Atividade Prática 09

Aluno: Raphael Henrique Braga Leivas

Matrícula: 2020028101

1 OBJETIVO

O objetivo da prática é visualizar os efeitos que diferentes critérios de balanceamento tem em uma árvore binária de pesquisa, em particular nas operações de inserção e remoção, que exigem rotações nas árvores de modo a mantê-las balanceadas.

2 EXECUÇÃO

O critério de balanceamento usado será um dos critérios sugeridos como exemplo no enunciado da prática: a diferença entre as alturas das sub-árvores direta e esquerda é no máximo 2.

Com esse critério, acredito que a árvore não ficará desbalanceada, independente de quantas inserções forem feitas, uma vez que é feita uma rotação toda vez que altura de uma das sub-árvores de um nó exceder a altura da outra em mais de 2. Note que, como elementos maiores que o nó atual são inseridos à direita dele, e elementos menores são inseridos à esquerda (assumindo que nunca será inserido um elemento que já existe, isto é, todos os nós são diferentes), temos que rotações sempre ocorrerão de modo a manter a regra das alturas das sub-árvores ser sempre menor que 2.

2.1 MODIFICAÇÕES NO CÓDIGO

Esse critério foi escolhido de modo a minimizar as alterações que devem ser feitas no código fonte disponibilizado no moodle. Nesse sentido, podemos reutilizar a função BF já implementada no código fonte do AVL fornecido no moodle, de modo que basta trocar as condicionais feitas de BF(node) == -2 ou +2 nas funções insert e Delete para BF(node) == +3 ou -3. Isso é feito uma vez que um fator de balanceamento igual a +- 3 fere o nosso novo critério estabelecido, exigindo assim uma rotação para manter o balanceamento da árvore.

Assim, as modificações feitas foram

- função insert: muda os dois ifs de BF(T) == 2 para BF(T) == 3 (um fica +3 e outro -3)
- função Delete: muda os dois ifs de BF(T) == 2 para BF(T) == 3 (um fica +3 e outro -3)

Para facilitar as modificações do valor do critério, foi criado a variável *balancingCriterion* , que recebe um inteiro. No código original, temos que ela tinha o valor 2, com a modificação ela passa a ter valor 3, como mostrado abaixo

```
if (BF(T) == balancingCriterion) {

}
```

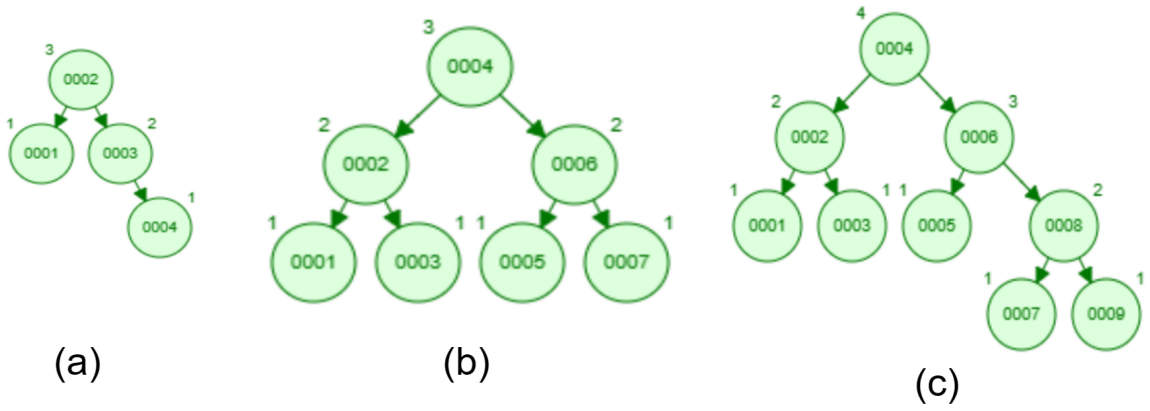
Assim, basta alterar o valor dessa variável que automaticamente é alterado nos ifs necessários, facilitando o estudo do código.

2.2 EFEITOS NA INSERÇÃO

O primeiro passo é entender o comportamento de uma árvore AVL, que usa critério de balanceamento a diferença entre as alturas das sub-árvores direta e esquerda ser no máximo 2. A Figura 2.1 exibe as animações de uma árvore AVL ao se inserir a seguinte sequência A de 10 números:

$$A = [1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10]$$

Figura 2.1. Balançeamento de árvore AVL ao inserir a sequência A após inserção do (a) quarto elemento, (b) sétimo elemento e (c) último elemento.

