Universidade Estadual do Paraná

Rafael Francisco Ferreira

Pesquisa: Árvore Balanceada

Apucarana

2016

O que é?

Árvore Balanceada (ou AVL), em Ciência da Computação, é uma árvore de busca binária autobalanceada. Em tal árvore, as alturas das duas sub-árvores a partir de cada nó diferem no máximo em uma unidade. As operações de busca, inserção e remoção de elementos possuem complexidade O(log n) (no qual n é o número de elementos da árvore). Inserções e remoções podem também requerer o rebalanceamento da árvore, exigindo uma ou mais rotações.

O nome AVL vem de seus criadores soviéticos Adelson Velsky e Landis, e sua primeira referência encontra-se no documento "Algoritmos para organização da informação" de 1962.

Estrutura e Propriedades

A árvore é na verdade um conjunto de nós interligados. Cada nó pode ter até duas ligações: uma com seu filho da esquerda e outra com seu filho da direita.

Cada nó contém as seguintes informações:

Altura **h**: Número inteiro que determina a altura do nó. O número zero representa a raiz da árvore.

Valor **v**: Valor da chave do nó

Filho da esquerda: Pode existir ou não. Caso exista o valor da chave do filho da esquerda deve obrigatoriamente ser menor que o valor da chave do pai.

Filho da direita: Pode existir ou não. Caso exista o valor da chave do filho da direita deve obrigatoriamente ser maior que o valor da chave do pai.

Balanceamento

Uma árvore AVL é dita balanceada quando, para cada nó da árvore, a diferença entre as alturas das suas sub-árvores (direita e esquerda) não é maior do que um. Caso a árvore não esteja balanceada é necessário seu balanceamento através da rotação simples ou rotação dupla. O balanceamento é requerido para as operações de inserção e remoção de elementos. Para definir o balanceamento é utilizado um fator específico para nós.

O fator de balanceamento de um nó é dado pelo seu peso em relação a sua subárvore. Um nó com fator balanceado pode conter 1, 0, ou -1 em seu fator. Um nó com fator de balanceamento diferente dos citados é considerado uma árvore não AVL e requer um balanceamento por rotação ou dupla-rotação. Para garantirmos essa propriedade a cada inserção ou remoção a diferença de altura dos nós afetados e dos nós superiores deve ser recalculada de modo que a altura do nó que está sendo recalculado seja igual a altura do nó esquerdo menos a altura do nó direito.

Uma árvore AVL sempre terá um tamanho menor que:

$$\log_\varphi(\sqrt{5}(n+2)) - 2 = \frac{\log_2(\sqrt{5}(n+2))}{\log_2(\varphi)} - 2 = \log_\varphi(2) \cdot \log_2(\sqrt{5}(n+2)) - 2 \approx 1.44 \log_2(n+2) - 0.328$$

Onde n é o número de elementos da árvore e φ é a proporção áurea.

Complexidade

A árvore AVL tem complexidade O(log n) para todas operações e ocupa espaço n, onde n é o numero de nós pertencentes à arvore.

	Média	**Pior Caso**
Espaço	O(n)	O(n)
Busca	O(log n)	O(log n)
Inserção	O(log n)	O(log n)
Deleção	O(log n)	O(log n)

Complexidade da árvore AVL em notação O.

Operações

1. Busca

Para buscarmos um nó basta comparar o seu valor com o valor do nó que está sendo analisado (raiz no caso da primeira iteração). Se o valor buscado for menor que o valor do nó que está sendo analisado devemos efetuar a busca no filho da esquerda do nó atual, caso contrário deveremos efetuar a busca no filho da direita do nó atual. Se o nó não for encontrado com essa busca podemos ter certeza que ele não existe dentro da árvore.

2. Inserção

Para inserirmos um novo nó de valor K em uma árvore AVL é necessária uma busca por K nesta mesma árvore. Após a busca o local correto para a inserção do nó K será encontrado. Depois de inserido o nó, a altura do nó pai e de todos os nós acima deve ser atualizada. Em seguida o algoritmo de rotação deve ser acionado para o nó pai e depois para todos os nós superiores caso um desses nós caia em uma das condições de rotação.

