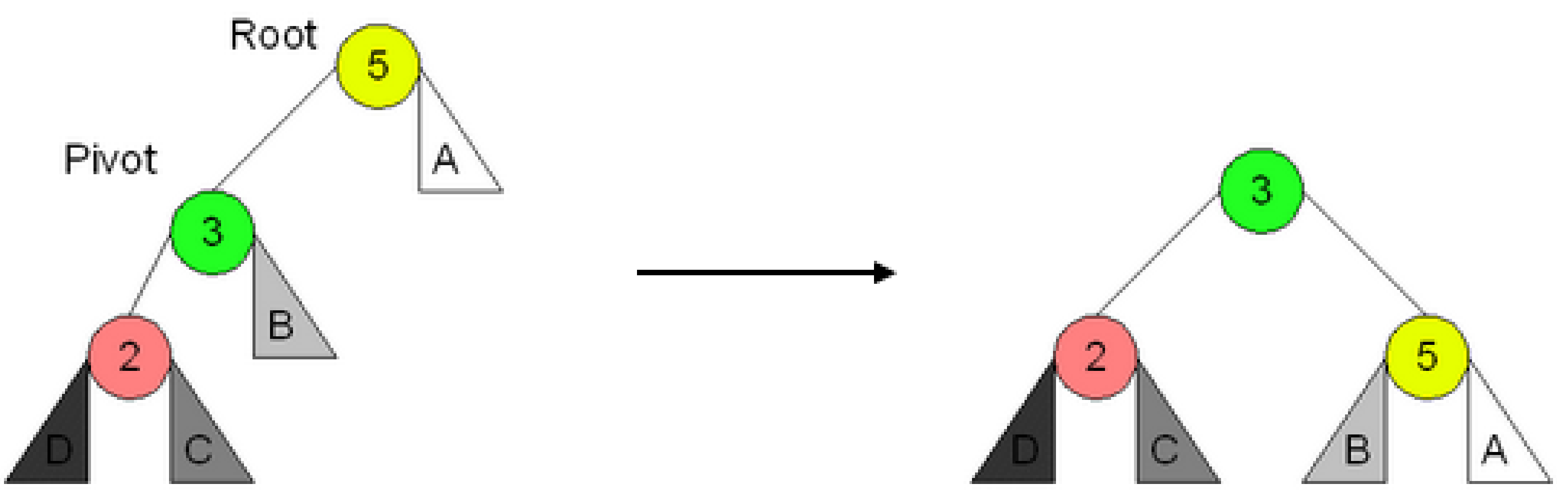
Árvore Balanceada e Árvore AVL

Centro Federal de Educação Tecnológica Celso Suckow da Fonseca
CEFET-RJ

Introdução

Para dispor de um melhor desempenho de uma estrutura de dados árvore é necessário que os elementos estejam distribuídos de forma homogênea pelas subárvores.

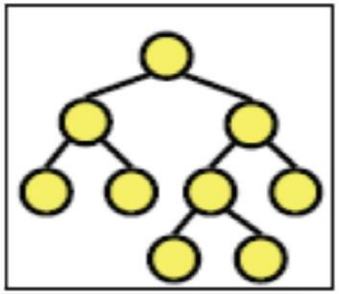


Introdução

- **Mesmo após a inserção e remoção de vários elementos, o custo de acesso deve ser mantido na mesma ordem de grandeza de uma árvore ótima ⇨ O (log n).**
  - **A estrutura deve ser alterada, periodicamente, para acomodar os elementos.**
  - **O custo dessas alterações precisa se manter em O (log n).**
- **Uma estrutura que possui essas características é denominada *balanceada*.**

Balanceamento

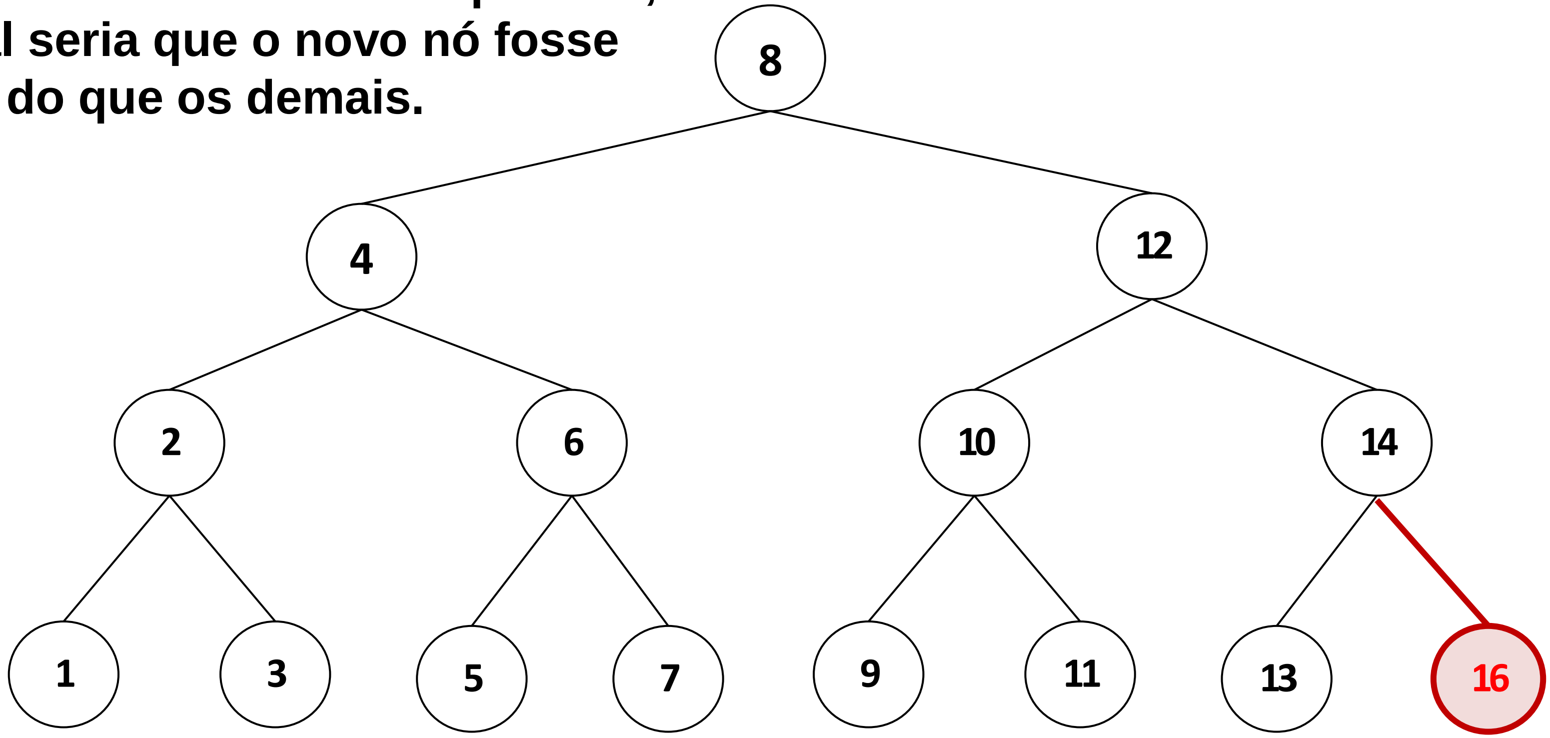
- **Uma ideia é manter a árvore completa (árvore estritamente binária onde todos os nós folhas encontram-se ou no último ou no penúltimo nível da árvore).**



- **Problema:**
  - **Após algumas inserções ou remoções a árvore pode assumir uma forma pouco recomendável para o problema de busca.**
- **Solução:**
  - **Aplicar algum algoritmo que mantenha a árvore completa, se após uma inclusão ou remoção a estrutura perder esta característica.**

Balanceamento

Para incluir um novo elemento nessa árvore e manter o equilíbrio, o ideal seria que o novo nó fosse maior do que os demais.



Balanceamento

Mas e se o elemento fosse menor do que todos os outros?

