Relatório de Implementação de Projeto Prático com uma Árvore com Números Variados de Filhos

Gislainy Crisostomo Velasco

[1]Instituto de Informática – Universidade Federal de Goiás (UFG)
Caixa Postal 131 — 74001-970 — Goiânia — GO -– Brasil

gislainycrisostomo@gmail.com

Abstract. *This extended summary presents a solution to a problem with the URI Online Judge platform. The proposed solution uses the Tree with Varied Numbers of Children. This work discusses the solution adopted, its complexity, as well as the test cases and the results obtained according to the instances.*

Resumo. *Este resumo estendido apresenta solução de um problema da plataforma URI Online Judge. A solução proposta utiliza da Árvore com Números Variados de Filhos. Neste trabalho é discutido a solução adotada, sua complexidade, bem como os casos de testes e os resultados obtidos de acordo com as instâncias.*

1. Introdução

Em Ciência da Computação tradicionalmente são utilizadas abstrações do mundo real para resolver problemas computacionais. Um exemplo é a estrutura de dados árvore que se assemelha a estrutura física de uma árvore encontrada na natureza, pois existe uma relação estrutural entre seus elementos e sua organização. Nesse sentindo, ao longo do tempo foram propostas vários algoritmos de árvores que possibilitaram armazenar e manipular os dados com eficácia e eficiência.

Usualmente são utilizados árvores que possuem como característica um número predefinidos de filhos e também uma relação de organização dos seus dados. Nesse sentido, tem se a Árvore Binária, uma estrutura conhecida e diversos outros algoritmos utiliza seus princípios como base, nela todos os nós a esquerda são menores que a raiz e a direita, maiores [Cormen et al. 2009]. Com isso, caso for necessário a representação de uma árvore da forma que é encontrada na natureza, um galho pode possuir de 0 a N filhos, sem conhecimento do valor máximo de filhos, é necessário a utilização de outra estrutura de dados que permita armazenar um número variados de filhos por nós.

Dado esse cenário, este trabalho tem como propósito a implementação de um algoritmo que tem como principio o armazenamento de número variado de filhos para um determinado nó. O problema proposto foi extraído da plataforma URI Online Judge[1]. O algoritmo e testes realizados foram feitos em um contexto acadêmico, proposto e supervisionado pelos professores da disciplina Análise de Dados e Projeto de Algoritmos, de um Programa de Pós Graduação em Ciência da Computação.

Este artigo está dividido da seguinte forma: a Seção 2 apresenta os Conceitos Básicos; a Seção 3 apresenta a Descrição do Problema bem como suas entradas e saídas;

[1]https://www.urionlinejudge.com.br/judge/pt/problems/view/2575

a Seção 4 discorre sobre a Solução Adotada, bem como fluxo do algoritmo implementado e sua complexidade; a Seção 5 apresenta os Casos de Testes realizados, comportamento do algoritmo de acordo com as entradas e suas possíveis saídas, além de apresentar as Instâncias e Resultados; por fim, a Seção 6 apresenta a Conclusão.

2. Conceitos Básicos

Árvore é uma estrutura de dados extremamente importante em Ciência da Computação, na qual possui uma relação estrutural entre seus dados, denominados de nós, sendo uma relação de hierarquia, onde um conjunto de nós são hierarquicamente subordinados a outro [Monard and Nicoletti 1993]. Formalmente, uma árvore é um conjunto finito de um ou mais nós onde:

1. existe um nó especial denominado raiz da árvore;
2. os demais nós formam n conjuntos disjuntos ($n > 0$), onde cada um destes conjuntos T ($1 < i < n$) é, por sua vez, uma árvore. As árvores T, são chamadas de subárvores da raiz.

Nesse contexto, existem inúmeras representações dessa estrutura, como a Árvore Binária, AVL, Rubro Negro, *Heapsort*. As abordagens citadas possuem em comum que existe um número predefinido de filhos para cada nó [Cormen et al. 2009]. Não sendo possível representar uma estrutura de dados que não se tem conhecimento de quantos filhos um nó pode ter. Nesse sentido, para representar uma árvore com um número variados de filhos, Figura 1, os filhos de um nó são representados por uma lista, onde:

1. um nó aponta para o seu primeiro filho;
2. cada filho aponta para o próximo irmão.