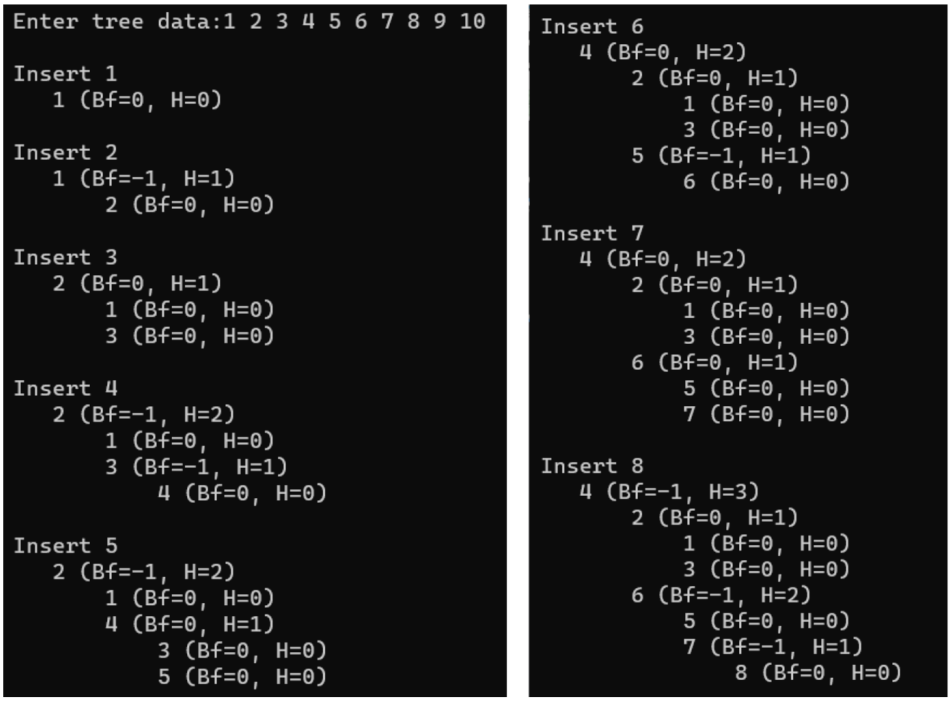


Fonte: Disponível em: <https://www.cs.usfca.edu/~galles/visualization/AVLtree.html>. Acesso em 15 de jun. de 2023.

Note, na Figura 2.1, que a raiz da árvore mudou de 1 para 4 entre as Figuras (a) e (b), ou seja, houve uma rotação na árvore de modo a manter o balanceamento. Além disso, note que a sequência de entradas A irá exigir constantes rotações na árvore de modo a manter o balanceamento, uma vez que ela é crescente e por consequência há uma tendência da árvore ficar adicionando só elementos para o lado direito.

Agora vamos ver como o código original do moodle responde para essa mesma sequência. O resultado está exibido na Figura 2.2.

Figura 2.2. Resposta do código com o critério original.



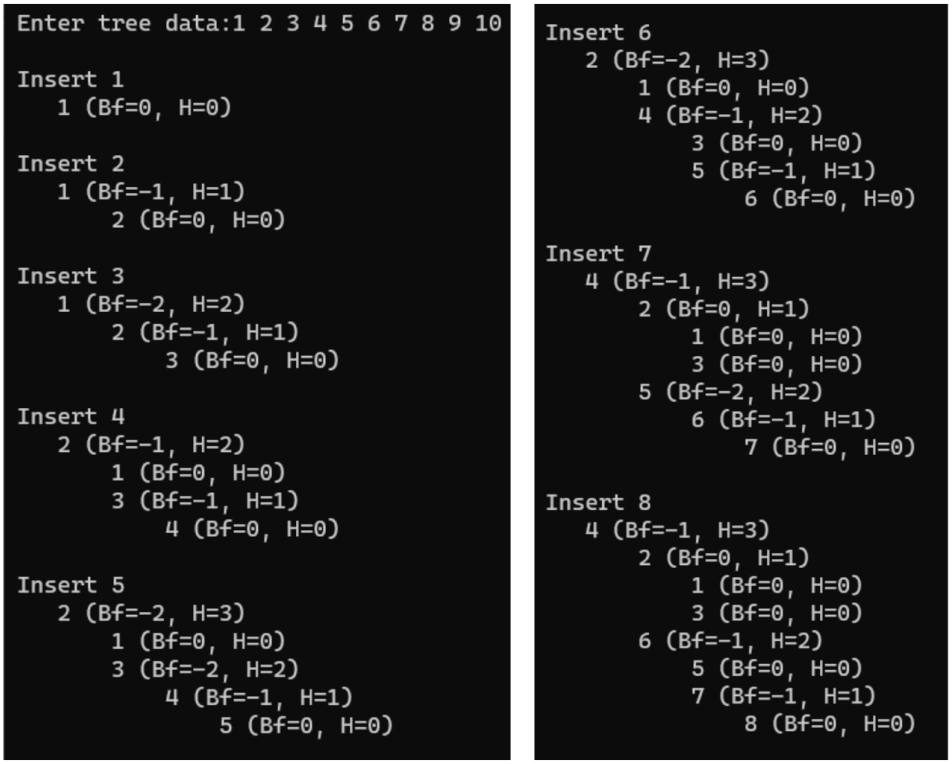
Fonte: elaboração própria.