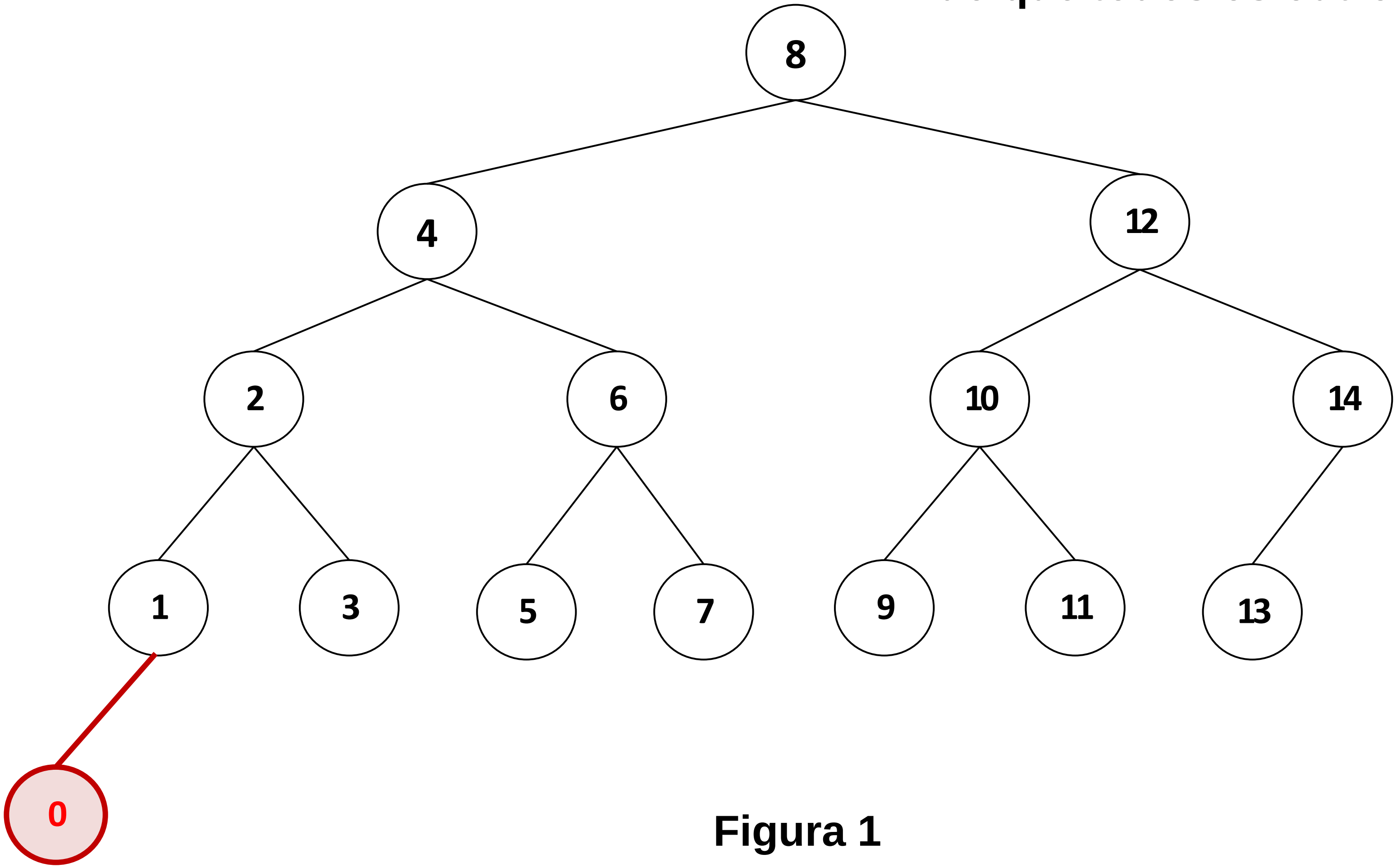


Figura 1

Balanceamento

Após reorganizar os nós na árvore:

