Estrutura de Dados (CC4652)

Aula 11 – Balanceamento

Prof. Luciano Rossi

Ciência da Computação
Centro Universitário FEI

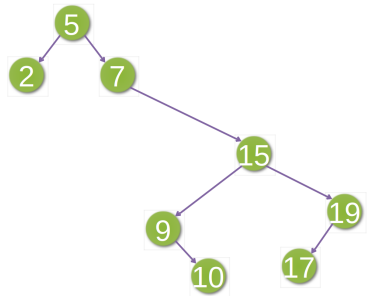
2º Semestre de 2023

Árvores Binárias de Busca

Balanceamento – Por que Árvores Balanceadas?

- A propriedade estrutural é observada na árvore abaixo;
- Quantas comparações são necessárias para recuperar os dados satélites do vértice de chave 7?

1	5
2	7
3	2
4	15
5	9
6	19
7	17
8	10



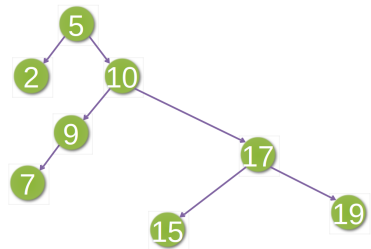
centro universitário FEI

Árvores Binárias de Busca

Balanceamento – Por que Árvores Balanceadas?

- A propriedade estrutural é observada na árvore abaixo;
- Quantas comparações são necessárias para recuperar os dados satélites do vértice de chave 7?

1	5
2	10
3	17
4	15
5	19
6	9
7	2
8	7



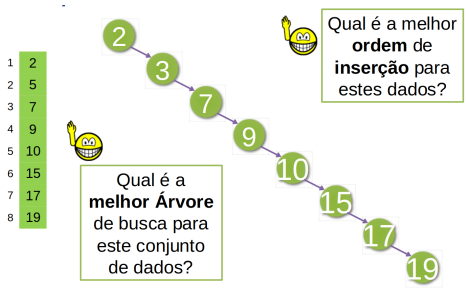
- Qual a conclusão a partir dos exemplos anteriores?

centro universitário FEI

Árvores Binárias de Busca

Balanceamento – Por que Árvores Balanceadas?

- A ordem de inserção dos valores na árvore binária de busca altera sua organização;
- Desse modo, a eficiência de busca será impactada.

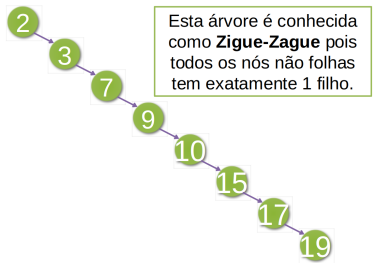


Árvores Binárias de Busca

Balanceamento – Por que Árvores Balanceadas?

- Em uma árvore binária de busca com essa organização a complexidade de busca é da ordem de $O(n)$ (complexidade observada em listas).

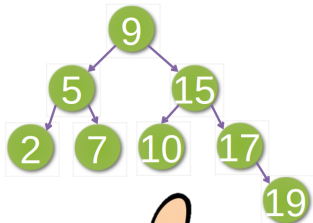
1	2
2	5
3	7
4	9
5	10
6	15
7	17
8	19

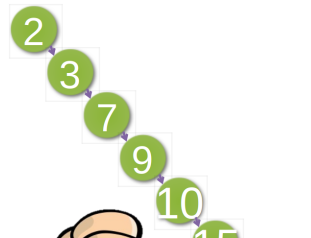


Esta árvore é conhecida como **Zigue-Zague** pois todos os nós não folhas tem exatamente 1 filho.

Árvores Binárias de Busca

Balanceamento – Qual é melhor para a busca?





centro universitário FEI

Árvores Binárias de Busca

Balanceamento - Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree

- São árvores auto-balanceadas
- As árvores AVL garantem que:

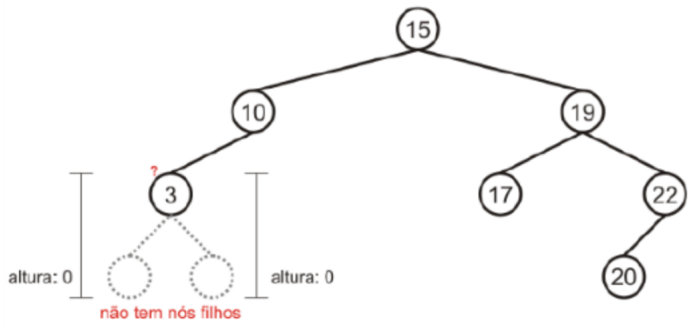
Árvores AVL

Dado qualquer nó da árvore, a diferença entre a altura do seu ramo direito e a altura do seu ramo esquerdo é de, no máximo, uma unidade.

