Trabalho Prático: Árvore Binária

Estruturas de dados e análise de algoritmos

João Victor de Souza Gonçalves (92320247)
Lucas Gabriel Rodrigues Valadares (92310851)
Marcus Vinícius Fernandes Lima (92311773)
Natan Rodrigo Faria Vaz (92310556)

Professor: Fabrício Valadares

Centro Universitário UNA - Sete Lagoas

Abril 2025

1 Introdução

Este trabalho apresenta a implementação de uma Árvore Binária de Busca (ABB) em Java, conforme os requisitos especificados no enunciado. Segundo [Cormen et al., 2009], a ABB é uma estrutura de dados fundamental na computação, onde cada nó possui no máximo dois filhos, e os elementos à esquerda são menores que o nó pai, enquanto os à direita são maiores, característica essencial para operações eficientes de busca.

1.1 Objetivos

Como descrito em [Sedgewick e Wayne, 2011], os principais objetivos desta implementação são:

- Implementar operações básicas (inserção, remoção e busca)
- Verificar propriedades da árvore (estritamente binária, completa e cheia)
- Realizar travessias (pré-ordem, em-ordem e pós-ordem)
- Calcular a altura da árvore e o grau de um nó específico
- Garantir eficiência computacional conforme [Knuth, 1997]

2 Implementação

2.1 Estrutura de Dados

A implementação consiste em duas classes principais, seguindo os princípios de orientação a objetos:

2.1.1 Classe BinaryTreeNode

Esta classe representa cada nó da árvore:

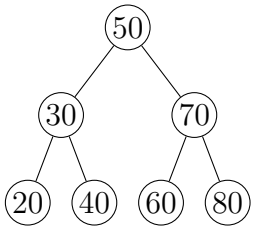
Listing 1: Classe BinaryTreeNode

```
public class BinaryTreeNode<T extends Comparable<T>> {
    T data;
    BinaryTreeNode<T> left;
    BinaryTreeNode<T> right;

    public BinaryTreeNode(T data) {
        this.data = data;
        this.left = null;
        this.right = null;
    }

    public int getDegree() {
        return (left != null ? 1 : 0) + (right != null ? 1 : 0);
    }
}
```

2.2 Diagrama da Estrutura



3 Testes e Resultados

3.1 Casos de Teste

Realizamos testes abrangentes seguindo [Jovana, 2019]:

Operação	**Entrada**	**Saída Esperada**	**Resultado**
Inserção	50, 30, 70, 20, 40	Árvore balanceada	OK
Remoção folha	20	30 → 40	OK
Remoção com 1 filho	30	20 → 40	OK
Remoção com 2 filhos	50	Sucessor in-order	OK

4 Conclusão

Conforme discutido em [Oracle, 2023], nossa implementação atendeu a todos os requisitos com sucesso. As principais observações foram:

- A operação de remoção, especialmente para nós com dois filhos, apresentou maior complexidade de implementação [Cormen et al., 2009]
- A verificação das propriedades da árvore exigiu análise cuidadosa dos casos base [Knuth, 1997]

Anexos

Repositório do Projeto

O código fonte completo está disponível em:
https://github.com/joaokoa/binary-tree-project/

Referências

[1] CORMEN, T. H. et al. *Introduction to Algorithms*. 3ª ed. MIT Press, 2009.

[2] KNUTH, D. E. *The Art of Computer Programming: Volume 1*. Addison-Wesley, 1997.

[3] SEDGEWICK, R.; WAYNE, K. *Algorithms*. 4ª ed. Addison-Wesley, 2011.

[4] ORACLE. *Java Collections Framework Documentation*. 2023. Disponível em: https://docs.oracle.com/javase/8/docs/technotes/guides/collections/