DCC405 – ESTRUTURA DE DADOS II

Aula 04 – Árvores Balanceadas AVL

Árvores Balanceadas

Como vimos, muitas operações em árvores binárias de busca têm eficiência proporcional à **altura (h)**:

- busca - O(h).
- min (max) - O(h).
- predecessor (sucessor) - O(h).
- percurso ordenado - O(n).
- seleção - O(h).
- inserção - O(h).
- remoção - O(h).

Árvores Balanceadas

Como vimos, muitas operações em árvores binárias de busca têm eficiência proporcional à **altura (h)**:

- busca - O(h).
- min (max) - O(h).
- predecessor (sucessor) - O(h).
- percurso ordenado - O(n).
- seleção - O(h).
- inserção - O(h).
- remoção - O(h).

Árvores Balanceadas

Lembre que a altura de uma árvore binária varia entre lg n e n,

lg n <= h <= n

- Assim, surge o desejo de limitar o crescimento da altura da árvore.

Uma árvore binária é balanceada se

- sua altura é da ordem de lg n, i.e., O(lg n).

Note que é fácil uma árvore ficar muito desbalanceada.

- Por exemplo, basta inserir os elementos em ordem.

Árvores Balanceadas

Lembre que a altura de uma árvore binária varia entre lg n e n,

$$lg\ n <= h <= n$$

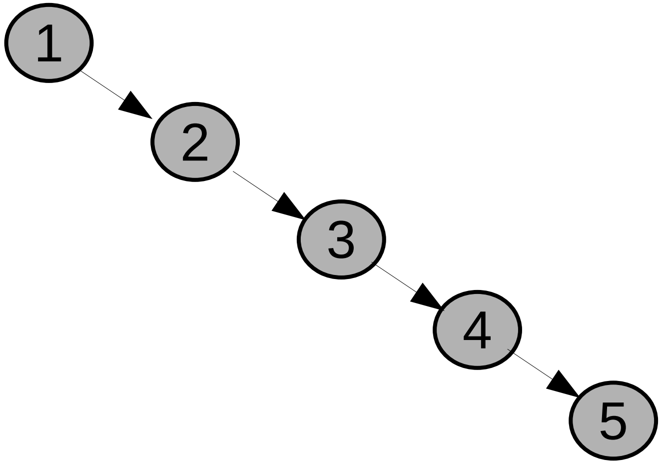
- Assim, surge o desejo de limitar o crescimento da altura da árvore.

Uma árvore binária é balanceada se

- sua altura é da ordem de lg n, i.e., O(lg n).

Note que é fácil uma árvore ficar muito desbalanceada.

- Por exemplo, basta inserir os elementos em ordem.



Árvores Balanceadas

Lembre que a altura de uma árvore binária varia entre lg n e n,

$$\lg n <= h <= n$$

- Assim, surge o desejo de limitar o crescimento da altura da árvore.

Uma árvore binária é balanceada se

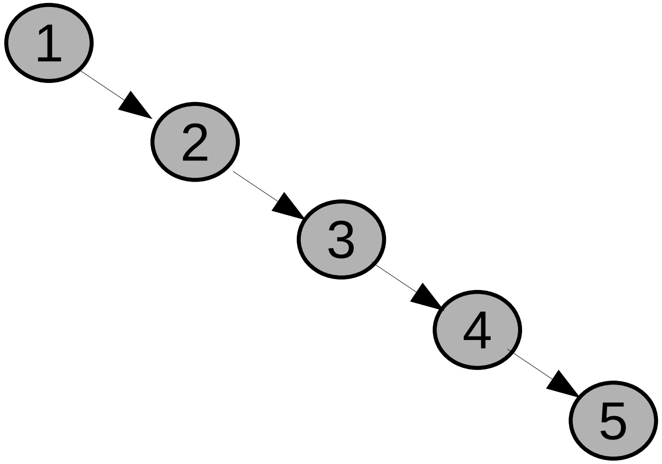
- sua altura é da ordem de lg n, i.e., O(lg n).

Note que é fácil uma árvore ficar muito desbalanceada.

- Por exemplo, basta inserir os elementos em ordem.

→ Note que, apenas as operações de inserção e remoção
- podem acabar mudando o balanceamento de uma árvore.

→ Por isso, serão estas operações que modificaremos nas próximas aulas.



Árvores Balanceadas

Existem diversas estratégias para resolver o problema do balanceamento

- e estas dão origem a diferentes árvores.
- Ex: árvores AVL, árvores rubro-negras, splay-trees, árvores 2-3, árvores B.

Várias bibliotecas possuem implementações de árvores balanceadas. Por exemplo:

- a classe **map** em C++,
- a classe **TreeMap** em Java.