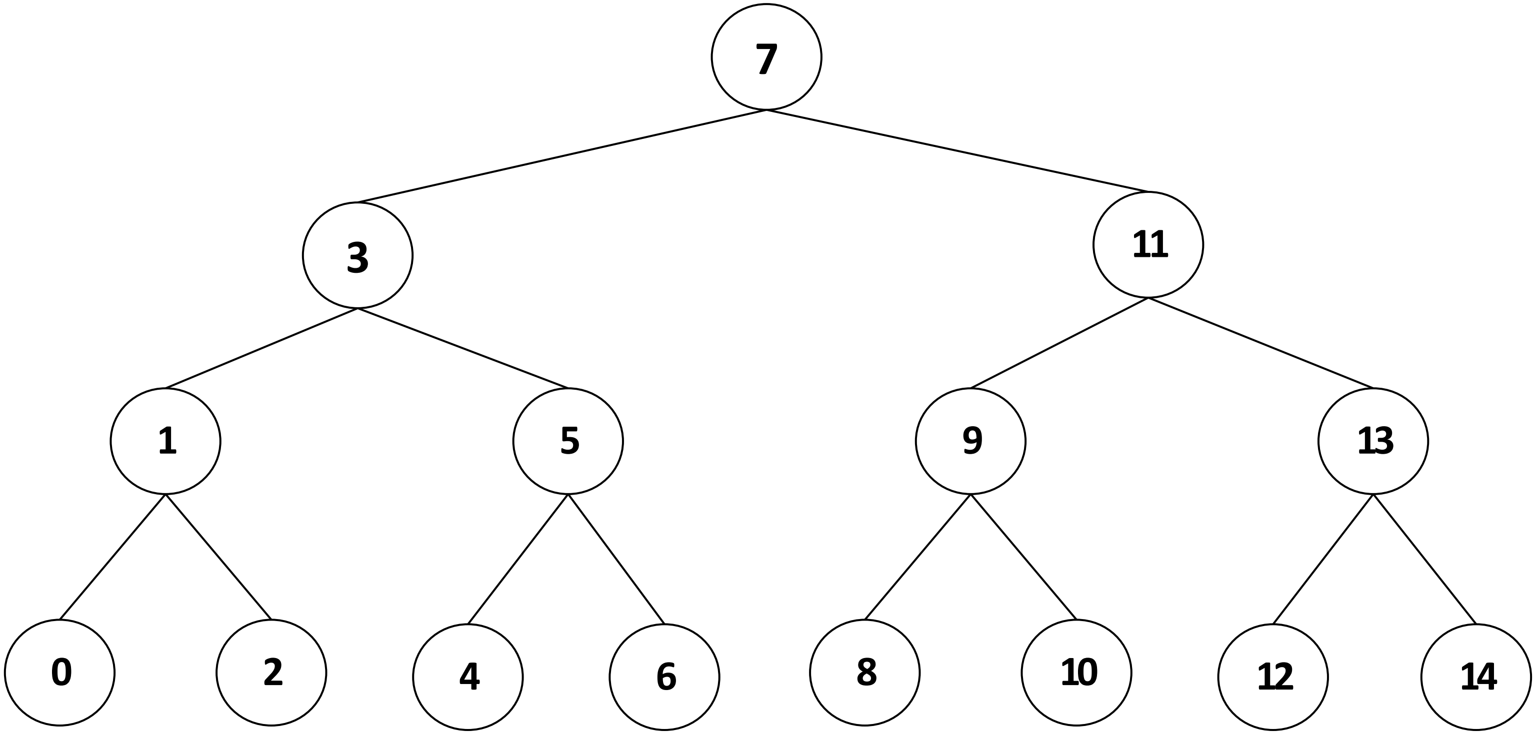


Figura 2

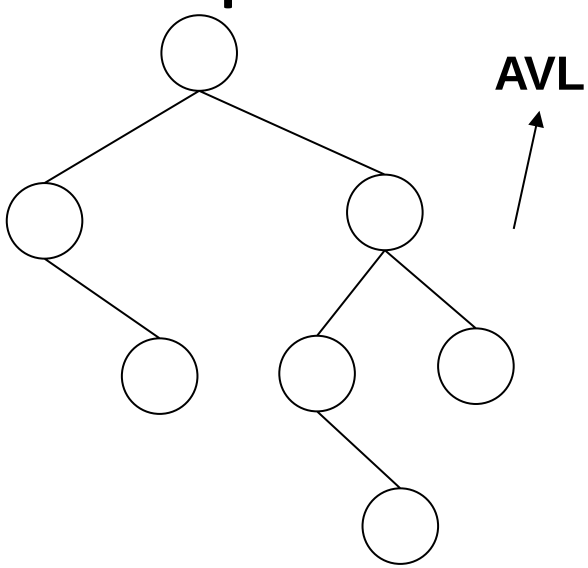
Balanceamento

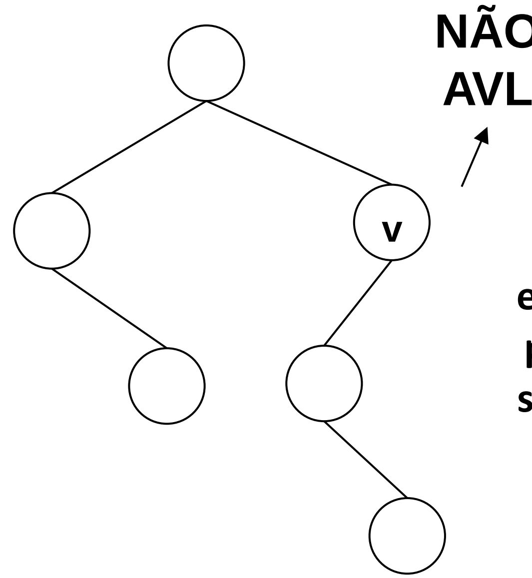
- **Para efetuar a transformação da Figura 1 para a Figura 2 após a inserção do nó 0 e manter a árvore completa foi necessário percorrer todos os nós da árvore ⇨ O(n).**
  - **Custo excessivo se comparado que as operações de inserção e remoção numa árvore seriam efetuadas em O (log n) passos.**

As árvores completas não são recomendadas para aplicações que necessitam de estruturas dinâmicas.

Árvore Balanceada

- **Um tipo de árvore binária de busca balanceada é a Árvore AVL (*Adelson Velsky* e *Landis*), também conhecida como árvore balanceada pela altura.**





Subárvore esquerda do nó v possui altura 2 e subárvore direita do nó v possui altura zero.

Uma árvore binária T é denominada AVL quando, para qualquer nó de T, as alturas das suas duas subárvores, esquerda e direita, diferem no máximo uma unidade (em módulo).

Balanceamento de Árvores AVL

- **Caso a árvore AVL não esteja balanceada é necessário seu balanceamento através da rotação simples ou rotação dupla.**
  - **Após as operações de inserção e remoção pode ser requerido o balanceamento.**
  - **Para definir como realizar o balanceamento é utilizado um fator de balanceamento (FB).**

Fator de Balanceamento (FB)

- **O FB de um nó é dado pelo seu peso em relação a sua subárvore.**

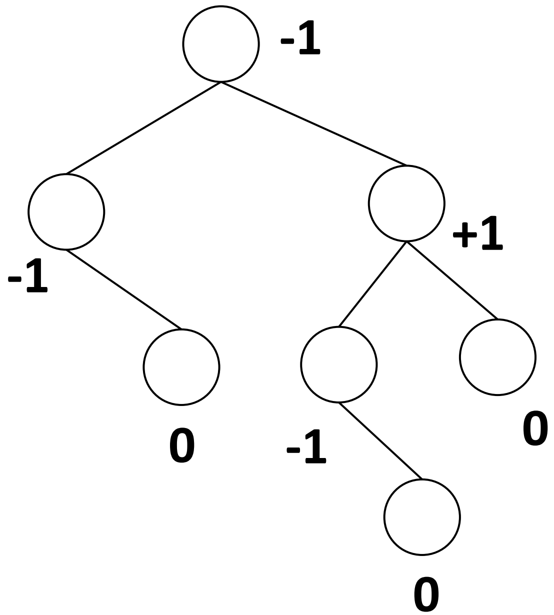
- **Não é necessário balancear se um nó tiver um FB = 0, 1 ou -1.**

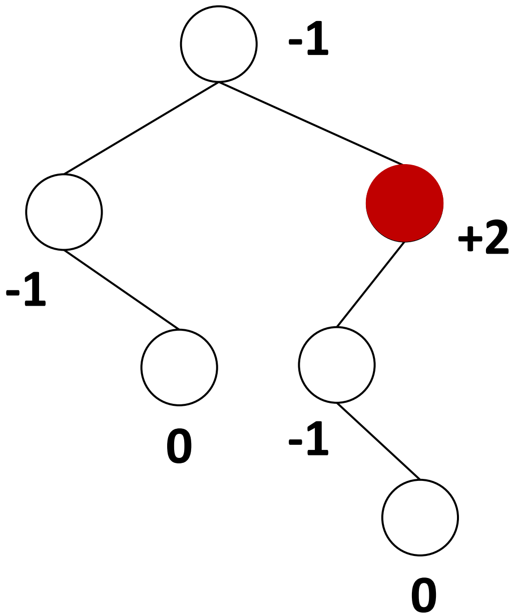
- **Um nó com FB > 1 é uma árvore não AVL e deve ser balanceada.**

  - **Após cada inclusão ou remoção verificar se algum nó da árvore está desregulado, ou seja, a diferença de altura entre as duas subárvores é maior do que 1.**

Exemplo de Cálculo de FB

1. **Todo nó folha tem FB = 0**
2. **Calcular FB para cada nó (h da subárvore esq. - h da subárvore dir.)**
   - **Altura (h) é o número de níveis e não quantidade de nós.**

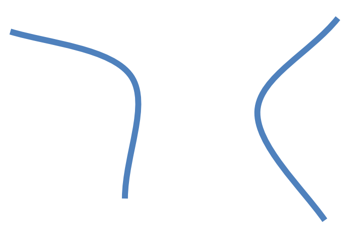




Rotação

- **A rotação na árvore AVL ocorre devido ao seu desbalanceamento.**
  - **Após a inserção ou remoção de um nó.**
- **Uma rotação simples ocorre quando um nó está desbalanceado e seu filho estiver no mesmo sentido da inclinação, formando uma linha reta.**

- **Uma rotação dupla ocorre quando um nó estiver desbalanceado e seu filho estiver inclinado no sentido inverso ao pai, formando um "joelho".**

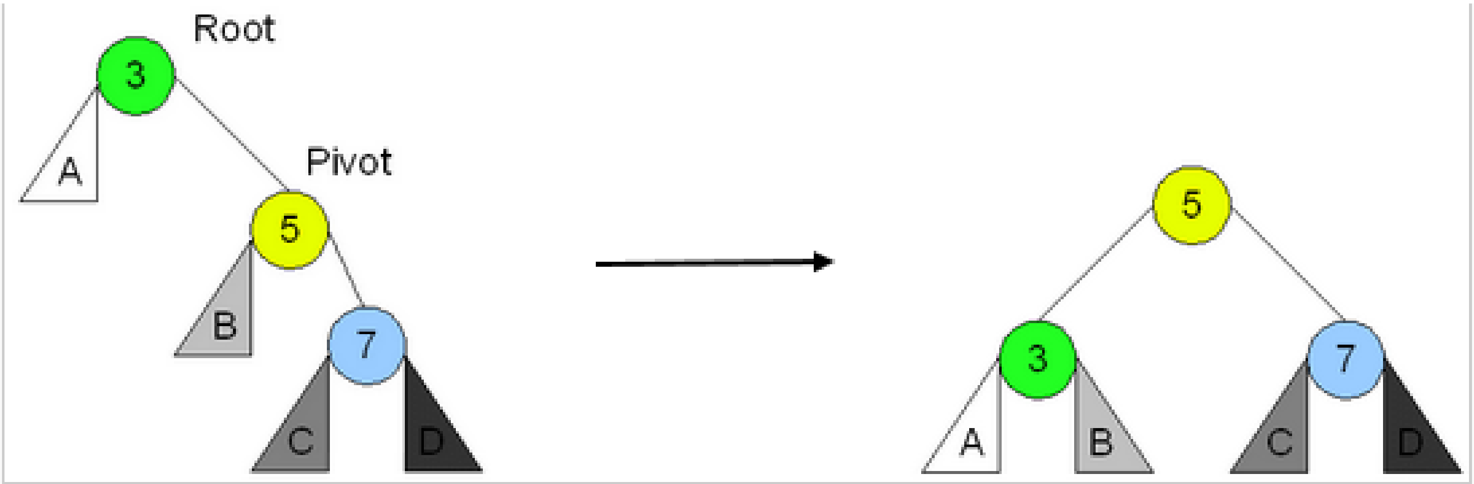


Rotação

- **Seja P o nó pai, FE o filho à esquerda de P e FD o filho à direita de P.**
- **Existem quatro tipos de rotação:**
  - **Rotação Simples à Esquerda (RSE)**
  - **Rotação Simples à Direita (RSD)**
  - **Rotação Dupla à Esquerda (RDE)**
  - **Rotação Dupla à Direita (RDD)**

