Petrópolis-RJ, 17 de dezembro de 2024

Dicionário AVL

Augusto Freitas Franco Gomes
Felipe Hiroshi Correia Katsumoto

1

AEDs-II
Engenharia de Computação - CEFET-RJ

Resumo. *O Dicionário AVL implementa uma árvore AVL utilizada em forma de dicionário. Ele permite criar uma árvore vazia, inserir, buscar, remover palavras e imprimir a árvore, mantendo-a balanceada conforme as regras de balanceamento da AVL.*

Palavras-Chave. *Dicionário, Árvore, Inserir, Buscar, Remover, Imprimir.*

Abstract. *The AVL Dictionary implements an AVL tree used as a dictionary. It allows creating an empty tree, inserting, searching, deleting words, and printing the tree while keeping it balanced according to AVL balancing rules.*

Keywords. *Dictionary, AVL, Tree, Inserting, Searching, Deleting, Printing.*

Petrópolis-RJ, 17 de dezembro de 2024

1. Introdução

Árvore AVL é uma árvore binária de busca balanceada. Árvores balanceadas são aquelas que minimizam o número de comparações efetuadas no pior caso para uma busca com chaves de probabilidades de ocorrência idênticas. Contudo, para garantir essa propriedade em aplicações dinâmicas, é necessário reconstruir a árvore para seu estado ideal a cada operação sobre seus nós (inclusão ou exclusão), por meio de operações de troca de ponteiros, conhecidas como rotações (RR, RL, LL, LR), a fim de alcançar um custo de algoritmo com o tempo de pesquisa tendendo a O(log n).

Este trabalho apresenta um programa que implementa um dicionário utilizando uma árvore AVL. O dicionário permite operações comuns, como criação de uma árvore vazia, remoção de palavras, inserção de palavras e seus significados, busca por palavras e impressão da estrutura atual da árvore.

O programa foi desenvolvido considerando uma interface interativa que permite ao usuário inserir palavras e seus significados, pesquisar termos cadastrados, remover entradas e visualizar a árvore resultante. A cada operação de inserção e remoção, o programa ajusta automaticamente a árvore para manter seu balanceamento.

2. Código

2.1. TADs

A estrutura ”No”representa um nó da árvore AVL. Cada nó contém:

A palavra armazenada no nó.

significado: O significado associado à palavra.

fatorB: O fator de balanceamento do nó.

pai: Ponteiro para o nó pai.

esq: Ponteiro para o nó filho à esquerda.

dir: Ponteiro para o nó filho à direita.

Estrutura ”Arvore”contém apenas um ponteiro para o nó raiz da árvore AVL, representada por:

raiz: Ponteiro para o nó raiz da árvore.

2.2. Funções

As funções do código são:

2.2.1. Balanceamento

balancear: Verifica o fator de balanceamento do nó e aplica a rotação necessária (LL, RR, RL ou LR) para manter a árvore balanceada.

calcular altura: Calcula a altura de um nó, ou seja, o maior número de arestas no caminho entre o nó e uma folha da árvore.

atualizar FB: Atualiza o fator de balanceamento de um nó com base na altura de seus filhos à esquerda e à direita.

2.2.2. Rotações

LL: Realiza uma rotação simples à esquerda para balancear a árvore. Utilizada quando o fator de balanceamento do nó é positivo.

RR: Realiza uma rotação simples à direita para balancear a árvore. Utilizada quando o fator de balanceamento do nó é negativo.

RL: Realiza uma rotação dupla para balancear a árvore. Composta por uma rotação à direita (RR) seguida por uma rotação à esquerda (LL). Utilizada quando o fator de balanceamento do nó é negativo e o fator de balanceamento do filho à direita é positivo.

LR: Realiza uma rotação dupla para balancear a árvore. Composta por uma rotação à esquerda (LL) seguida por uma rotação à direita (RR). Utilizada quando o fator de balanceamento do nó é positivo e o fator de balanceamento do filho à esquerda é negativo.

2.2.3. Manipulações da árvore

inserir: Insere uma nova palavra no dicionário. Caso a palavra já exista, a inserção não é realizada. Após a inserção, a árvore é balanceada.

remover: Remove uma palavra do dicionário. Caso a palavra tenha dois filhos, o nó antecessor é movido para o lugar do nó removido (nó antecessor é encontrado pela função ”encontrarAntecessor”). Após a remoção, a árvore é balanceada.

busca: Realiza uma busca na árvore para encontrar uma palavra. A função imprime a palavra encontrada e sua altura na árvore.

percurso em ordem: Realiza um percurso em ordem na árvore e imprime as palavras e suas respectivas alturas.

2.2.4. Main

O programa oferece um menu para o usuário realizar as operações de manipulação da árvore AVL:

[1] Criar árvore vazia: Inicializa uma nova árvore vazia.

[2] Remover palavra: Remove uma palavra do dicionário.

[3] Inserir palavra: Insere uma nova palavra no dicionário.