Vamos estudar a estratégia de rotações

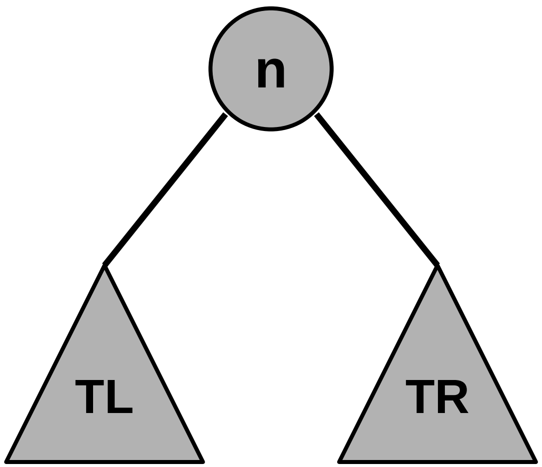
- e veremos árvores balanceadas baseadas nessa estratégia:
  - árvores AVL,
  - árvores rubro-negras.

Árvores Balanceadas - Importante

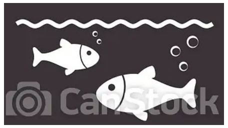
Input: 3, 2, 1

- Balanceamento
- Rotações
- Implementação

Balanceamento



Altura (height):

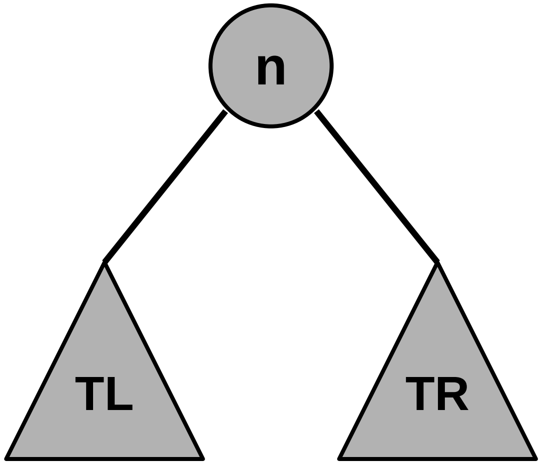


H(∅) = -1

H(single node**) = 0**

H(n) = max(H(TL)**,** H(TR)**) + 1**

Balanceamento



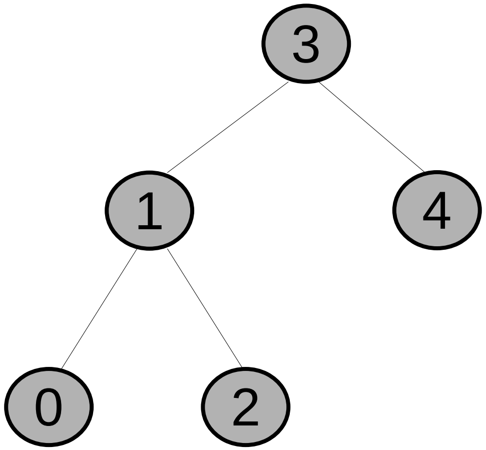
Balanceamento (balance):

