**Análise e Síntese de Algoritmos (1º Projeto)**

Grupo 57

Isabel Soares (89466)

Tiago Afonso (89546)

1. **Breve introdução**

Foi-nos proposto desenvolver um projeto no âmbito da cadeira de Análise e Síntese de Algoritmos que se baseia na identificação de sub-redes e em reconhecer routers que levariam ao aumento do número de sub-redes caso estes tenham sido atacados ou desligados.

Deste modo, apresentamos o problema como um grafo orientado onde utilizamos um algoritmo de procura de componentes fortemente ligadas nomeadamente o algoritmo de Tarjan lecionado nas aulas.

1. **Descrição da solução**

A implementação do programa foi realizada em linguagem C.

Neste grafo, os vértices(N) correspondem aos routers e as ligações entre eles(M) correspondem às arestas.

Começámos por criar uma lista de adjacência que contém toda a informação sobre a rede no grafo. De seguida, adicionámos cada uma das ligações entre routers da rede no mesmo grafo.

A partir deste ponto, procurámos o número de componentes fortemente ligadas através de uma DFS. Necessitámos de criar uma função que nos desse os maiores identificadores de cada sub-rede.

Utilizámos, também, o algoritmo qsort da biblioteca de C para ordenar os identificadores das sub-redes.

De seguida, através do Algoritmo de Tarjan, descobrimos os vértices que quebram uma sub-rede, ou seja, quando encontramos pontos de articulação. Os pontos de articulação surgem se, quando removido um router, há um aumento das sub-redes.

1. **Análise Teórica**

Em relação à análise teórica do nosso algoritmo, mais concretamente relacionado com a execução de cada ciclo, tendo em conta que V é o número de vértices de um grafo (neste caso o número de routers) e E o número de arestas de um grafo (neste caso o número de ligações temporais entre elas), temos:

* Inicialização do grafo: O(V);
* Inserção de um arco: O(1);
* Remoção de um arco: O(V);
* DFS: O(V + E);
* Algoritmo RadixSort usando CountingSort: O(d(V + numCC)), onde numCC é o número de componentes fortemente ligadas e d sendo o número de dígitos do valor mais alto de uma lista;
* Algoritmo Tarjan: O(V + E);

Logo, a complexidade final do nosso algoritmo é de O(V + E).

1. **Avaliação experimental dos resultados**
2. **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
* <https://en.wikipedia.org/wiki/Biconnected_component>