UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE

INSTITUTO METRÓPOLE DIGITAL (IMD)

BACHARELADO EM TECNOLOGIA DA INFORMAÇÃO

ANCHEL VITOR VARELA DA SILVA

DANRLEY ARAUJO DE LIMA

Relatório - Árvore Binária de Busca

NATAL – RN

2022

1. Resumo

Este trabalho tem como objetivo a implementação de uma árvore binária de busca, sem uso de estruturas de dados como vetores e filas, buscando a maior eficiência possível.

2. Métodos e suas complexidades

2.1. busca, inserção e remoção

A complexidade da inserção e remoção é dominada pela complexidade da busca, que no pior caso é O(n). Porém, o caso médio é O(log n), pois não é necessário percorrer toda a árvore para buscar, inserir ou remover um elemento.

2.2. enesimoElemento(int n)

Esta função que retorna o enésimo elemento de uma ABB. Seu pior caso é O(n), quando a árvore tem um formato linear como um vetor. Porém, quando não é linear, temos como complexidade O(log n), pois não é necessário percorrer toda a árvore para retornar o elemento na posição “n” ao percorrer em ordem.

A cada nó visitado é calculado a quantidade de nós à esquerda do nó visitado. Caso a quantidade + 1 seja igual à posição n, retornamos o valor do nó. Caso a quantidade seja maior, chamamos o filho à esquerda do nó. Caso a quantidade seja menor, chamamos o filho à direita e retiramos a quantidade +1, deixando apenas quantos elementos faltam para chegar à posição desejada.

2.3. posicao(int x)

Esta função retorna a posição de um elemento a partir de um valor recebido por parâmetro. Esta função tem como pior caso O(n) caso a árvore tenha a forma linear. Quando não, sua complexidade é O(log n), pois não é necessário percorrer toda a árvore para retornar a posição do elemento que tem valor “x”.

Seu funcionamento é similar à função enesimoElemento. A cada iteração é calculado a quantidade de nós à esquerda e sua posição é dada por esse valor + 1. Caso o valor do nó seja igual ao valor x, retorna a posição que o elemento se encontra. Caso o valor seja maior que x, chamamos o filho à esquerda e decrementa a posição. Caso o valor seja menor que x, chamamos o filho à direita.

2.4. mediana()

Esta função retorna o elemento da mediana da ABB. Sua complexidade é dominada pela chamada à função enesimoElemento, passando por parâmetro o índice

da mediana e retornando o elemento da mediana. Caso seja uma árvore linear, sua complexidade é O(n/2) = O(n). Caso não seja uma árvore linear, sua complexidade será de O(log n).

Quando a árvore tem um número par de elementos, é pego o menor valor na mediana.

2.5. media(int x)

Retorna a média aritmética dos elementos da árvore que tem um valor “x” como raiz. Primeiro é feito uma busca pelo nó de valor “x”. Esta função tem complexidade O(n), pois é dominada pela complexidade do algoritmo de percurso em pré ordem, que é utilizado para percorrer os elementos e somar seus valores.

2.6. ehCheia()

A função percorre por todos os nós da árvore recursivamente e faz verificações para saber quantos filhos não nulos cada nó tem. Caso algum nó tenha apenas 1 filho não nulo, logo, esta árvore não é cheia. Complexidade O(n) por percorrer todos os nós da árvore.

2.7. ehCompleta()

Função que retorna se a árvore é uma árvore completa. Sua complexidade é dominada pela função de calcular altura, que tem complexidade O(n), pois percorre cada nó filho para calcular a altura da raiz.

Para ser completa, a árvore tem que obedecer a regra:

$$2^{h-1} <= n <= 2^{h} - 1.'$$

Onde:

h é a altura da raiz;

n é a quantidade de nós da árvore.

2.8. preOrdem()

Função para imprimir a árvore em percurso de pré-ordem. Tem complexidade O(n) por percorrer todos os elementos.

2.9. imprimirBarras(No raiz, int tracos, int blank)

Está função imprime no terminal a árvore no modelo “formato 1” indicado nos requisitos do trabalho. Existem 2 loops, um para incrementar uma variável afim de aumentar o número de “-” que irão ser impressos no terminal, o outro loop tem praticamente o mesmo objetivo do primeiro, porém, para aumentar o número de “ “. Esses 2 loops vão controlar a identação e nível dos nós. Na chamada recursiva são

passados parâmetros atualizados que representam o nível da árvore. Complexidade de O(n) pois irá percorrer todos os elementos da árvore.

3.0. imprimirParenteses(No raiz)

Está função imprime no terminal a árvore no modelo “formato 2” indicado nos requisitos do trabalho. São feitas chamadas recursivas desta mesma função para montar a variável “saida”. De acordo com a altura e seus elementos nulos é que vão ser feitas diferentes chamadas para concatenar a saída. Complexidade de O(n) pois irá percorrer todos os elementos da árvore.