Instituto Superior Técnico

Análise e Síntese de Algoritmos

Projeto 1

Grupo 64:

Jorge Rebelo Albergaria Pacheco, nº 86457

Introdução

No âmbito da cadeira de Análise e Síntese de Algoritmos pertencente à licenciatura de Engenharia Informática de Computadores da Universidade do Instituto Superior Técnico, foi proposto como primeiro projeto implementar um algoritmo destinado a calcular componentes fortemente ligados, assim como dizer o número de arcos não pertencentes a esses componentes e demonstrar, ordenadamente, quais são.

O algoritmo foi desenvolvido na linguagem de programação C, dado que, embora mais estruturas e componentes tenham que ser programadas manualmente, é também a linguagem que o elemento do grupo se familiariza mais.

Descrição da Solução

A um nível básico, o algoritmo geral irá disponibilizar de duas estruturas: “Dot” e “List”. Dot assume o comportamento de um vértice de um grafo e possui as características Id, discovery, low, OnStack e group, sendo estes todos variáveis do tipo *Integer*. Por outro lado, a estrutura List assume o comportamento de um elemento de uma lista ligada, guardando um Dot, assim como um ponteiro para o próximo elemento da lista.
Estas estruturas são organizadas em vetores localizados globalmente no programa, dado que os seus conteúdos irão ser mudados em várias partes do programa, e passá-las como argumentos entre funções requer muita memória, especialmente dado o tamanho elevado que estes vetores podem ter em grafos mais complexos.

O programa consiste em quatro fases: criação e alocação de objetos (Vértices e Adjacências), cálculo do número de componentes fortemente ligados, ordenação dos resultados e impressão dos mesmos.

O algoritmo começa por interpretar o número de vértices e arcos, assim como criar um array que será preenchido com vértices e outro que guardará as listas de adjacências de cada vértice que foram introduzidas. Esta fase é constituída por um ciclo “for” com o número de iterações igual ao número de vértices, onde estes são criados, assim como outro ciclo “for” com o número de iterações igual ao número de arcos, onde estes são estabelecidos na lista de adjacências. Noutras palavras, o tempo da fase de criação será O(V + E).

Na segunda fase, a determinação do número de componentes fortemente ligados é feita utilizando o algoritmo das Componentes Fortemente Ligadas de Tarjan, sendo o seu custo O(V + E).

A terceira fase, ou seja, a fase de ordenação dos resultados consiste em aplicar o algoritmo Quicksort (ou “qsort”, utilizando a biblioteca “<stdlib.h>”) a um vetor que contém todos os arcos entre as componentes fortemente ligadas, cujo tempo de execução, no pior caso, será O(N^2), onde N é o número de arcos entre componentes fortemente ligadas, enquanto que a determinação desses arcos será O(V + E), pois todas as adjacências de cada vértice têm de ser analisadas.

Por fim, na última fase, esse vetor de arcos determinados deve ser percorrido duas vezes, uma para determinar o número de arcos únicos (ou seja, sem repetições), e outra para imprimir esses arcos. O tempo desta fase pode, então, ser descrito como O(2 * N).

No total, prevê-se que o algoritmo deva ser O(3(V + E) + 2N + N^2).

Demonstração dos resultados

Seguem-se dois gráficos que demonstram a evolução do tempo em relação ao grafo fornecido, assim como a utilização de memória em relação ao mesmo grafo. Uma rápida visualização de ambos permite concluir que o desempenho do algoritmo (em tempo e memória), é exponencial, potencialmente devido à implementação do algoritmo “qsort”.