Assim como a Figura 2.1, é possível perceber as rotações na árvore entre a quarta e sexta inserção, no qual temos alterações na raiz da árvore.

Agora vamos modificar a variável balancingCriterion de 2 para 3. O resultado obtido está exibido na Figura 2.3.

Comparando as Figuras 2.2 e 2.3, percebemos que as rotações demoram mais acontecer: enquanto na 2.2 elas ocorrem entre as inserções 2-3 e 5-6, na 2.3 ocorrem entre 4-5 e 6-7. Isso é conforme o esperado, pois o novo critério permite que a diferença de tamanho entre as árvores direita e esquerda seja maior que o critério original. Além disso, vemos na Figura 2.3 que o novo critério permite árvores mais “longas” para a direita, uma vez que a rotação demora mais para acontecer.

Figura 2.3. Resultado com o novo critério de balanceamento.



Fonte: elaboração própria.

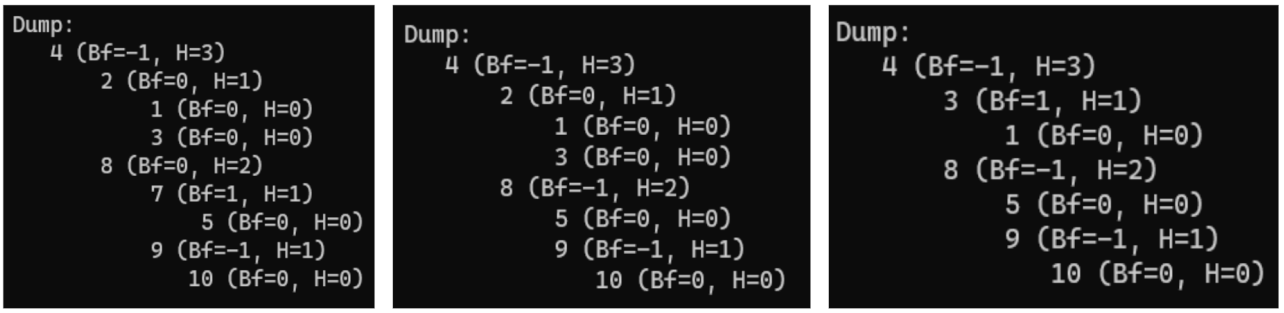
2.3 EFEITOS NA REMOÇÃO

Para averiguar os efeitos da mudança do critério de balanceamento nas árvores, vamos remover os seguintes itens das árvores

$$B = [6,\ 7,\ 2]$$

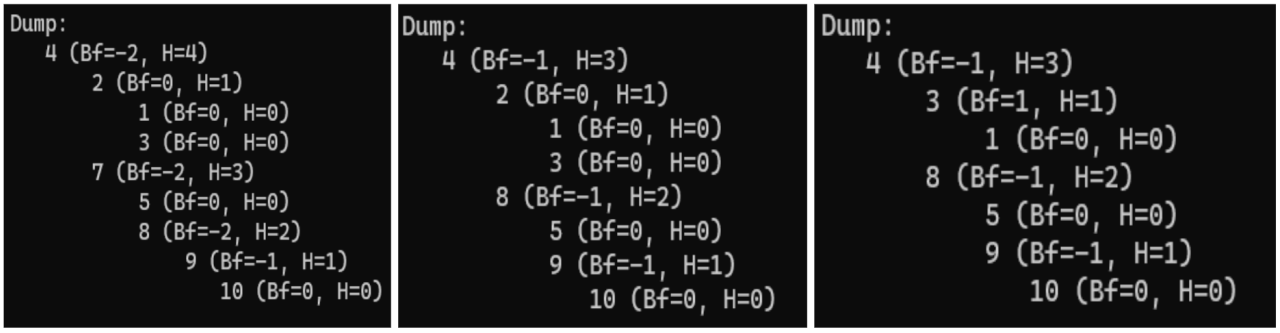
A Figura 2.4 mostra os efeitos da remoção na árvore com o critério original, ao passo que a Figura 2.5 mostra os efeitos com o novo critério.

Figura 2.4. Remoções da esquerda para a direita de 6, 7 e 2 com o critério original.



Fonte: elaboração própria.

Figura 2.5. Remoções da esquerda para a direita de 6, 7 e 2 com o novo critério.



Fonte: elaboração própria.

Comparando as Figuras 2.4 e 2.5, observamos que a remoção dos elementos provocou rotações semelhantes nas árvores. Nesse sentido, vemos que os efeitos na interseção foram mais acentuados que os efeitos na remoção. Como esperado, é possível perceber que as árvores na Figura 2.5 são mais “longas” que as da Figura 2.4, novamente devido ao novo critério de balanceamento adotado.

3 CONCLUSÃO

Tendo em vista o objetivo da prática, foi possível comparar o comportamento de árvores binárias de pesquisa com diferentes critérios de balanceamento, bem como perceber as diferenças nas rotações nas operações de inserção e remoção.