Instituto Superior Técnico

Análise e Síntese de Algoritmos

Relatório 2º Projeto - Grupo 48

Afonso Mercier - 83416, Pedro Santos - 83550

Introdução

Com este projeto foi introduzido o problema de custo mínimo de uma rede de transportes. Considera-se que uma cidade que tenha um aeroporto se consegue ligar a outras cidades com aeroportos. Por outro lado, as cidades sem aeroporto necessitam de estradas para outras cidades por forma a ficarem ligadas. O objetivo passa então por construir aeroportos e estradas de forma a que todas as cidades fiquem ligadas minimizando o custo total das obras.

O ficheiro de *input* fornecido é constituído por uma linha que contém o número de Cidades V, uma linha com o número de potenciais aeroportos a construir A, seguido de uma sequência de A linhas especificando onde podem ser construídos e com que custo e uma linha com o número de potenciais estradas E seguido de uma sequência de E linhas especificando entre que cidades podem ser construídas e com que custo.

O problema tem 2 resultados possíveis:

- Insuficiente caso não seja possível construir a rede;
- Caso seja possível construir a rede deverá ser apresentado o custo total da construção bem como o número de aeroportos e o número de estradas a construir.

Descrição da solução

A solução do problema foi concebida utilizando a linguagem de programação C. No desenvolvimento deste projeto representamos as cidades V como vértices de um **grafo pesado e não dirigido**. As estradas E e os aeroportos A foram representados como arcos no grafo. O peso de um arco representa o custo de construção de uma determinada infraestrutura. O grafo foi representado utilizando o número de cidades V e o número de possíveis construções $E + A$.

Este problema reduz-se a encontrar a árvore abrangente de menor custo (MST) do grafo e para isso foi utilizado o algoritmo de **Kruskal** tendo em conta que estamos na presença de um grafo maioritariamente esparso. É um algoritmo baseado na progressiva seleção de arcos por ordem crescente de peso para a construção da MST.

Para a resolução do problema executamos 2 vezes o algoritmo de Kruskal sobre o grafo. Em primeiro lugar sobre o grafo contendo apenas as estradas como arcos. Em segundo lugar sobre o grafo composto pelo conjunto das cidades mais um vértice extra ligado a todas as cidades que possam construir aeroportos com um arco de peso igual ao

custo de construção de um aeroporto na respetiva cidade. É verificada por cada execução do algoritmo se a MST abrange todas as cidades do grafo. Se nem todas as cidades são atingidas pela rede considera-se a execução do algoritmo como insuficiente.

O resultado é escolhido com base na insuficiência e no custo total de ambas as execuções do algoritmo de Kruskal sobre os dois grafos. Se ambas as execuções forem insuficientes o resultado será insuficiente. Se uma delas for insuficiente o resultado será a outra MST sobre o grafo. Se ambas forem suficientes será considerada aquela que possuir o menor custo total.

Análise Teórica

Pseudocódigo do algoritmo implementado:

Kruskal(G, w):

1. X = { }
2. **for** each v ∈ V[G]
3. **do** Make-Set(v)
4. Ordenar arcos de E[G] por ordem de peso não decrescente
5. **for** each $(u, v) \in E[G]$
6. **do if** (Find-Set(u) ≠ Find-Set(v))
7. **then** X = X ∪ $\{(u, v)\}$
8. Union (u, v)
9. **return** X

Seja V o conjunto de cidades (vértices no grafo). Em primeiro lugar o algoritmo de Kruskal é aplicado ao grafo constituído apenas pelas cidades. É contado durante a execução do algoritmo o número de arcos E usado na construção da MST. Em segundo lugar o algoritmo é executado sobre o conjunto $V \cup D$ em que D representa uma cidade auxiliar fictícia (vértice extra acrescentado ao grafo). D encontra-se ligado a todas as cidades que possam construir aeroportos com uma aresta de peso igual ao custo da construção do aeroporto. Ou seja,

$$w(u, D) = custo\ aeroporto\ em\ u, \forall u \in V\ que\ possa\ construir\ aeroporto$$

Sempre que, durante a execução do algoritmo, é adicionada uma aresta ao conjunto X em que o vértice origem ou o vértice destino são o vértice D (cidade fictícia) é incrementado o número de aeroportos a construir (linha 7).