[4] Buscar palavra: Busca uma palavra no dicionário e exibe seu significado.

[5] Imprimir árvore: Imprime as palavras armazenadas na árvore em ordem crescente.

[6] Encerrar: Finaliza a execução do programa.

Petrópolis-RJ, 17 de dezembro de 2024

3. Testes

Os testes no programa foram realizados de duas maneiras:

Teste manual: o próprio usuário executava o código e selecionava as opções disponibilizadas (1 para criar a árvore, 2 para remover uma palavra, 3 para inserir uma palavra, 4 para buscar uma palavra, 5 para imprimir a árvore e 6 para encerrar o programa), implementando a árvore conforme seu critério.

Teste com arquivos: o programa lê um arquivo com números de 1 a 6, representando os comandos do código, e palavras diversas, implementando a árvore conforme a ordem dos comandos e das palavras dispostas no arquivo de texto.

4. Resultados

Os resultados foram coletados em termos de tempo de execução para inserção e balanceamento, com base em diferentes quantidades de palavras e ordens de inserção (ordenada e aleatória). Os resultados obtidos foram:

10 palavras - 0,002 segundos: Com um número reduzido de palavras, o tempo de execução é quase desprezível. A árvore AVL é balanceada rapidamente, sem acarretar grandes custos computacionais.

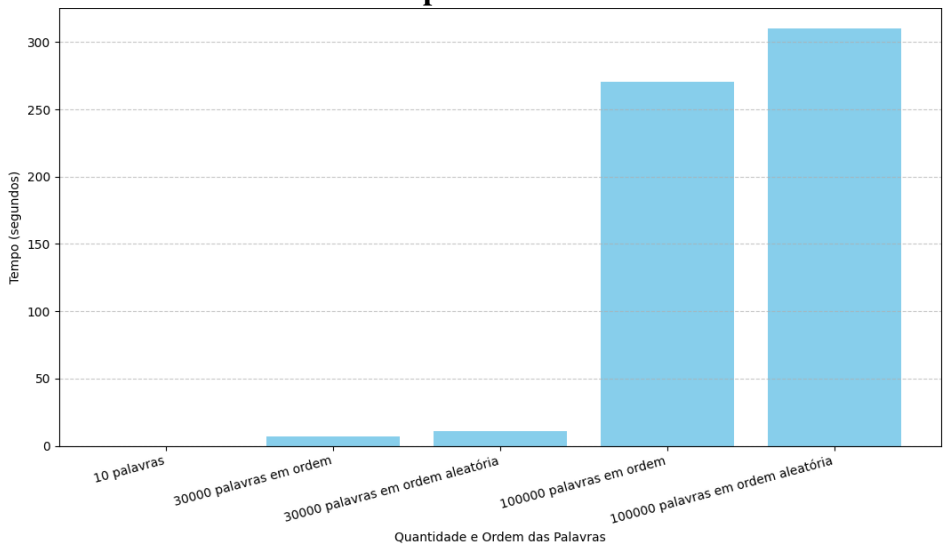
30.000 palavras em ordem - 6,77 segundos: O tempo de execução é significativamente maior devido ao aumento no número de inserções e balanceamentos necessários.

30.000 palavras em ordem aleatória - 10,685 segundos: A inserção de palavras em ordem aleatória aumenta o tempo de execução, pois a árvore precisa realizar mais rotações para se manter balanceada.

100.000 palavras em ordem - 4 minutos e 30,861 segundos: Com um volume ainda maior de palavras, o tempo de execução se torna mais longo.

100.000 palavras em ordem aleatória - 5 minutos e 9,881 segundos: O tempo de execução para a inserção de 100.000 palavras aleatórias é ainda maior, evidenciando o impacto negativo da aleatoriedade no desempenho da árvore.

Gráfico do crescimento do tempo de processamento por quantidade e ordem das palavras



Petrópolis-RJ, 17 de dezembro de 2024

5. Conclusão

O desenvolvimento do programa que implementa um dicionário utilizando uma árvore AVL demonstrou a eficiência desse tipo de estrutura para armazenar e gerenciar palavras e significados de maneira balanceada. Os testes realizados confirmaram a consistência do sistema, com tempo de execução moderado mesmo para grandes volumes de dados. A aplicação mostra-se adequada para sistemas que demandam buscas rápidas e consistentes, destacando a importância de estruturas de dados bem projetadas.

Referências

Assis, L. (2024a). Árvores avl (inserção). [apresentação de slides]. cefet/rj uned petrópolis.

Assis, L. (2024b). Árvores avl (remoção). [apresentação de slides]. cefet/rj uned petrópolis.

H. Cormen, T. (1989). et al. algoritmos: Teoria e prática.

Luiz Szwarcfiter, Jayme, e. L. M. (1994). *Estruturas de Dados e Seus Algoritmos*.

[Assis 2024a] [Assis 2024b][Luiz Szwarcfiter 1994], and [H. Cormen 1989].