Relatório 2º projecto ASA 2022/2023

Grupo: AL060

Aluno(s): Henrique Caroço (103860) e Luís Calado (103883)

Descrição do Problema e da Solução

A solução utilizada é baseada no Algoritmo de Kruskal, mas ao invés de calcular a árvore abrangente de menor custo, pretende-se maximizar as trocas comerciais, maximizando assim o custo da árvore e respeitando a triangulação de Delaunay.

Para o mapeamento do problema são usados dois vetores de pares. O primeiro vetor, para cada índice i (0 <= i <= V - 1), apresenta um par que contém o predecessor e o rank do vértice i. No segundo vetor, o primeiro elemento é o peso do arco (w) e o outro elemento é um par correspondente ao arco (u,v).

Análise Teórica

Seja V o número de vértices do grafo e E o número de arcos:

- Leitura do input - O(E);
- MakeSet(int v) – Cria um conjunto disjunto para o próprio vértice, sendo ele o seu próprio antecessor e o seu rank é inicializado a zero - O(1). Como a operação é realizada V vezes, esta componente tem complexidade: O(V);
- Ordenação dos arcos por ordem decrescente dos seus pesos - O(E log(E));
- Função usada por Union: FindSet(int v) – percorre os antecessores de v até encontrar o representante do conjunto em que se encontra - O(1) , sendo realizada E vezes, logo: O(E);
- Union(x, y) – une os conjuntos que contêm os elementos x e y - O(E) no total;
- Output do resultado final - O(1).

Complexidade global da solução: O(E log(V))

Avaliação Experimental dos Resultados

Para gerar os gráficos, foi usado o gerador de grafos realistas, segundo a triangulação de Delaunay.

Foram calculadas 32 instâncias, sendo que cada instância foi calculada pelo menos 3 vezes, sendo usado como tempo a média dos tempos medidos. As instâncias usadas para o número de vértices foram:

Relatório 2º projecto ASA 2022/2023

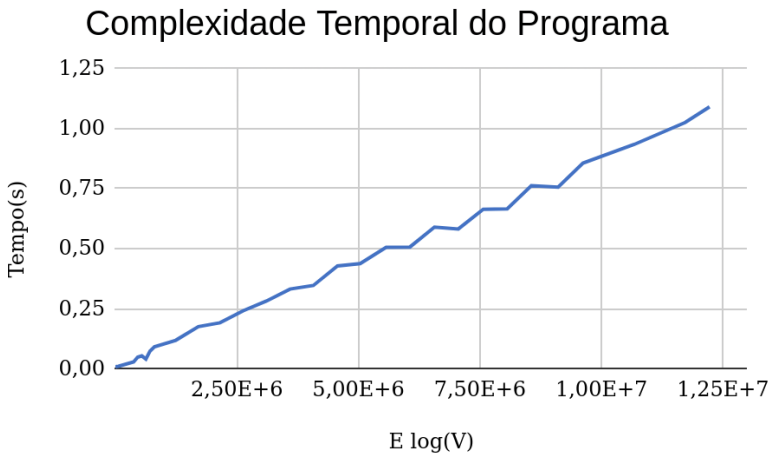
Grupo: AL060

Aluno(s): Henrique Caroço (103860) e Luís Calado (103883)

10, 100, 1.000, 5.000, 10.000, 50.000, 60.000, 70.000, 80.000, 90.000, 100.000, 150.000, 200.000, 250.000, 300.000, 350.000, 400.000, 450.000, 500.000, 550.000, 600.000, 650.000, 700.000, 750.000, 800.000, 850.000, 900.000, 950.000, 1.000.000, 1.100.000, 1.200.000 e 1.250.000. Ou seja, foram usados valores entre 10 e 1.250.000.

O número de arcos foi gerado aleatoriamente pelo gerador de testes, e ao criar um teste foi sempre considerado como coordenada máxima V^2.

Para o gráfico da complexidade temporal colocamos o eixo dos X a variar de acordo com a análise teórica, ou seja, O(E log(V)).



Ao colocar E log(V) no eixo dos X vemos que se verifica uma relação aproximadamente linear com os tempos no eixo dos Y. Assim, podemos concluir que a nossa implementação está de acordo com a análise teórica de O(E log (V)).