centro universitário FEI

Árvores Binárias de Busca

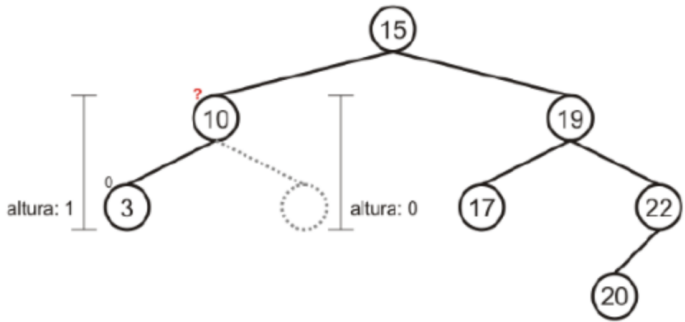
Balanceamento – Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree



- O vértice 3 está balanceado?

Árvores Binárias de Busca

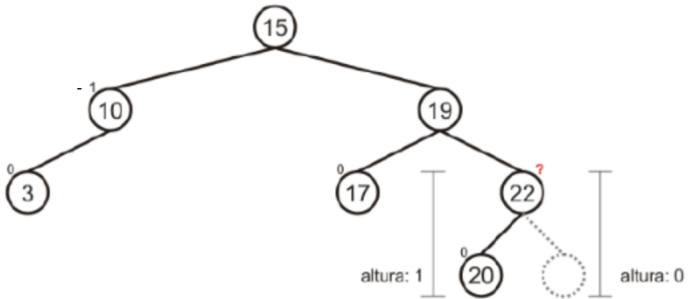
Balanceamento – Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree



- O vértice 10 está balanceado?

Árvores Binárias de Busca

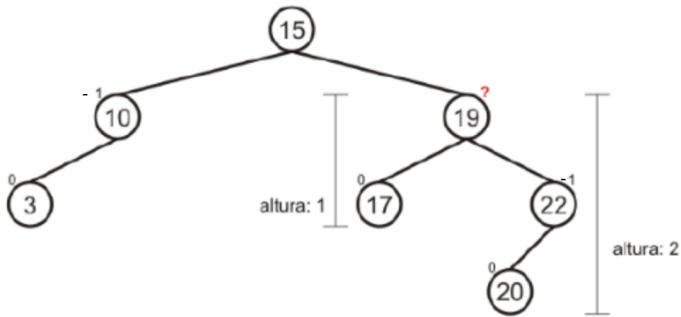
Balanceamento – Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree



- O vértice 22 está balanceado?

Árvores Binárias de Busca

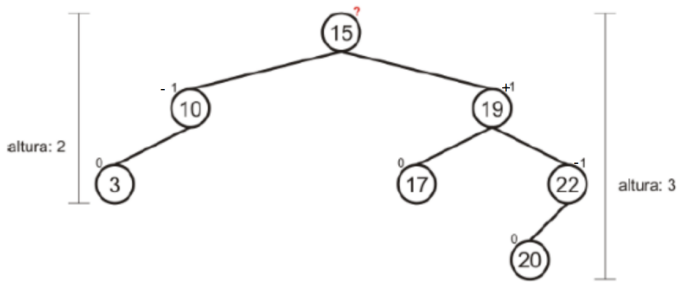
Balanceamento – Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree



- O vértice 19 está balanceado?

Árvores Binárias de Busca

Balanceamento – Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree



- O vértice 15 está balanceado?