B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

Também chamado de Fator de Balanceamento

Balanceamento - Exemplos

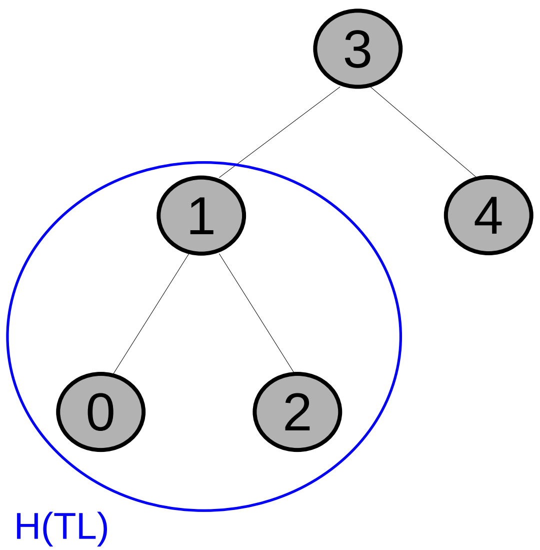


B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(3) =

Balanceamento - Exemplos

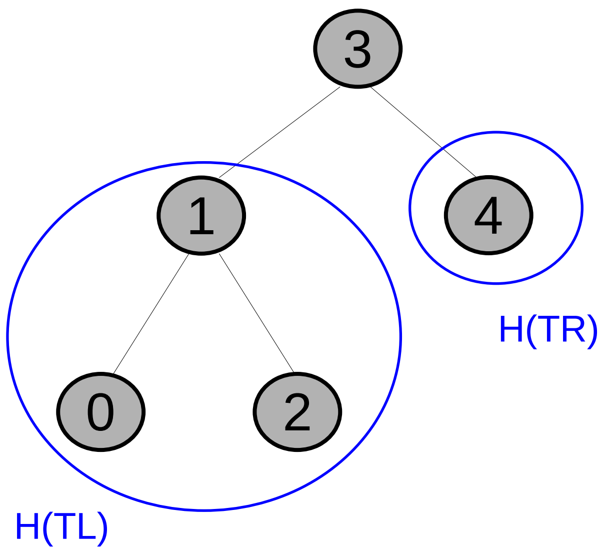


B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(3) = 1

Balanceamento - Exemplos

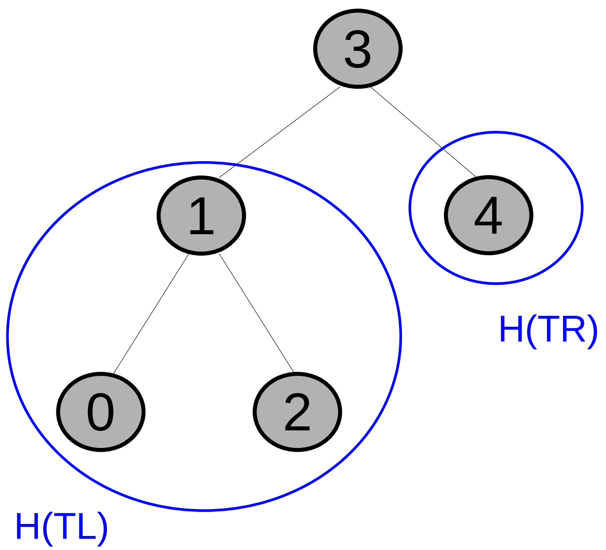


B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(3) = 1 – 0

Balanceamento - Exemplos

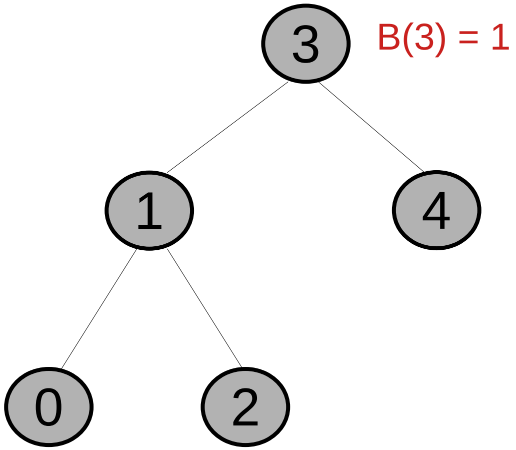


B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(3) = 1 – 0 = 1

Balanceamento - Exemplos



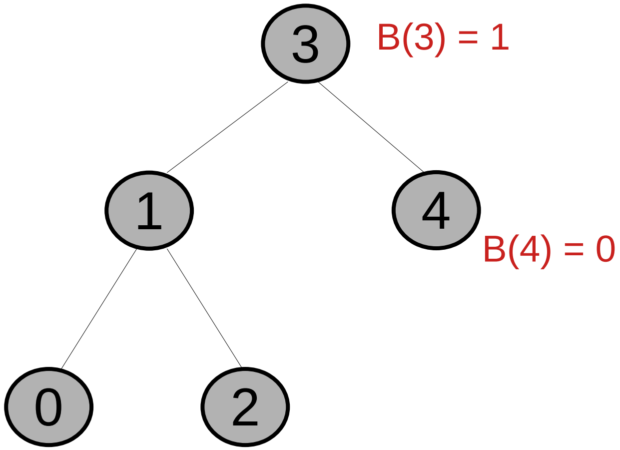
B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(3) = 1 – 0 = 1

O sinal importa.
+ : filhos à esquerda (pesa p/ esquerda)
- : filhos à direita (pesa p/ direita)

Balanceamento - Exemplos

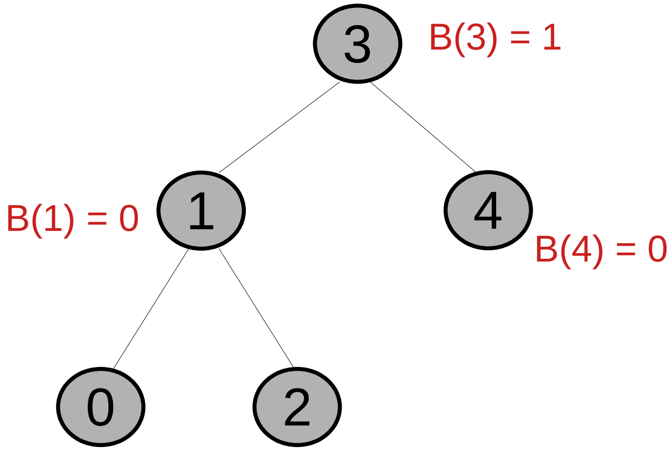


B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(3) = 1 – 0 = 1
B(4) = -1 - (-1) = 0