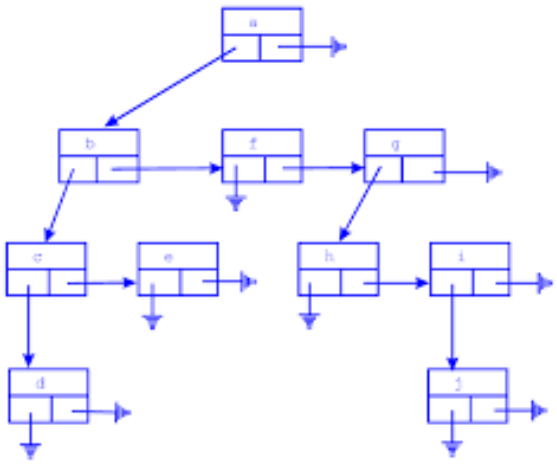

Figura 1. Representação de uma árvore com número variados de filhos

3. Descrição do problema

O problema foi extraído da plataforma URI Online Judge. É apresentado uma história na qual o personagem principal realiza cortes em uma árvore de natal. Para realizar tal operação, é definido o critério de beleza do galho, os valores negativos representam que o galho é feio. A beleza final da árvore é dada pela soma das belezas de seus galhos.

O método de poda tradicional que não utiliza nenhum critério de otimização é apresentado na Figura 2. Quando um galho é removido, todos os galhos filhos são removidos também. Na árvore à direita, os galhos cortados estão pontilhados, enquanto os nós e galhos que caíram após o corte estão tracejados. Neste exemplo, tanto a árvore sem corte, quanto a cortada possuem um valor 10 de beleza.

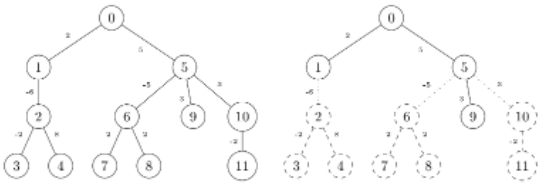

Figura 2. Corte sem critério

O método proposto utilizando critério de otimização, no qual é realizado o menor número de cortes para obter o maior grau de beleza da árvore é apresentado na Figura 3.

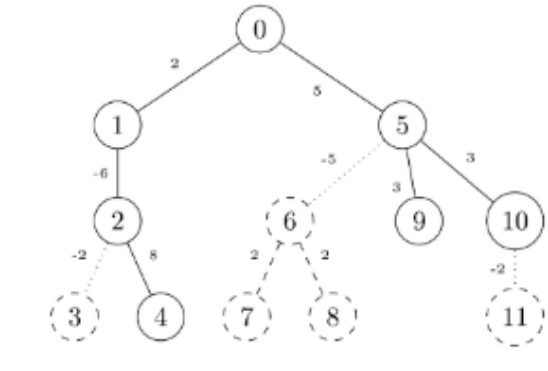

Figura 3. Corte ótimo

3.1. Entrada

A entrada consiste de um inteiro **N** ($2 < \mathbf{N} < 2^6$) que é a quantidade de nós da árvore. Nas próximas **N**-1 linhas, temos quatro inteiros **di** (0 < di < N-2), **ai** (0 < ai < **N**-1), **bi** (0 < **bi** < N-1), e **wi** (-1000 < **wi** < 1000) representando o identificador do galho **i**, que ele conecta o nó **ai** ao nó **bi**, e que ele possui **wi** de beleza pela classificação de Roberto. A árvore é sempre enraizada no nó 0. É garantido que o grafo da entrada é conexo e não possui ciclos.

3.2. Saída

A saída deve conter dois inteiros, D e M que representam o quão bonito é a árvore com o corte ótimo e quantos cortes precisam ser feitos, respectivamente. Se o número de cortes for maior do que 0, na próxima linha, imprima M inteiros **dj** em ordem crescente e separados por espaço, onde **dj** representa o identificador de cada galho **j** a ser cortado.

Caso exista mais de uma árvore com o mesmo grau de beleza, imprima aquela que possui menos galhos. Se ainda existir mais de uma árvore que satisfaça os mesmos critérios, imprima a que possua menos cortes.