3. Remoção

Para removermos um nó de valor K na árvore, devemos buscar K nesta árvore e, caso K seja folha da árvore, apenas deletá-lo. Caso K pertença à árvore, mas não seja uma folha da árvore devemos substituir o valor de K com o valor mais próximo possível menor ou igual a K pertencente à árvore.

Para encontrar este valor basta percorrer a subárvore da direita do filho da esquerda de K, até encontrarmos o maior valor M desta subárvore. O valor de K será substituído por M, K será deletado da árvore e caso M tenha um filho à esquerda esse filho ocupará sua antiga posição na árvore.

4. Rotação

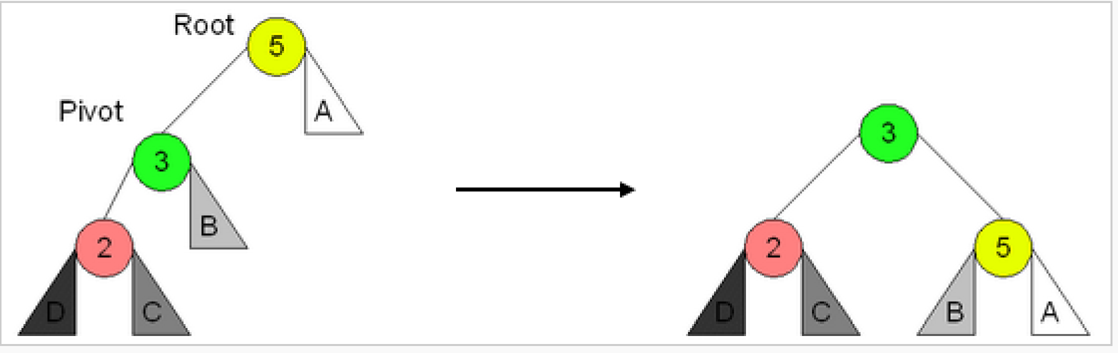
A operação básica em uma árvore AVL geralmente envolve os mesmos algoritmos de uma árvore de busca binária desbalanceada. A rotação na árvore AVL ocorre devido ao seu desbalanceamento, uma rotação simples ocorre quando um nó está desbalanceado e seu filho estiver no mesmo sentido da inclinação, formando 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".

Para garantirmos as propriedades da árvore AVL rotações devem ser feitas conforme necessário após operações de remoção ou inserção. Seja P o nó pai, FE o filho da esquerda de P e FD o filho da direita de P podemos definir 4 tipos diferentes de rotação:

5. Rotação à direita

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 FE é igual a 1. O nó FE deve tornar o novo pai e o nó P deve se tornar o filho da direita de FE. Segue pseudocódigo:

- 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

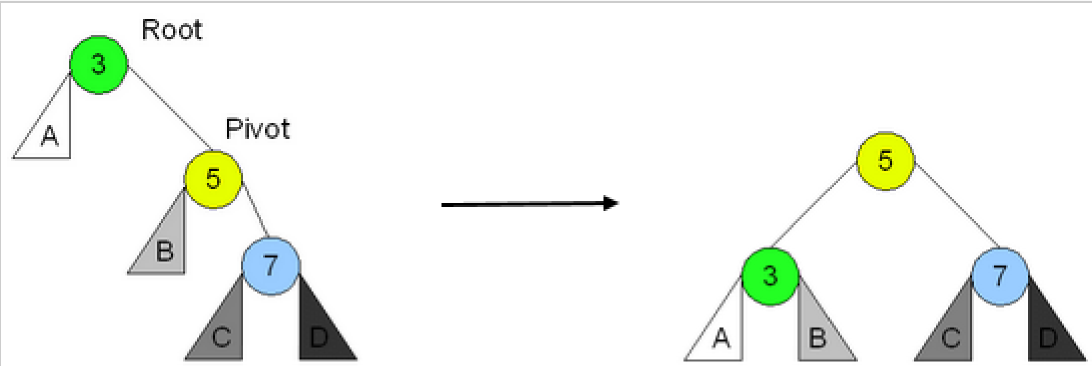


A árvore da esquerda ficou desbalanceada após a inserção do nó 2, necessitando assim de uma rotação a direita. Após a rotação o nó 3 virou o pai dos nós 2 e 5.