Rotação Simples à Esquerda (RSE)

- **Deve ser efetuada quando a diferença das alturas h dos filhos de P é igual a -2 e a diferença das alturas h dos filhos de FD é igual a -1.**
- **O nó FD torna-se o novo pai e o nó P torna-se o filho da esquerda do FD.**
  - **Seja Y o filho à direita de X**
  - **Torne o filho à esquerda de Y o filho à direita de X.**
  - **Torne X filho à esquerda de Y**

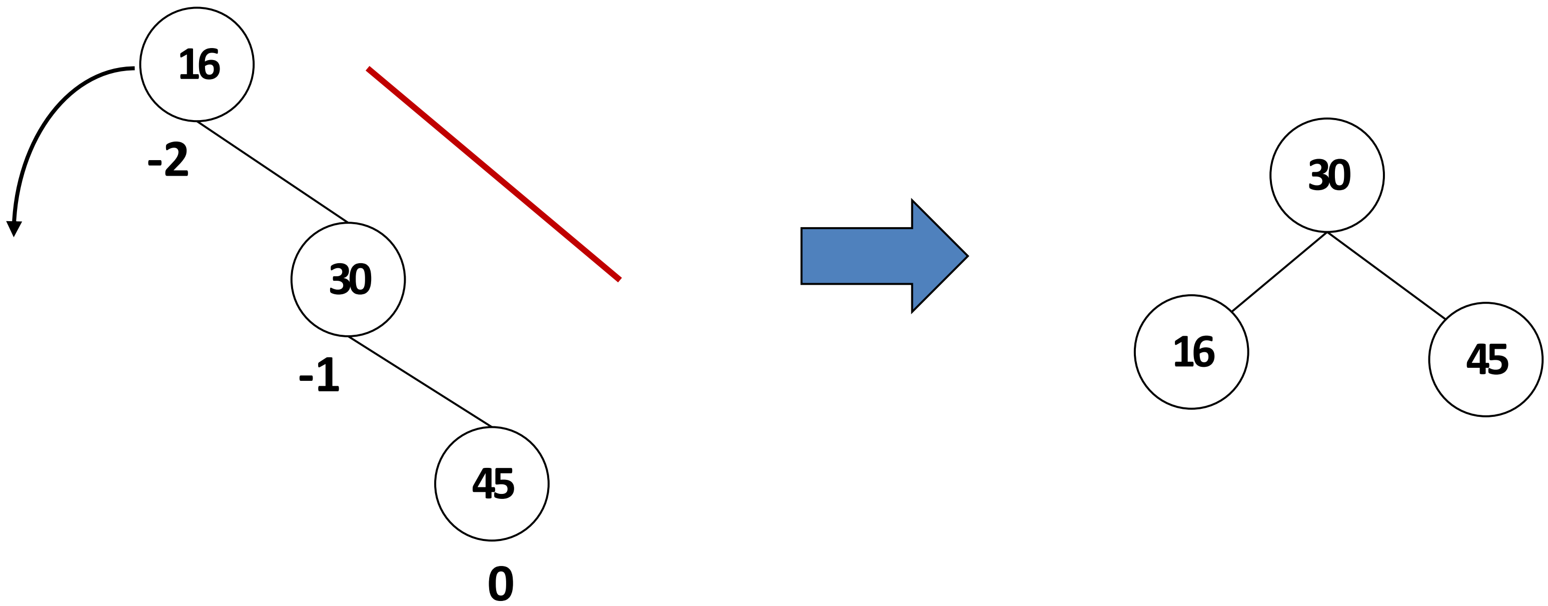


A árvore da esquerda ficou desbalanceada após a inserção do nó 7, necessitando assim de uma rotação a esquerda. Após a rotação o nó 5 virou o pai dos nós 3 e 7.

Rotação Simples à Esquerda (RSE)

- **Construir uma árvore AVL usando a sequência:**

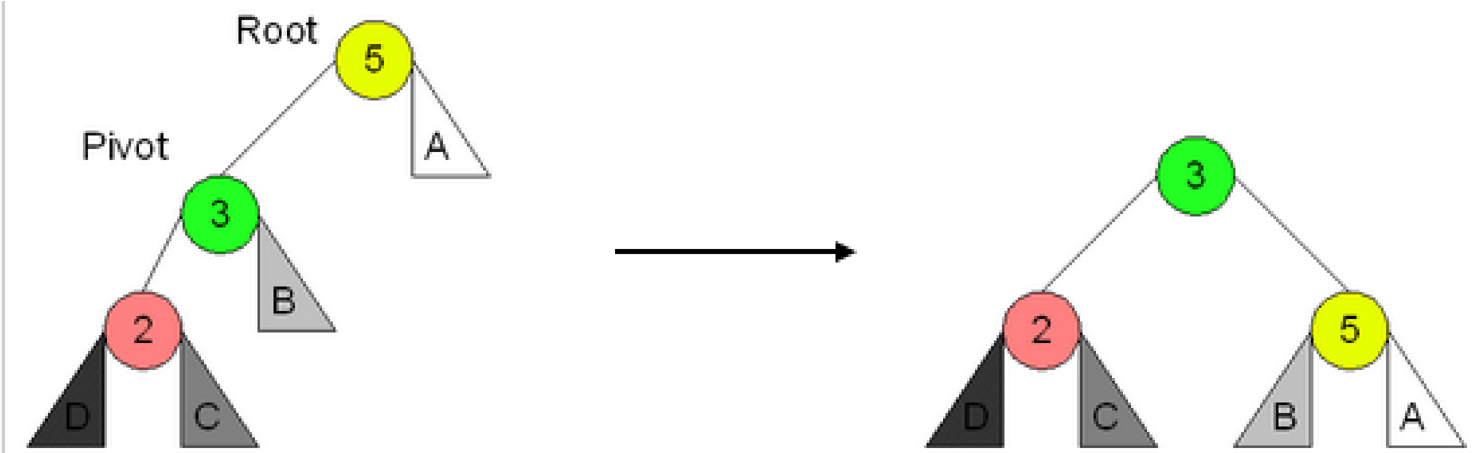
16 - 30 - 45 - 10 - 8 - 70 - 60 - 9 - 12 - 20 - 13



- **Sinais negativos significa que a árvore tem mais peso à direita.**
- **Sinais iguais (-- ou ++) temos uma reta. Neste caso a reta tendendo para o lado direito ⇨ aplicar RSE. Sentido anti-horário.**
- **Nó P (16) com problema: o FD (30) torna-se pai. E o nó P (16) torna-se filho à esquerda do FD (30).**

Rotação Simples à Direita (RSD)