Balanceamento - Exemplos

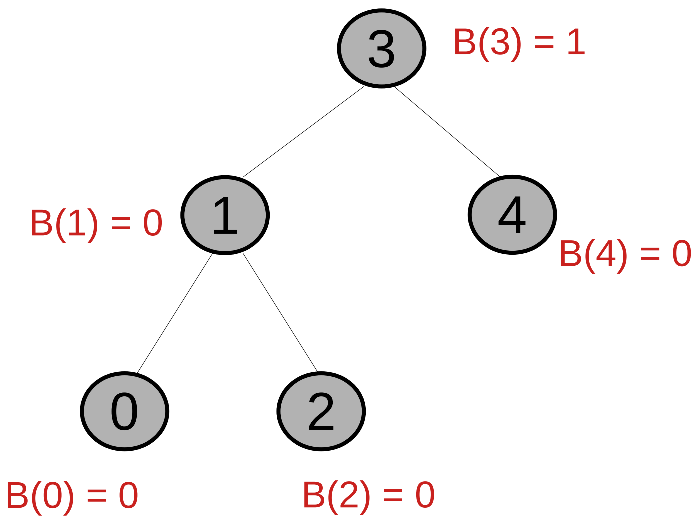


B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(3) = 1 – 0 = 1
B(4) = -1 - (-1) = 0
B(1) = 0 – 0 = 0

Balanceamento - Exemplos

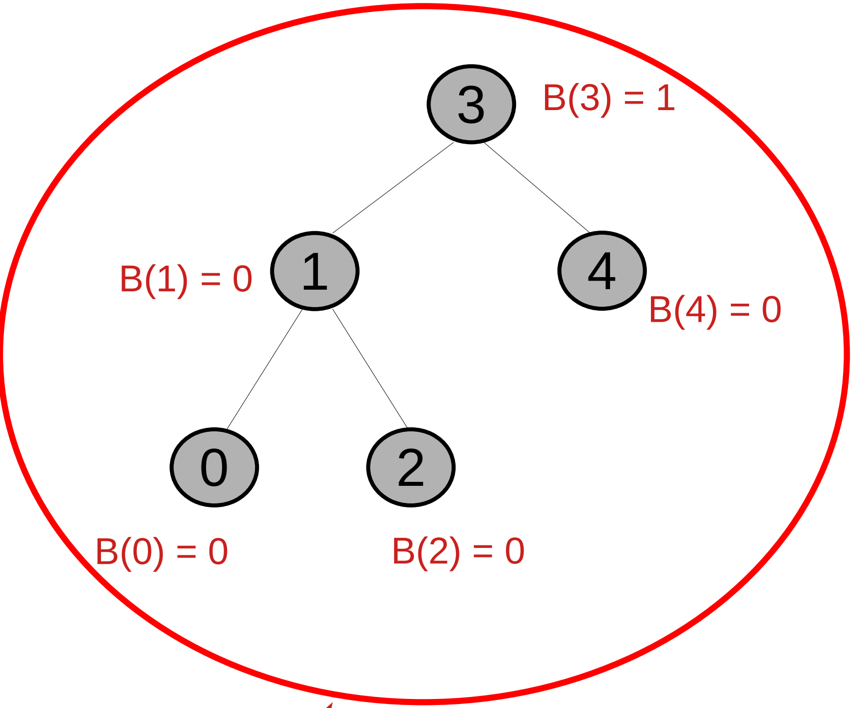


B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(3) = 1 – 0 = 1
B(4) = -1 - (-1) = 0
B(1) = 0 – 0 = 0
B(0) = -1 - (-1) = 0
B(2) = -1 - (-1) = 0

Balanceamento - Exemplos



Esta é uma Árvore AVL

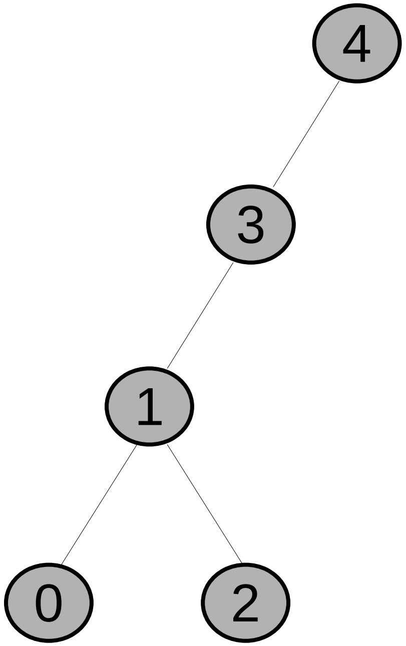
B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(3) = 1 – 0 = 1
B(4) = -1 - (-1) = 0
B(1) = 0 – 0 = 0
B(0) = -1 - (-1) = 0
B(2) = -1 - (-1) = 0

→ Ela está balanceada e seu tempo nas principais operações nessa estrutura é um tempo logarítmico.