6. Rotação à esquerda

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** deve tornar o novo pai e o nó **P** deve se tornar o filho da esquerda de **FD**. Segue pseudocódigo:

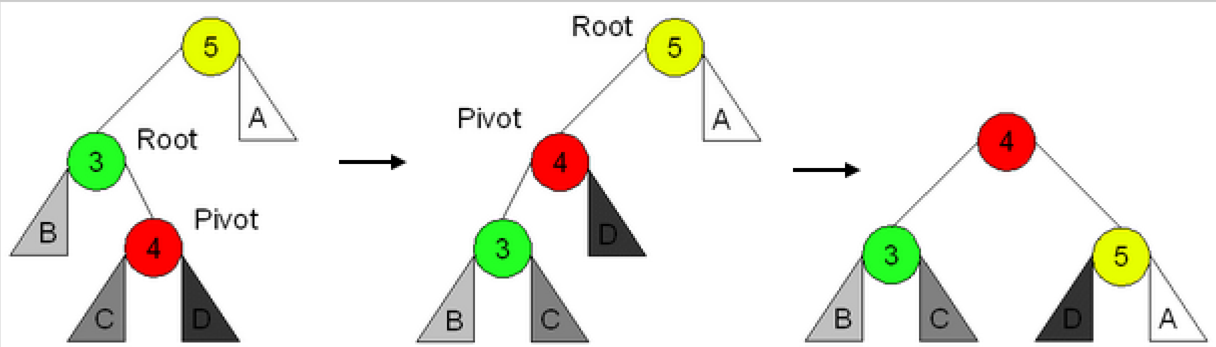
- Seja Y o filho à direita de X
- Torne o filho à esquerda de Y o filho à direita de X.
- Torne X filho à esquerda de Y



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.

7. Rotação dupla à direita

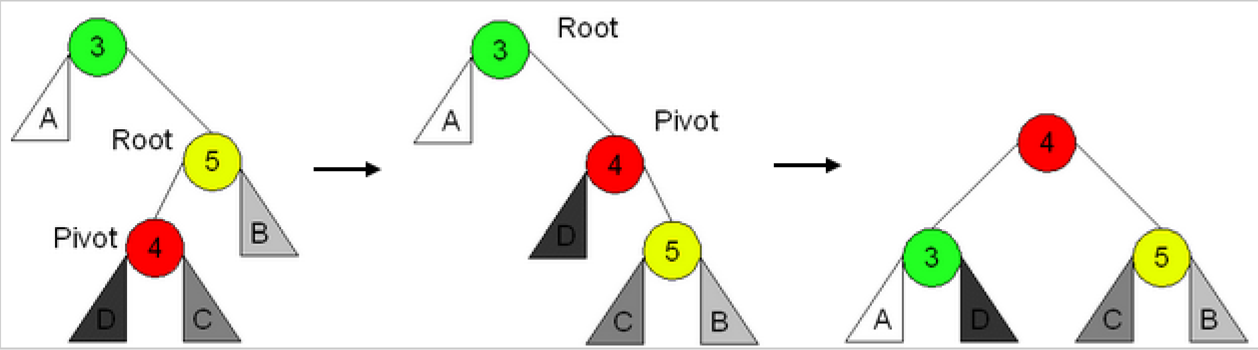
Deve ser efetuada quando a diferença das alturas **h** dos filhos de **P** é igual a **2** e a diferença das alturas **h** dos filhos de **FE** é igual a -**1**. Nesse caso devemos aplicar uma rotação à esquerda no nó **FE** e, em seguida, uma rotação à direita no nó **P**.



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).

8. Rotação dupla à esquerda

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**. Nesse caso devemos aplicar uma rotação à direita no nó **FD** e, em seguida, 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).

É interessante observar que as rotações duplas nada mais são que duas rotações simples seguidas, independentes se à direita ou à esquerda.