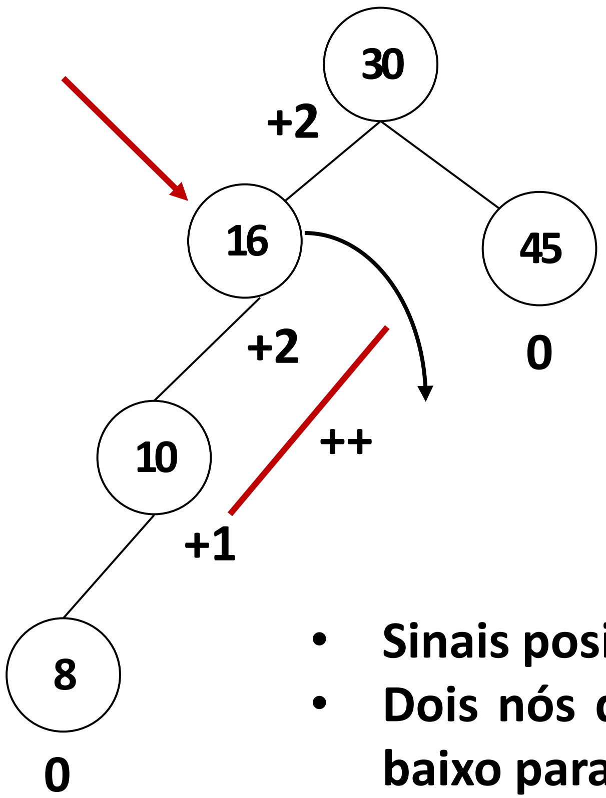
- **Deve ser efetuada quando a diferença das alturas h dos filhos de P é igual a +2 e a diferença das alturas h dos FE é igual a +1.**
- **O nó FE torna-se o pai e o nó P torna-se o filho da direita de FE.**
  - **Seja Y o filho à esquerda de X.**
  - **Torne o filho à direita de Y o filho à esquerda de X.**
  - **Torne X o filho à direita de Y.**

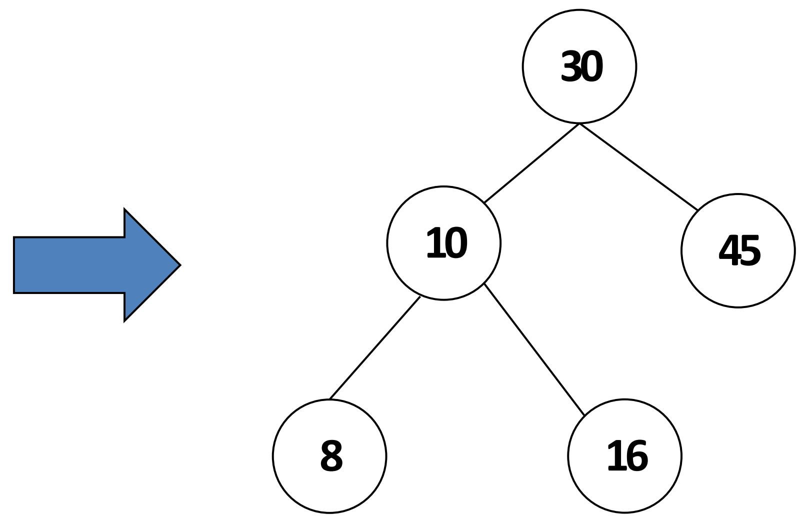


A árvore da esquerda ficou desbalanceada após a inserção do nó 2, necessitando assim de uma rotação a direita. Após a rotação o nó 3 virou o pai dos nós 2 e 5.

Rotação Simples à Direita (RSD)

~~16~~ - ~~30~~ - ~~45~~ - 10 - 8 - 70 - 60 - 9 - 12 - 20 - 13

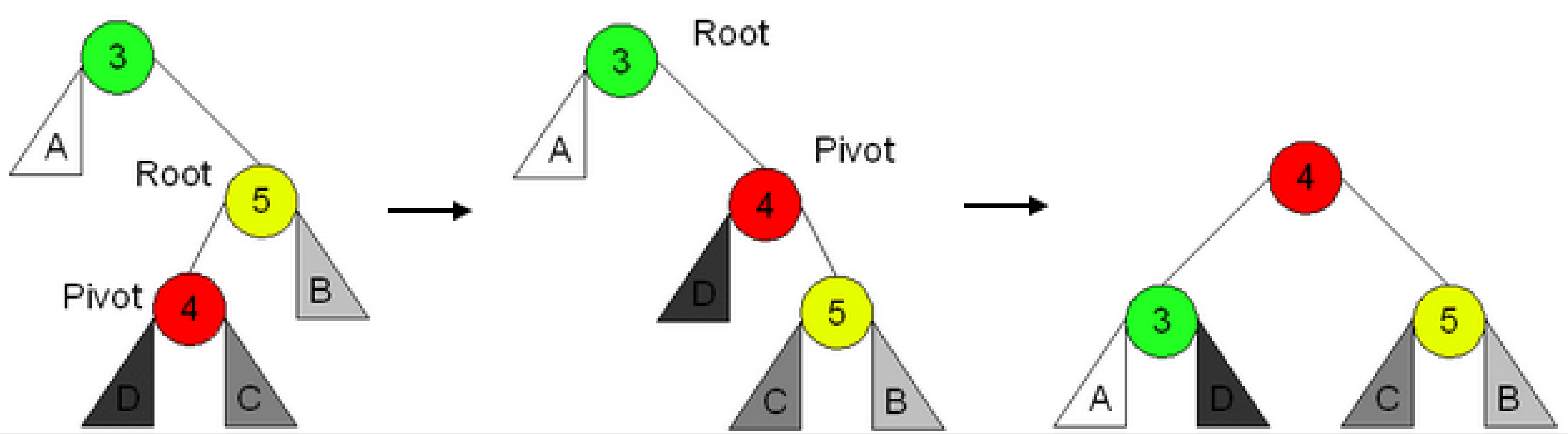




- **Sinais positivos ⇨ árvore tem mais peso à esquerda.**
- **Dois nós desbalanceados (16 e 30). Tratar o primeiro deles: de baixo para cima (16).**
- **Sinais iguais (-- ou ++) temos uma reta. Neste caso a reta tendendo para o lado esquerdo ⇨ aplicar RSD. Sentido horário.**
- **Nó P (16) com problema: o FE (10) torna-se pai. E o nó P (16) torna-se filho à direita do FE (10).**

Rotação Dupla à Esquerda (RDE)

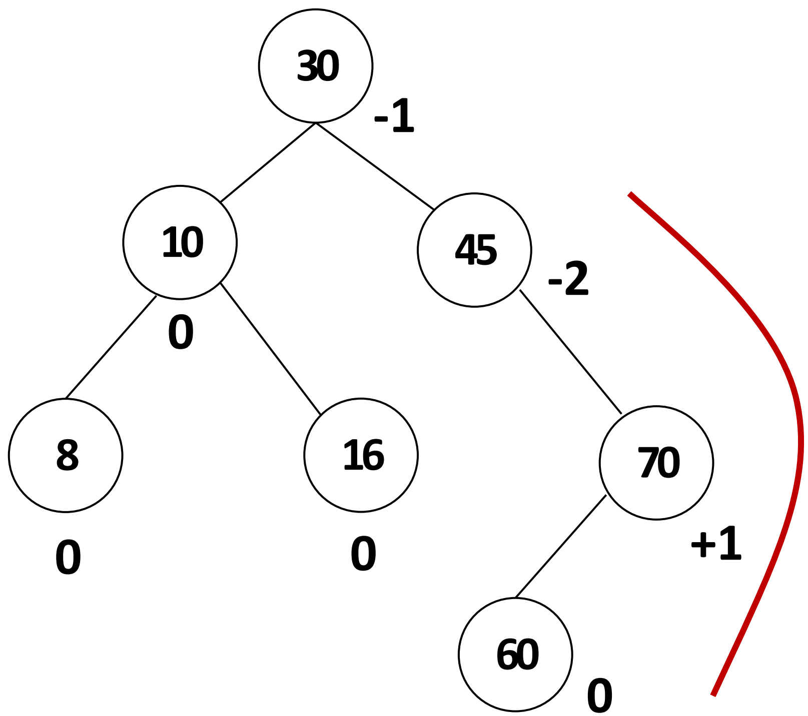
- **Efetuada quando a diferença das alturas h dos filhos de P é igual a -2 e a diferença das alturas h dos FD é igual a +1.**
  - **Aplicar uma rotação à direita no nó FD e depois uma rotação à esquerda no nó P.**



A árvore da primeira figura ficou desbalanceada após a inserção do nó 4. Para balanceá-la foi executada uma rotação dupla à esquerda, ou seja, primeiro foi executada uma rotação à direita no nó 5 (segunda figura) e depois foi executada uma rotação à esquerda no nó 3 (terceira figura).