Balanceamento - Exemplos



B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(4) =

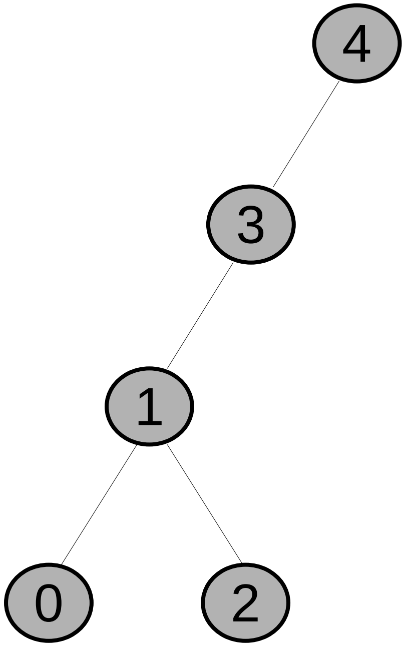
B(3) =

B(1) =

B(0) =

B(2) =

Balanceamento - Exemplos



~~**Árvore AVL**~~

B(n) = H(TL) - H(TR)

AVL Tree = | B(n) | <= 1

B(4) = 2 - (-1) = 3

B(3) = 1 - (-1) = 2

B(1) = 0

B(0) = 0

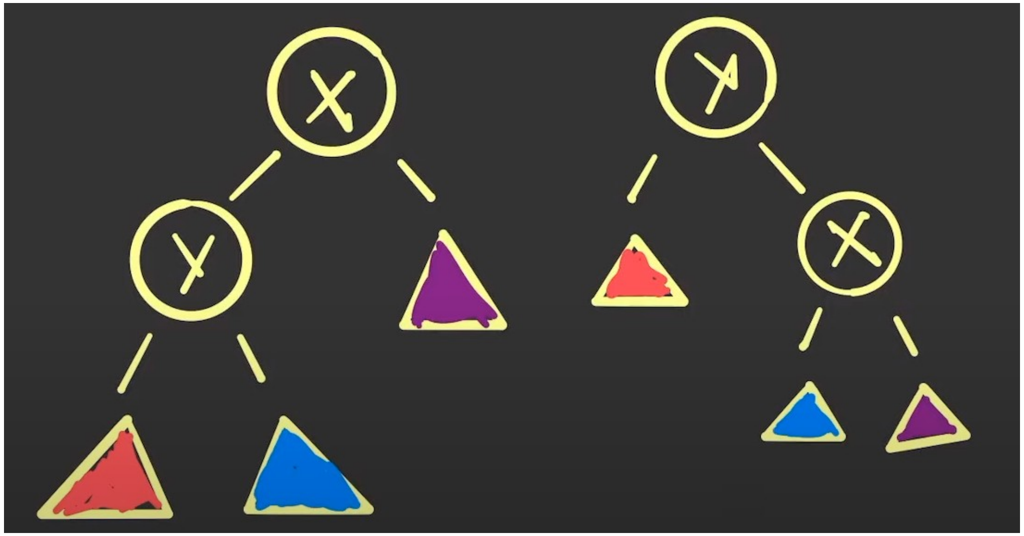
B(2) = 0

Rotações

→ Rotações servem para corrigir o **desbalanceamento**.

Desbalanceado para a Esquerda
1 – Right
2 – Left-Right

Desbalanceado para a Direita
3 – Left
4 – Right-Left



Rotações

Demonstrar

B(n) = H(TL) - H(TR)

