Tópico 10: Árvores AVL

Prof. Dr. Juliano Henrique Foleis

Estude com atenção os vídeos e as leituras sugeridas abaixo.

Vídeos

Árvores de Busca Binária: Árvores AVL

Estudo do Código

Uma implementação da inserção em uma Árvore AVL, construída da forma descrita no vídeo está disponível no github da disciplina, na pasta *bin_trees/avl*. O código está amplamente comentado nos pontos mais cruciais. Estude o código com cuidado para compreender como as rotações funcionam e como que elas reestabelecem a propriedade AVL na árvore.

O arquivo *bin_trees/avl/main.c* contém vários casos de teste comentados. Descomente um por vez, e insira chamadas a *AAVL_Imprimir* e/ou *AAVL_GenDOT* entre as inserções para ver os rebalanceamentos e como alteram a estrutura da árvore localmente.

Exercícios

1. Implemente a função *int AAVL_Balanceada(AAVL *A)* que retorna 1 se a árvore respeita a propriedade AVL, e 0, caso contrário. Implemente tanto a versão preguiçosa, quanto a ansiosa.

2. No vídeo discuto brevemente como pode ser implementada a remoção de um nó em uma árvore AVL.

A idéia básica é:

- Realizar a remoção do nó utilizando o mesmo algoritmo que foi usado para a remoção em uma ABB;
- Na volta da remoção, atualizar os fatores de balanceamento dos nós no caminho. Em cada nó, verificar se houve violação da propriedade AVL. De acordo com o caso (E, D, ED, DE), aplicar as rotações necessárias.

É importante lembrar que, diferentemente da inserção, é possível que seja necessárias vários rebalanceamentos em uma só remoção.

De acordo com esse algoritmo, implemente a função *void AAVL_Remover(AAVL **A, int chave)*. Caso seja necessário, use essa função como *wrapper* para a função recursiva subjacente.

Leitura Sugerida

HALIM, S; *et. al. AVL Tree* (Link)

JANKUN-KELLY, T. J. *AVL Trees: Understanding Rotations* (Link)

RAJINIKANTH, B. *AVL Tree :: Data Structures* (Link)

BONS ESTUDOS!