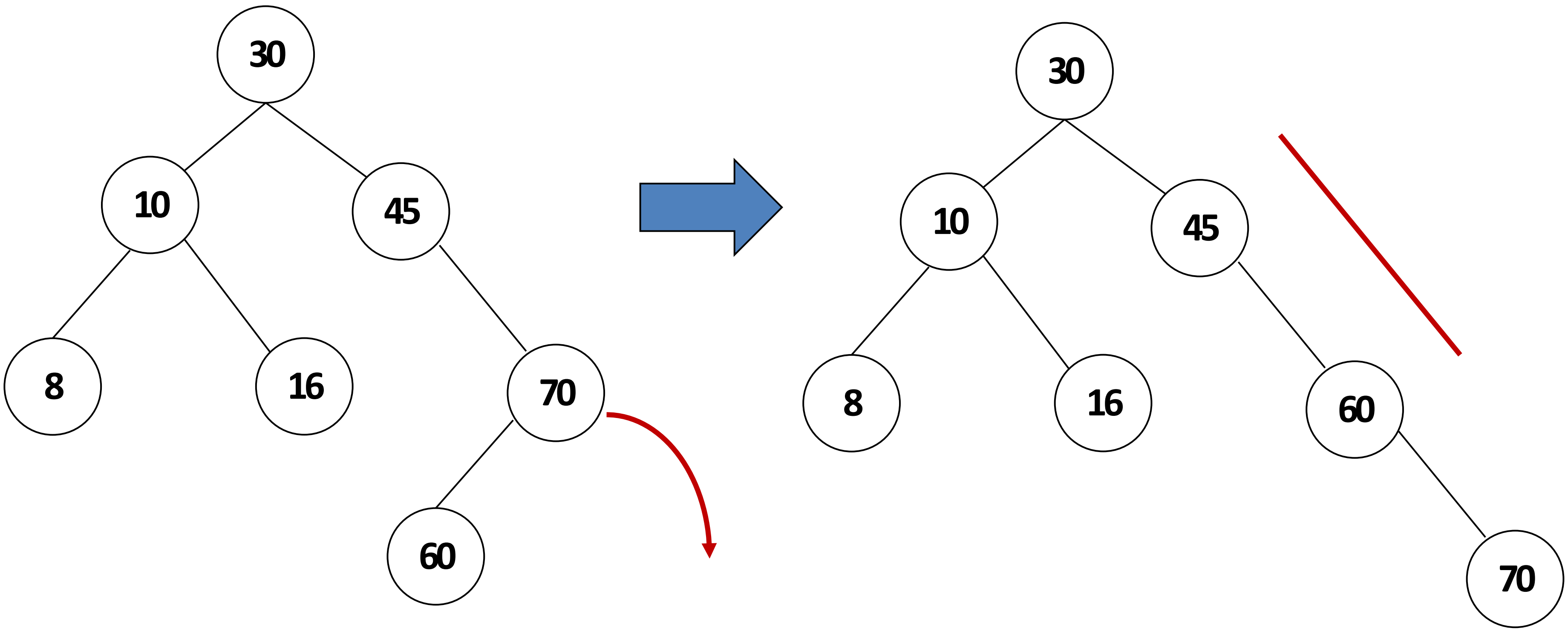
Rotação Dupla à Esquerda (RDE)

~~16~~ - ~~30~~ - ~~45~~ - ~~10~~ - ~~8~~ - 70 - 60 - 9 - 12 - 20 - 13



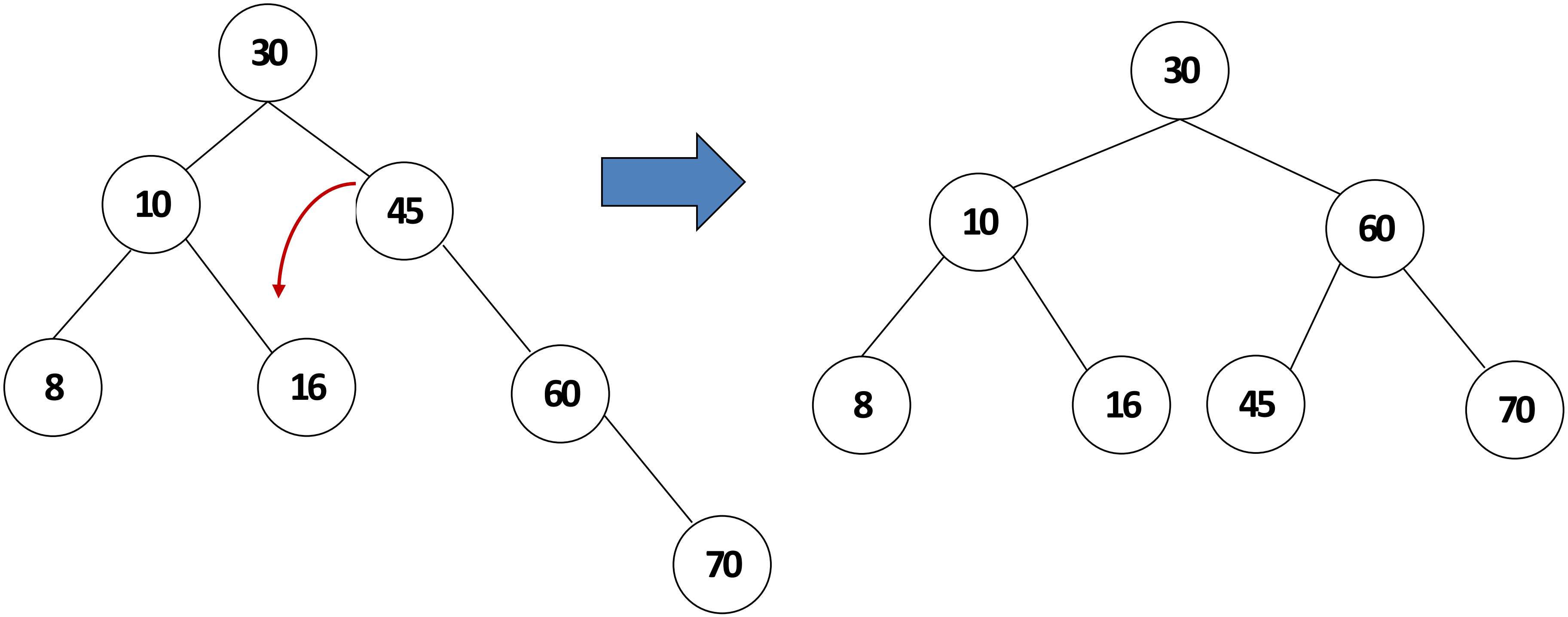
- **Nó com problema = 45. Fator negativo ⇨ verificar sinal do FD (70). Neste caso sinal positivo.**
- **Sinais diferentes (+- ou -+) temos um "joelho". No exemplo, o "joelho" está tendendo para o lado direito ⇨ aplicar RDE (RSD + RSE).**
- **RSD vai converter o "joelho" numa reta e RSE vai balancear a árvore.**

RDE (1ª fase: aplicar RSD(70,60))



- **Aplicar a RSD nos nós 70 e 60.**
- **Árvore continua desbalanceada ⇨ Sinais dos FBs iguais (--)**
- **Aplicar a RSE.**

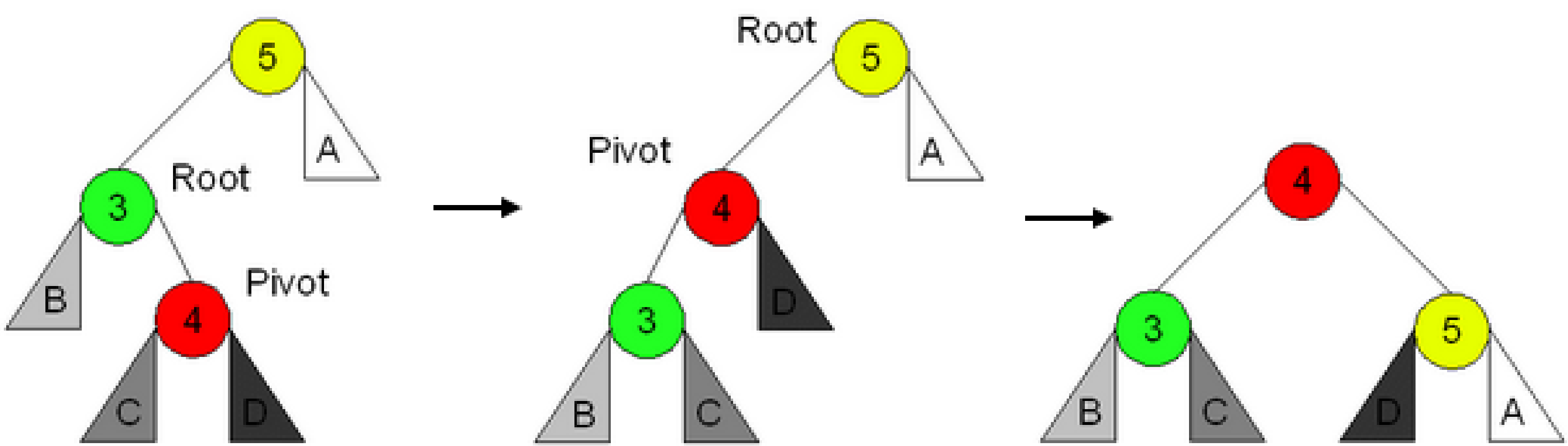
RDE (2ª fase: aplicar RSE(45,60)