Right

Insert: 3, 2, 1

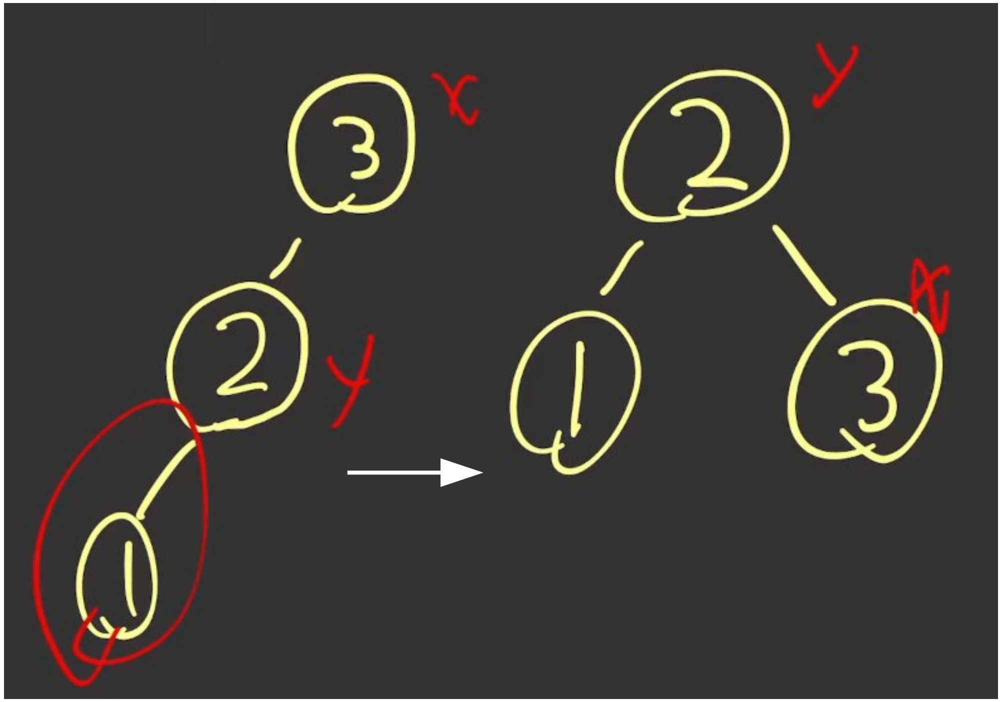
Rotações

Resposta

B(n) = H(TL) - H(TR)

Right

Insert: 3, 2, 1



(Rotação simples a direita)

Rotações

Demonstrar

B(n) = H(TL) - H(TR)

Left-Right

Insert: 3, 1, 2

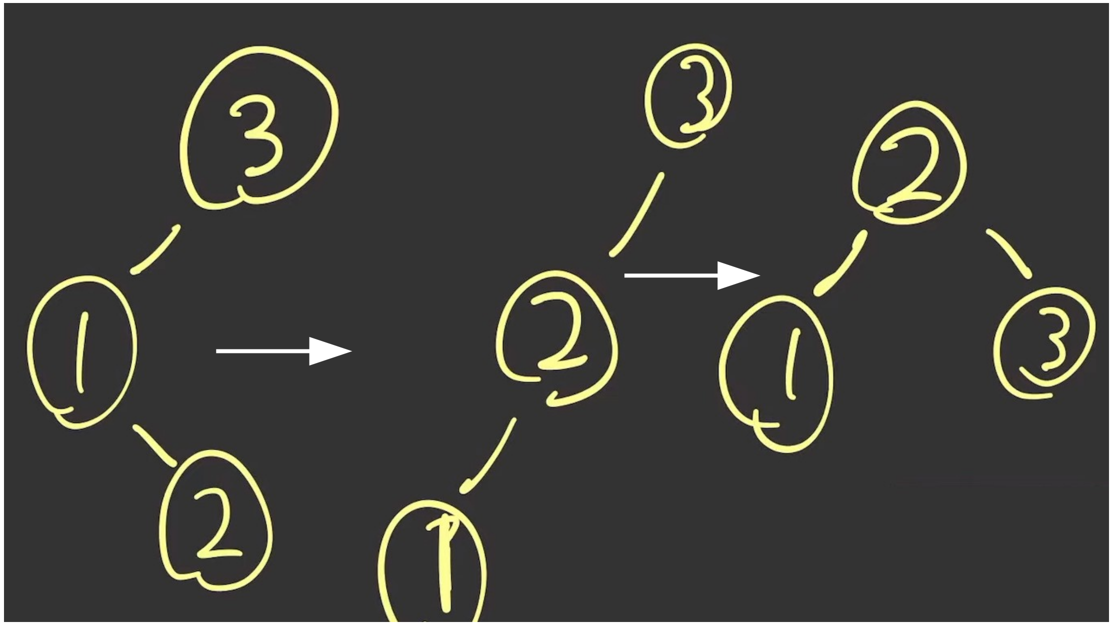
Rotações

Resposta

B(n) = H(TL) - H(TR)

Left-Right

Insert: 3, 1, 2



(Rotação dupla a direita)

Rotações

Demonstrar

B(n) = H(TL) - H(TR)