Árvores Binárias de Busca

Balanceamento - Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree

```
1  int MAX(int x, int y) {
2      if(x >= y)
3          return x;
4      else
5          return y;
6  }
7
8  int altura(Vertice *x) {
9      if (x == NULL) {
10         return -1;
11     }
12     return MAX(altura(x->esq), altura(x->dir)) + 1;
13 }
14
15 int fatorBalanceamento(Vertice *x) {
16     return altura(x->dir) - altura(x->esq);
17 }
```

Árvores Binárias de Busca

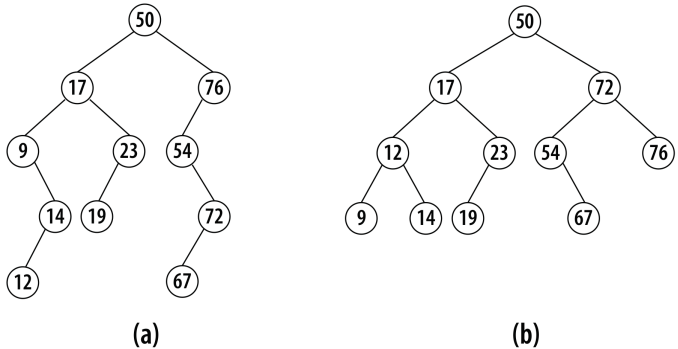
Balanceamento - Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree

- O fator de balanço (fb) de um vértice v é o valor $h_d(v) - h_e(v)$, um vértice é balanceado quando seu fb é -1, 0 ou 1.
- Uma árvore é AVL se todos os seus vértices são balanceados.

centro universitário FEI

Árvores Binárias de Busca

Balanceamento – Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree



Exemplo de árvore não AVL (a) e outra AVL (b)

Árvores Binárias de Busca

Balanceamento - Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree

- A análise das árvores AVL, por meio do fator de balanceamento de seus vértices, pode ser feita pela classificação de seus vértices em balanceados ou desbalanceados.
- Considere um vértice a ser analisado, quanto ao seu fator de balanceamento (fb).
- Dessa forma, podemos ter as seguintes classificações para vértices balanceados:
  - $fb(v) = 1$: a subárvore direita é mais alta que a esquerda;
  - $fb(v) = 0$: as subárvores têm alturas iguais;
  - $fb(v) = -1$: a subárvore esquerda é mais alta que a direita.

centro universitário FEI

- Da mesma forma, podemos classificar os vértices desbalanceados de acordo com os respectivos fatores de balanceamento:
  - $fb(v) > 1$: a subárvore direita está desbalanceando o vértice v;
  - $fb(v) < -1$: a subárvore esquerda está desbalanceando o vértice v.

Árvores Binárias de Busca

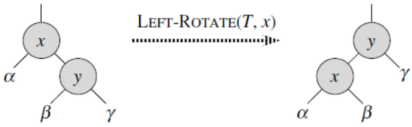
Balanceamento – Árvores Balanceadas AVL Adelson-Velskii and Landis' Tree

- A manutenção do balanceamento dos vértices em uma árvore AVL é feita sempre que se insere ou remove um vértice na árvore.
- Nesse sentido, há a necessidade de se transformar a árvore de modo que o percurso (busca) in ordem não seja alterado e que a árvore passe a ser classificada como balanceada.
- Essas transformações são denominadas rotações.

centro universitário FEI

Árvores Binárias de Busca

Balanceamento - Rotação para a esquerda

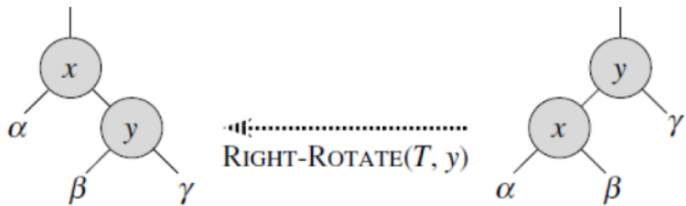



```
1  void RotacaoEsquerda(Arvore *arvore, Vertice *x) {
2    printf("Esquerda em %d\n", x->valor);
3    Vertice *pai = x->pai;
4    Vertice *y = x->dir;
5    Vertice *b = y->esq;
6    if (pai != NULL) {
7      if (pai->esq == x) {
8          pai->esq = y;
9      } else {
10         pai->dir = y;
11       }
12   } else {
13     arvore->raiz = y;
14   }
15   y->pai = pai;
16   x->pai = y;
17   y->esq = x;
18   x->dir = b;
19   if (b != NULL) {
20     b->pai = x;
21   }
22 }
```

Árvores Binárias de Busca

Balanceamento – Rotação para a direita

- Simétrica à rotação para a esquerda



Árvores Binárias de Busca

Balanceamento - Correção de Balanceamento

- A inserção e a remoção de nós em uma Árvore AVL requerem que os ancestrais do nó inserido/removido tenham seus fatores de balanceamento verificados.
- Para cada operação de inserção ou remoção, verifica-se cada nó ancestral, até a raiz.
- Caso o nó verificado tenha fator de balanceamento < -1 ou > 1, quatro casos (que na verdade são dois casos com simetria) devem ser considerados para efetuar as rotações e garantir o balanceamento.