- **Aplicar RSE (45, 60) - sentido anti-horário.**
- **Nó P (45) com problema: o FD (60) torna-se pai. E o nó P (45) torna-se filho à esquerda do FD (60).**

Rotação Dupla à Direita (RDD)

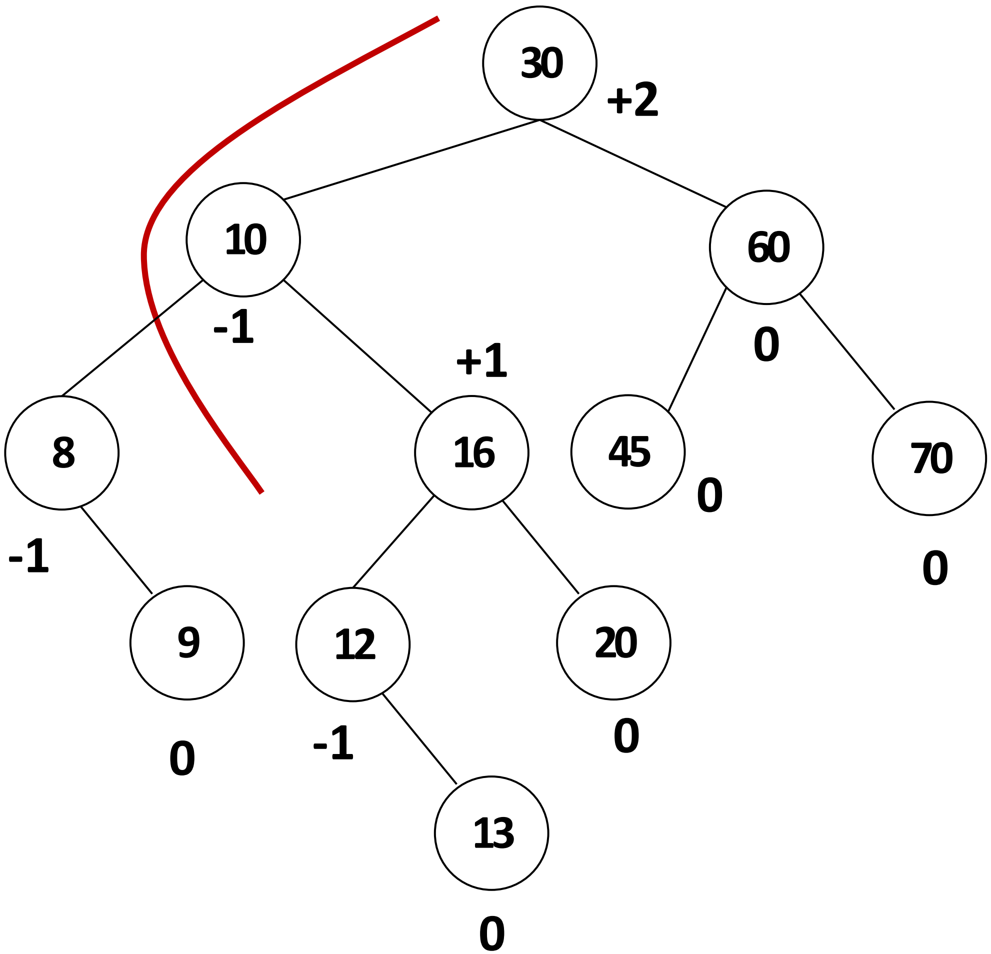
- **Efetuada quando a diferença das alturas h dos filhos de P é igual a +2 e a diferença das alturas h dos FE é igual a -1.**
  - **Aplicar uma rotação à esquerda no nó FE e depois uma rotação à direita no nó P.**



A árvore da primeira figura ficou desbalanceada após a inserção do nó 4. Para balanceá-la foi executada uma rotação dupla à direita, ou seja, primeiro foi executada uma rotação à esquerda no nó 3 (segunda figura) e depois foi executada uma rotação à direita no nó 4 (terceira figura).

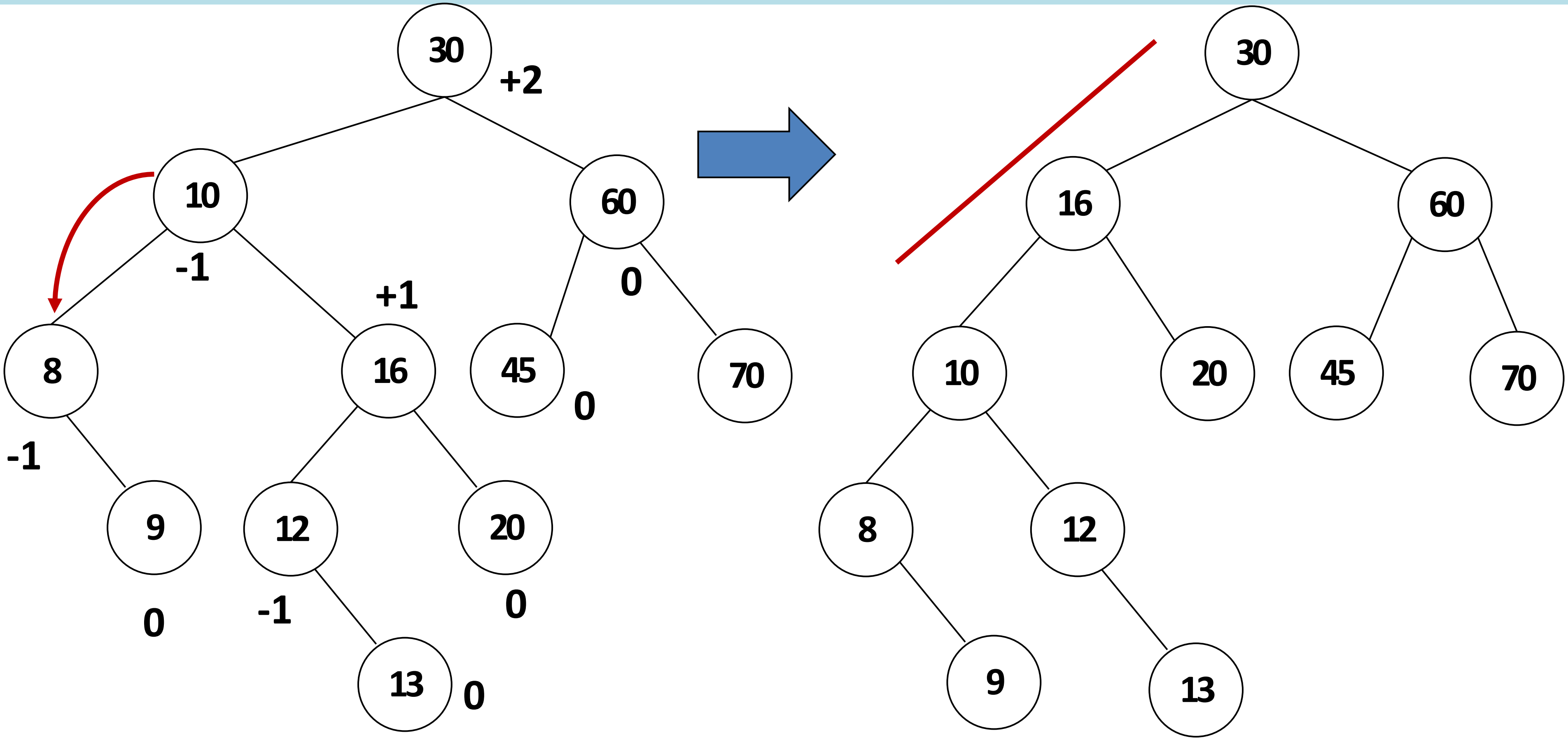
Rotação Dupla à Direita (RDD)