4. Solução adotada

A estrutura de dados escolhida para a solução[2] do problema foi a árvore com números variados de filhos não balanceada na qual foi adicionado além do valor do elemento, a referência do primeiro e próximo, a referência do nó pai, a beleza e uma variável de controle para remoção.

Toda a árvore é enraizada do nó 0, sendo assim, quando é realizada uma entrada é feito uma pesquisa em toda a árvore para encontrar a posição que do elemento a ser inserido, no pior caso é percorrido toda a estrutura. Dessa forma, para N-1 inserções é realizada $\sum_{n=0}^{n-1} i * n$ operações de busca para o pior caso, com complexidade O(n^2). Entradas inválidas são descartadas na inserção. Os nós folhas com beleza negativas são marcadas para remoção no momento da inserção.

Após o processo de inserção, é realizado uma busca na árvore por todos os nós que possuem valor de beleza menor igual a zero e armazenados em uma lista, sua complexidade é O(n). Para cada elemento da lista, é realizado um cálculo da beleza dos nós filhos desta subárvore, caso for menor igual a zero, todos os nós dessa subárvore são marcados para remoção.

Após esse processamento, é calculado a beleza final da árvore e apresentado os identificadores dos galhos que de menor profundidade que foi marcado para remoção, pois quando um galho é removido todos os seus dependentes são removidos automaticamente.

São removidos as subárvores com beleza igual a zero, pois há uma restrição do problema que se existir mais de uma árvore com o mesmo grau de beleza apresentar a que possui menos galhos.

5. Casos de Testes

Foram realizados diversos testes sendo observados 4 saídas possíveis de acordo com as instâncias.

1. Não existe nó a ser removido, a beleza da árvore é a somatória de todos os nós;
2. A somatória da beleza da árvore é menor ou igual a zero, logo o nó a ser removido é a raiz.
3. Existe um ou mais nós a ser removido e a beleza da árvore é maior que zero;
4. Existe mais de uma árvore com o mesmo grau de beleza, apresentar aquela que possui menos galhos. Se ainda existir mais de uma árvore que satisfaça os mesmos critérios, imprima a que possua menos cortes.

5.1. Instâncias e Resultados

Para cada saída, foram construídas 5 instâncias de teste, com entradas de N variando 2, 10, 100, 1000 e 10000. Todos os resultados apresentados corresponderam com os resultados apresentados no uDebug[3].

As instâncias com N igual a 2, obtiveram somente dois resultados, para a saída 2 foi apresentado que havia uma remoção e grau de beleza 0 e, para todos os outros resultados o grau beleza máxima da árvore sem nenhum corte.

[2]https://github.com/gislainy/2575-Christmas-Tree

[3]https://www.udebug.com/URI/2575

Para as saídas 1 e 2, todas as instâncias tiveram os resultados esperados, no caso 2, todas as saídas foram que havia grau de beleza 0 e o nó a ser removido era a raiz. No caso da saída 1, quanto maior o N, maior o número de nós a ser removido e consequentemente o grau de beleza.

No caso 4, foram construídas instâncias com o grau de filhos para cada nó igual a 10. Foram criadas 8 instâncias de testes, sendo 2 grupos de 4 instâncias, sendo um grupo com os galhos pares com grau de beleza positivo e ímpar negativo e o outro grupo o inverso. Para todos os testes, foi observado que os números de galhos a ser removidos correspondiam a 45% dos galhos totais da árvore.

6. Conclusões

A solução adotada atende os critérios estabelecidos para o problema proposto. O fato de não possui um conhecimento prévio de como os dados estão organizados traz uma complexidade maior ao realizar operações de busca e inserção, além das operações que foram inseridas para solucionar o problema, como a função de calcular os nós que serão removidos e cálculo da beleza da árvore resultante.

A árvore com um número variados de filhos não balanceada deve somente ser utilizada quando realmente não se sabe um número máximo de filhos que um nó pode ter, pois, quando maior a fica ao longo do tempo, maior o número de comparação para inserção e também exige maior controle para remoção de um determinado nó.

Referências

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to algorithms*. MIT press.

Monard, M. C. and Nicoletti, M. (1993). *Técnicas avançadas de programação Prolog para tratamento de árvores*. Icmsc-Usp.