Se no final de cada execução do algoritmo $E \neq \#V - 1$, ou seja, $\#X \neq \#V - 1$ (sendo $\#X$ o número de elementos do conjunto de arcos resultante da execução do algoritmo e $\#V$ o número de vértices) a respetiva MST não cobre todas as cidades, ou seja, não é possível ligar todas as cidades, sendo assim o seu resultado insuficiente.

Para ordenação dos arcos foi utilizado o algoritmo *MergeSort* apresentando uma complexidade de $O(E\ log(E))$.

Os conjuntos disjuntos usados pelo algoritmo foram representados usando uma estrutura baseada em árvores e foram usadas as técnicas de *Union by rank* e *Path Compression*. Assim, uma sequência de m operações, das quais n são operações de Make-Set leva a uma complexidade de $\theta(m\ \alpha(n))$ onde α é a função inversa de Ackermann.

São executadas $O(V)$ operações de Make-Set e $O(E)$ operações de Find-Set e Union logo temos que $O((V+E)\ \alpha(E,V))$. Como $|E| \geq V\ -\ 1$ porque o grafo é ligado temos que $O(E\ \alpha(E,V))$, logo é possível assegurar $O(E\ log(E))$. Como $E < V^2$ conclui-se $O(E\ log(V))$.

Sendo o algoritmo de Kruskal executado duas vezes obtemos assim uma complexidade final de $O(E+A)log(C))$[1].

Avaliação Experimental

Nesta seção é feita uma análise experimental dos resultados de forma a comprovar a complexidade do algoritmo implementado. Usando o gerador de testes fornecido pelo corpo docente, testamos a nossa solução medindo para cada input o tempo de execução usando a função *time* (*linux*).

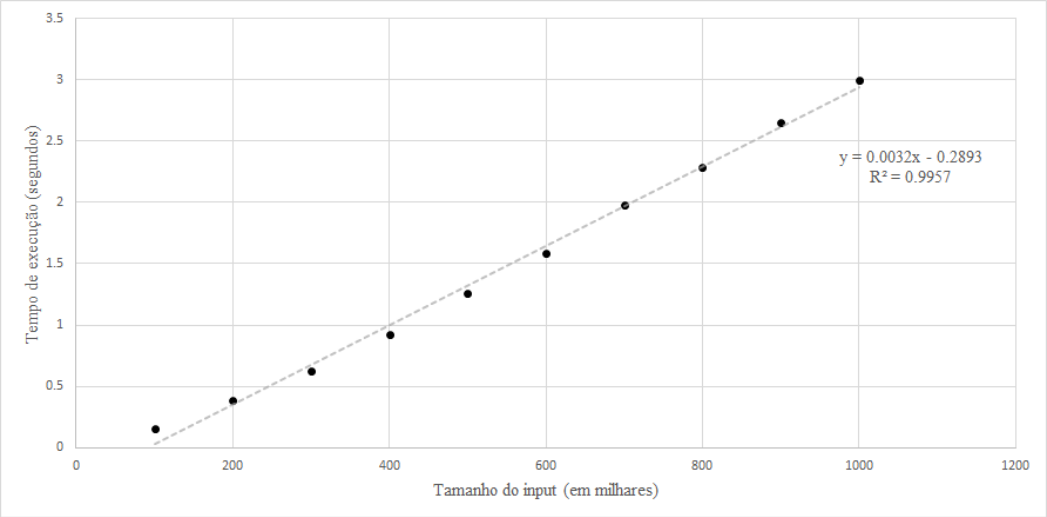

Figura 1: Gráfico da avaliação experimental.

Conclusão

Podemos então concluir que os resultados experimentais estão de acordo com os pressupostos teóricos. O algoritmo apresenta uma complexidade $O(E\ log(V))$, ou seja, trata-se de um algoritmo linearítmico já que o tempo de execução aumenta de forma logarítmica numa relação com o número de vértices proporcional ao número de arcos.

[1] A – número de aeroportos; C – número de cidades; E – número de estradas.

TÉCNICO LISBOA

Referências

- Introduction to Algorithms, Third Edition: Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein; September 2009 ISBN-10: 0-262-53305-7; ISBN-13: 978-0-262-53305-8.
- MIT OpenCourseWare: Lecture 16 - Disjoint-Set Data Structures. https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-design-and-analysis-of-algorithms-spring-2012/lecture-notes/MIT6_046JS12_lec16.pdf
- Wikipedia, Kruskal Algorithm (acedido em 04/2017). https://en.wikipedia.org/wiki/Kruskal%27s_algorithm