Left

Insert: 1, 2, 3

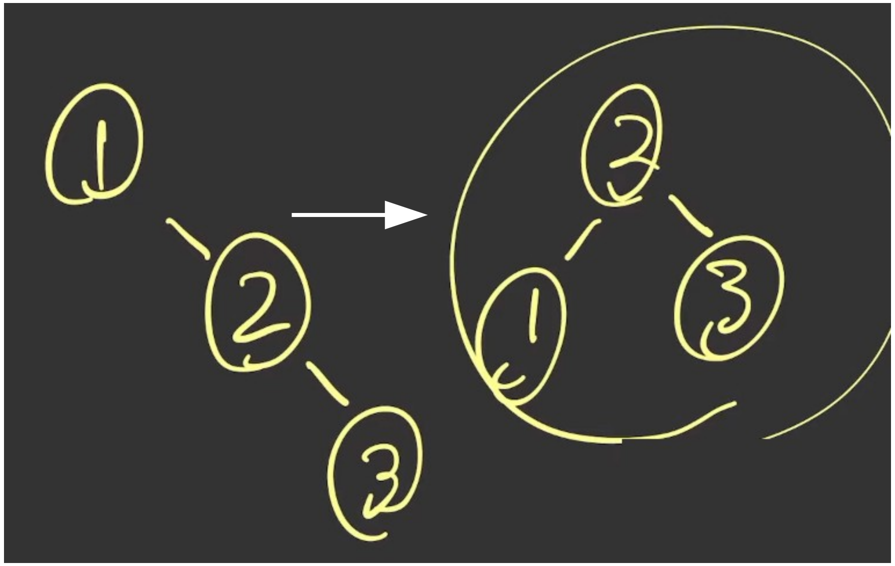
Rotações

Resposta

B(n) = H(TL) - H(TR)

Left

Insert: 1, 2, 3



(Rotação simples a esquerda)

Rotações

Demonstrar

B(n) = H(TL) - H(TR)

Right-Left

Insert: 1, 3, 2

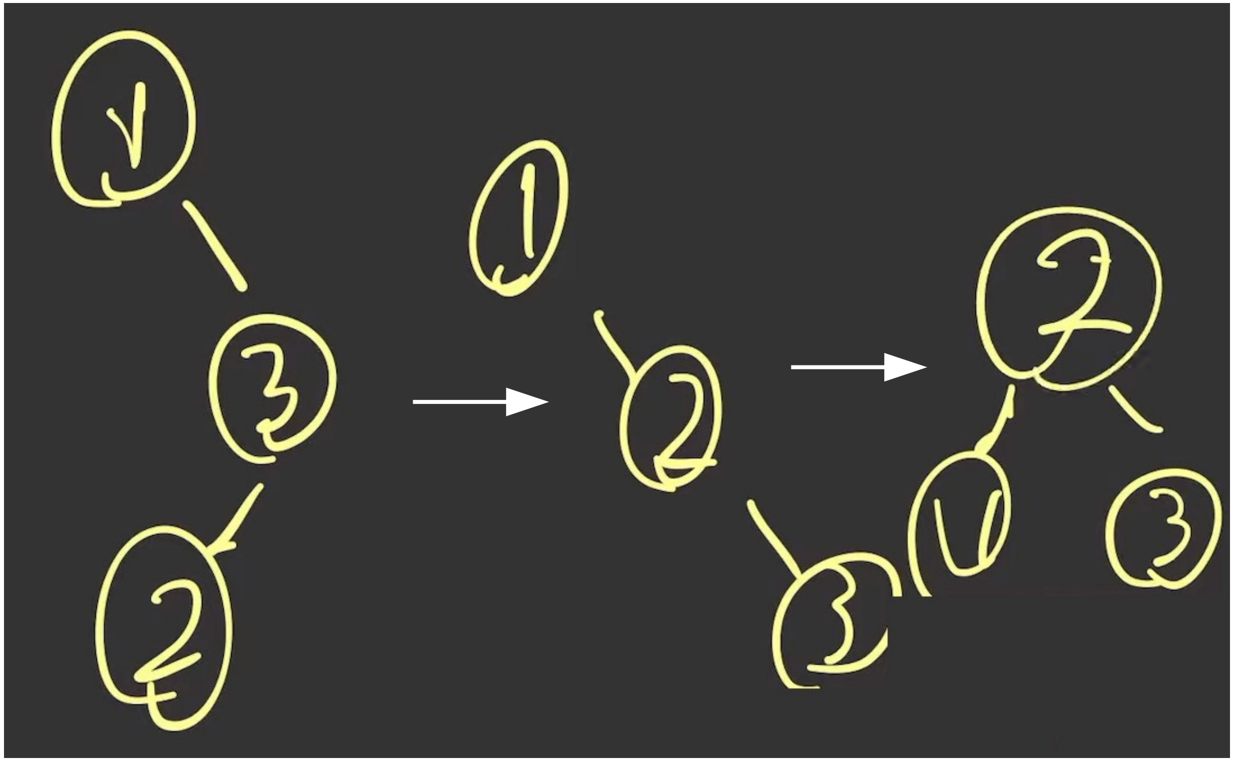
Rotações

Resposta

B(n) = H(TL) - H(TR)