centro universitário FEI

Árvores Binárias de Busca

Balanceamento - Correção de Balanceamento

- Seja R o nó desbalanceado, E seu filho esquerdo, D seu filho direito e P seu pai (que pode não existir se R for a raiz da árvore):
  - Caso 1: Se o fator de R é $\geqslant 2$ e o fator de D é $\geqslant 0$, então: Promover left-rotate em R;
  - Caso 2: Se o fator de R é $\geqslant 2$ e o fator de D é < 0, então: Promover right-rotate em D e left-rotate em R
  - Caso 3: Se o fator de R é $\leqslant -2$ e o fator de E é $\leqslant 0$, então: Promover right-rotate em R.
  - Caso 4: Se o fator de R é $\leqslant -2$ e o fator de E é > 0, então: Promover left-rotate em E e right-rotate em R.

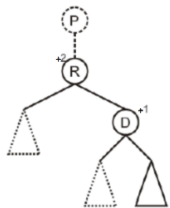
centro universitário FEI

Árvores Binárias de Busca

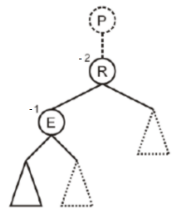
Balanceamento – Correção de Balanceamento

- Caso 1: Se o fator de R é $\geqslant 2$ e o fator de D é $\geqslant 0$, então: Promover left-rotate em R;
- Caso 3: Se o fator de R é $\leqslant -2$ e o fator de E é $\leqslant 0$, então: Promover right-rotate em R.

caso 1



caso 3

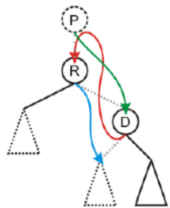


Árvores Binárias de Busca

Balanceamento – Correção de Balanceamento

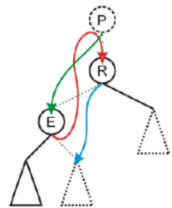
- Caso 1: Se o fator de R é $\geqslant 2$ e o fator de D é $\geqslant 0$, então: Promover left-rotate em R;
- Caso 3: Se o fator de R é $\leqslant -2$ e o fator de E é $\leqslant 0$, então: Promover right-rotate em R.

caso 1



left-rotate em **R**

caso 3

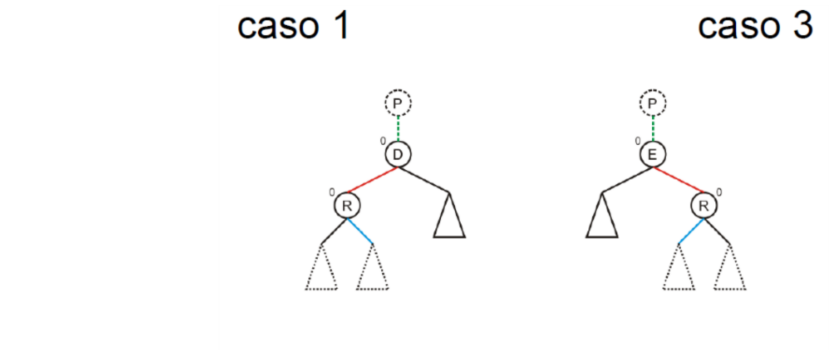


right-rotate em **R**

Árvores Binárias de Busca

Balanceamento - Correção de Balanceamento

- Caso 1: Se o fator de R é $\geqslant 2$ e o fator de D é $\geqslant 0$, então: Promover left-rotate em R;
- Caso 3: Se o fator de R é $\leqslant -2$ e o fator de E é $\leqslant 0$, então: Promover right-rotate em R.

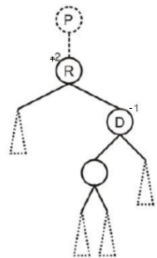


Árvores Binárias de Busca

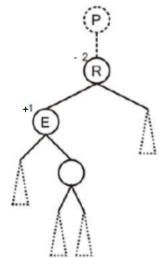
Balanceamento – Correção de Balanceamento

- Caso 2: Se o fator de R é $\geqslant 2$ e o fator de D é < 0, então: Promover right-rotate em D e left-rotate em R
- Caso 4: Se o fator de R é $\leqslant -2$ e o fator de E é > 0, então: Promover left-rotate em E e right-rotate em R.