~~16~~ - ~~30~~ - ~~45~~ - ~~10~~ - ~~8~~ - ~~70~~ - ~~60~~ - 9 - 12 - 20 - 13



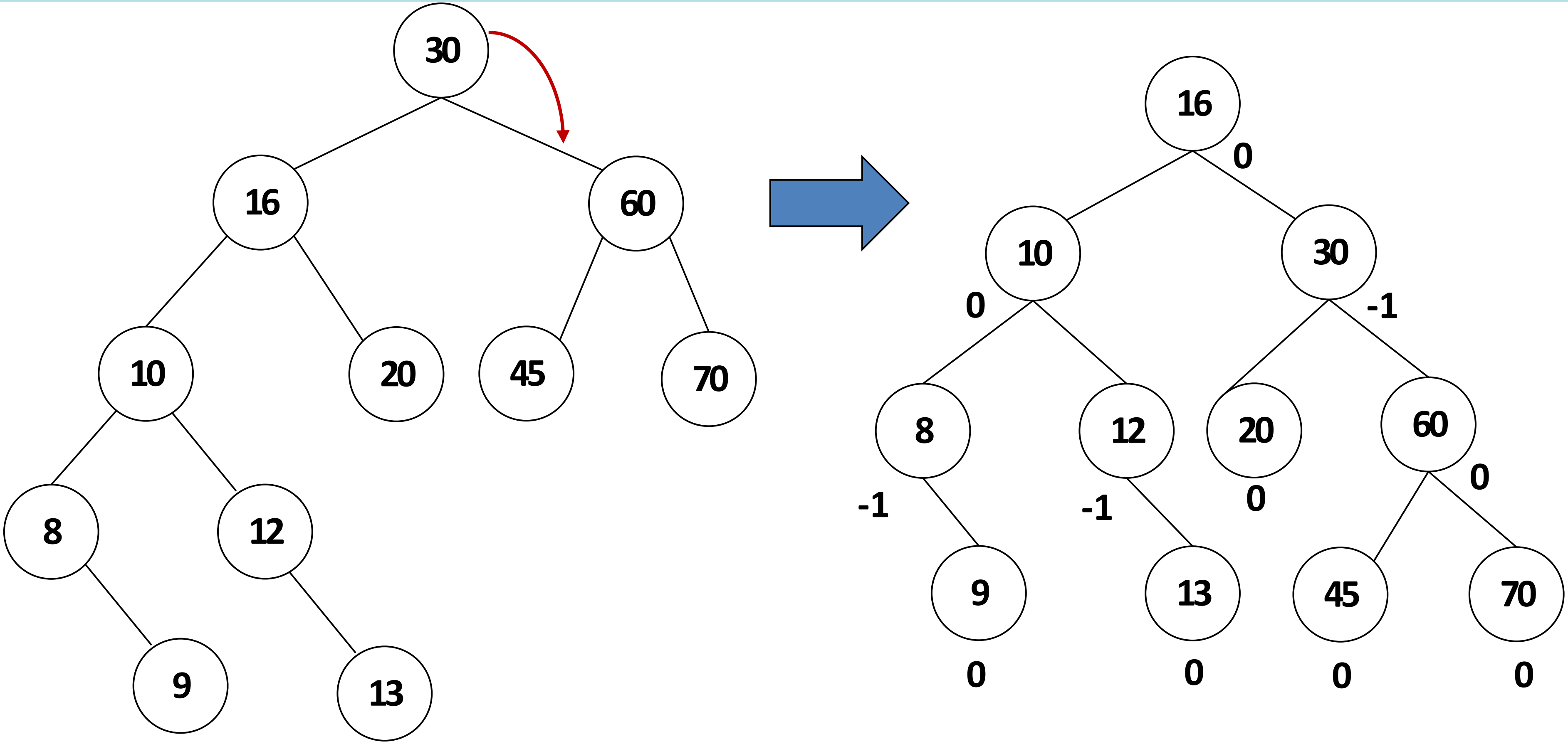
- **Nó desbalanceado = 30. Fator positivo (subárvore da esquerda mais pesada) ⇨ verificar sinal do FE (10) ⇨ sinal negativo.**
- **Sinais diferentes (+- ou -+) temos um "joelho".**
- **No exemplo, o "joelho" está tendendo para o lado esquerdo ⇨ aplicar RDD (RSE + RSD).**
- **RSE vai converter o "joelho" numa reta e RSD vai balancear a árvore.**

RDD (1ª fase: aplicar RSE(10,16)



- **Aplicar a RSE nos nós 10 e 16. O nó 10 será rotacionado à esquerda passando a ser filho do nó 16, tomando o lugar do nó 12. O nó que perdeu o lugar (12) passa a ser filho do nó que assumiu seu lugar (10).**
- **Árvore continua desbalanceada ⇨ Sinais dos FBs iguais (++)**
- **Aplicar a RSD (30,16).**

RDD (2ª fase: aplicar RSD(30,16)



- Aplicar RSD (30, 16) - sentido horário.
- Nó P (30) com problema: o FE (16) torna-se pai. E o nó P (30) torna-se filho à direita do FE (16).
- O nó 20 (FD do nó 16) passa a ser FE do nó 30.

Operações da Árvore AVL

- **Principais operações na Árvore AVL:**
  - **Busca de elementos**
  - **Inserção de novo elemento (nunca com chave igual)**
  - **Remoção de elemento**
- **Na árvore AVL, todas as operações possuem complexidade O(log n), onde *n* é o numero de nós pertencentes à arvore.**

Busca na Árvore AVL

- **Para realizar um operação de busca de um nó X na árvore AVL, deve-se primeiro comparar o valor do nó raiz com o valor do nó X.**
  - **Se X for menor do que a raiz efetua-se a busca pela subárvore esquerda.**
  - **Se X for maior do que a raiz efetua-se a busca pela subárvore direita.**
    - **Estes procedimentos são recursivos.**

Inserção na Árvore AVL

- **Para realizar a operação de inserção de um novo nó X na árvore AVL é necessário executar uma busca por X nesta árvore.**
  - **Após a busca o local correto para a inserção do nó X será encontrado.**
  - **Depois de inserido o nó, a altura do nó pai e de todos os nós acima deve ser atualizada.**
  - **Caso necessário o algoritmo de rotação deve ser chamado para o nó pai e depois para todos os nós superiores se um desses nós tiver com FB > 1.**

Remoção na Árvore AVL

- **Para realizar a operação de remoção de um novo nó X na árvore AVL é necessário executar uma busca por X nesta árvore.**
  - **X é folha da árvore ⇨ apagar nó X.**
  - **X não é folha da árvore ⇨ substituir o valor de X com o valor mais próximo possível menor ou igual a X pertencente à árvore.**
    - **Para encontrar este valor deve-se percorrer a subárvore da direita do filho da esquerda de X, até encontrar o maior valor M desta subárvore. O valor de X será substituído por M, X será apagado da árvore e caso M tenha um filho à esquerda esse filho ocupará sua antiga posição na árvore.**
    - **Depois de remover o nó, a altura do nó pai e de todos os nós acima deve ser atualizada.**
    - **Caso necessário o algoritmo de rotação deve ser chamado para o nó pai e depois para todos os nós superiores se um desses nós tiver com FB > 1.**

Referências

- **PIVA Jr, Dilermando et al. *Estruturas de Dados e Técnicas de Programação*. Ed. Campus**
- **MARKENZON e SZWARCFITER. *Estruturas de Dados e seus Algoritmos*. Ed. LTC**
- **FERRARI, Roberto et al. Estruturas de Dados com Jogos. Ed. Campus**