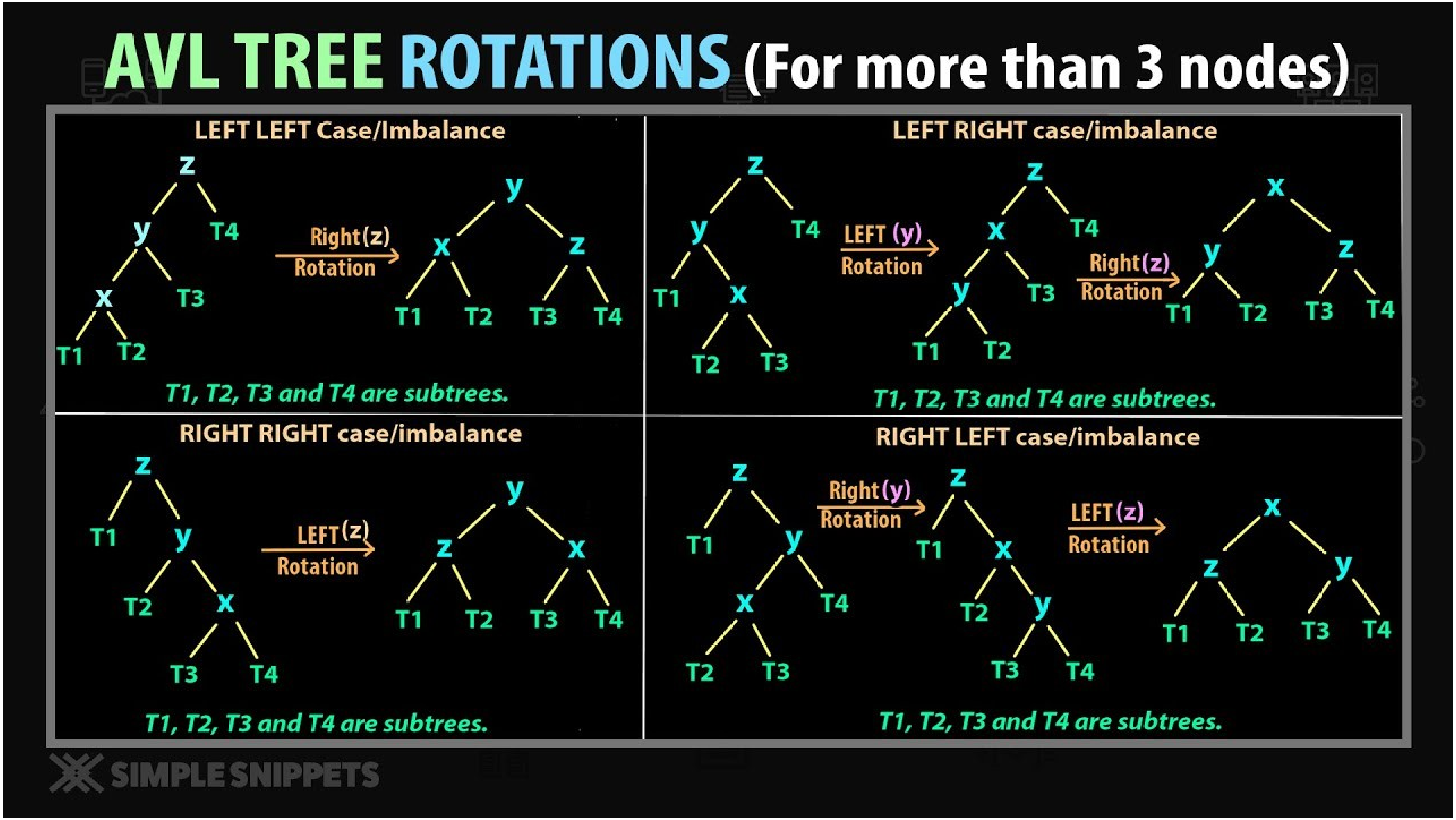
Right-Left

Insert: 1, 3, 2



(Rotação dupla a esquerda)

Rotações



AVL TREE ROTATIONS (For more than 3 nodes)
LEFT LEFT Case/Imbalance
Right(z) Rotation
T1, T2, T3 and T4 are subtrees.
LEFT RIGHT case/imbalance
LEFT (y) Rotation
Right(z) Rotation
T1, T2, T3 and T4 are subtrees.
RIGHT RIGHT case/imbalance
LEFT(z) Rotation
T1, T2, T3 and T4 are subtrees.
RIGHT LEFT case/imbalance
Right(y) Rotation
LEFT (z) Rotation
T1, T2, T3 and T4 are subtrees.
SIMPLE SNIPPETS

Implementação

Implementação...