caso 2



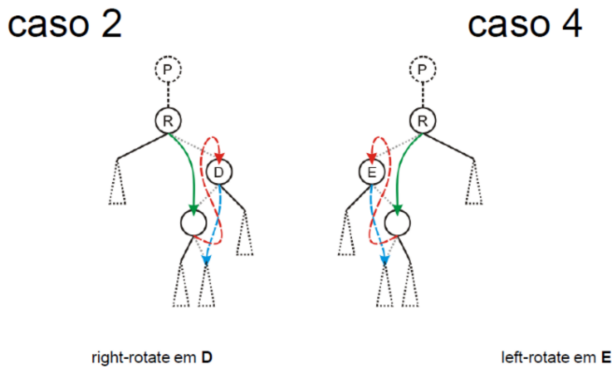
caso 4



Árvores Binárias de Busca

Balanceamento – Correção de Balanceamento

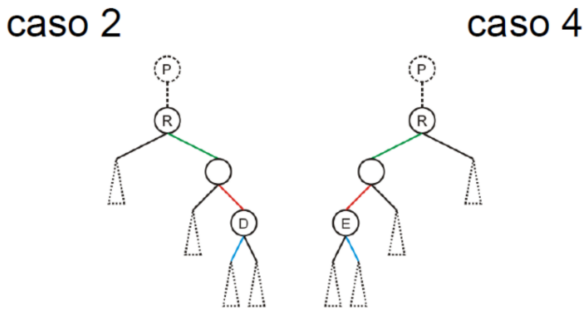
- Caso 2: Se o fator de R é $\geqslant 2$ e o fator de D é < 0, então: Promover right-rotate em D e left-rotate em R
- Caso 4: Se o fator de R é $\leqslant -2$ e o fator de E é > 0, então: Promover left-rotate em E e right-rotate em R.



Árvores Binárias de Busca

Balanceamento – Correção de Balanceamento

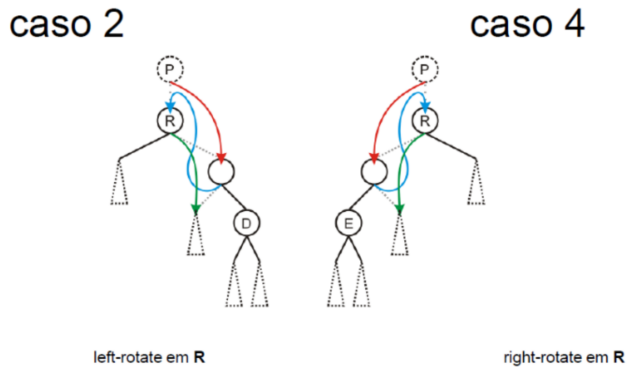
- Caso 2: Se o fator de R é $\geqslant 2$ e o fator de D é < 0, então: Promover right-rotate em D e left-rotate em R
- Caso 4: Se o fator de R é $\leqslant -2$ e o fator de E é > 0, então: Promover left-rotate em E e right-rotate em R.



Árvores Binárias de Busca

Balanceamento – Correção de Balanceamento

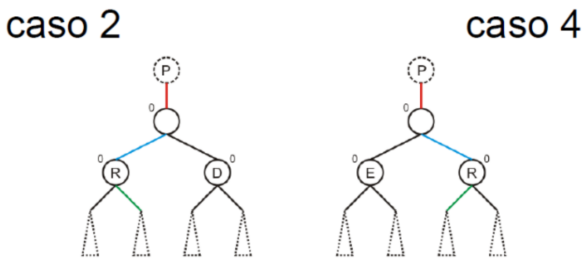
- Caso 2: Se o fator de R é $\geqslant 2$ e o fator de D é < 0, então: Promover right-rotate em D e left-rotate em R
- Caso 4: Se o fator de R é $\leqslant -2$ e o fator de E é > 0, então: Promover left-rotate em E e right-rotate em R.



Árvores Binárias de Busca

Balanceamento – Correção de Balanceamento

- Caso 2: Se o fator de R é $\geqslant 2$ e o fator de D é < 0, então: Promover right-rotate em D e left-rotate em R
- Caso 4: Se o fator de R é $\leqslant -2$ e o fator de E é > 0, então: Promover left-rotate em E e right-rotate em R.



Estrutura de Dados (CC4652)

Aula 11 – Balanceamento

Prof. Luciano Rossi

Ciência da Computação
Centro Universitário FEI

2º Semestre de 2023