Análise e Síntese de Algoritmos

1º Projeto - 23 de março de 2018

77906 António Sarmento 79763 André Dias Nobre

Introdução

O exemplo do Sr. João Caracol com a sua rede e o seu desejo de fazer auditoria serve para evidenciar um problema relacionado com componentes fortemente ligadas (SCC, *Strongly Connected Component*) num dado grafo não dirigido, tendo em conta que uma sub-rede é uma SCC. O objetivo consiste em conhecer o número de sub-redes e os identificadores de cada, e saber como a sua rede será afetada por um ataque aos pontos de articulação (representado como routers) das sub-redes.

Portanto, abstraindo estes dados, procuramos:

1. O número de SCCs na região;
2. As ligações entre as SCCs;
3. Representar as ligações entre as SCCs pelo ponto mais importante.

Análise Teórica

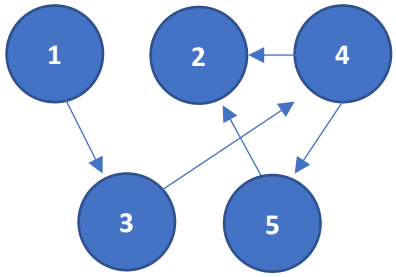
Como estrutura de dados para representar o grafo dirigido, temos os seguintes dados:

- Nº Vértices — V
- Nº Arcos — E

Um grafo G, representado por um vetor de arcos que recorre a três vetores:

1. Tamanho: V; Índice: Número do Vértice; Conteúdo: Índice do Arco;
2. Tamanho: E; Índice: Número do Arco; Conteúdo: Vértice Ligado;
3. Tamanho: E; Índice: Número do Arco; Conteúdo: Arco Adjacente;

Exemplo da representação de um Grafo Dirigido:



#	Vértice
1	1
2	0
3	2
4	3
5	5

#	Arco	Irmão
1	3	-
2	4	-
3	2	4
4	5	-
5	2	-

A nossa estrutura de dados tem a seguinte eficiência para cada operação:

- Espaço: O(V+E)
- Inicialização: O(1)
- Inserir Arco: O(E)
- Encontrar Arco: O(E)

Para a procura de SCCs e de pontos de articulação, aplicamos o algoritmo de Tarjan da seguinte forma:

1. Visitamos todos os vértices de um grafo *G* aplicando uma DFS, começando no vértice de índice 1.
2. A cada visita de um vértice-fonte *s*:
   a. Guardamos *s* numa pilha.
   b. Percorrer os adjacentes.
      i. Se cada vértice adjacente *d* não tiver sido visitado, visitar recursivamente e atualizar o *low* da fonte com o menor entre *s* e *d*.
      ii. Caso *d* esteja já na pilha, atualizar o *low* de s com o menor entre o *low* dele e o tempo de descoberta de *d*.
   c. Quando chegarmos a um vértice previamente visitado cujo o tempo de descoberta e o low sejam iguais, começamos a fazer *pop* dos elementos da pilha, guardando o vértice-mestre correspondente a cada vértice.
3. Correr o Tarjan uma 2ª vez, desta vez tendo em conta os pontos de articulação todos removidos, para saber a SCC de maior tamanho.
4. Reordenamos a ordem dos identificadores com `qsort`, acrescentando O(V log(V)) na complexidade temporal.
5. Expomos (por *print*) o conteúdo de *SCC*, ou seja o número de SCCs e os respetivos identificadores, o número de pontos de articulação e o maior número de vértices num SCC após a remoção de todos os pontos de articulação.

Esta aplicação tem uma complexidade O(V log (V)+E).

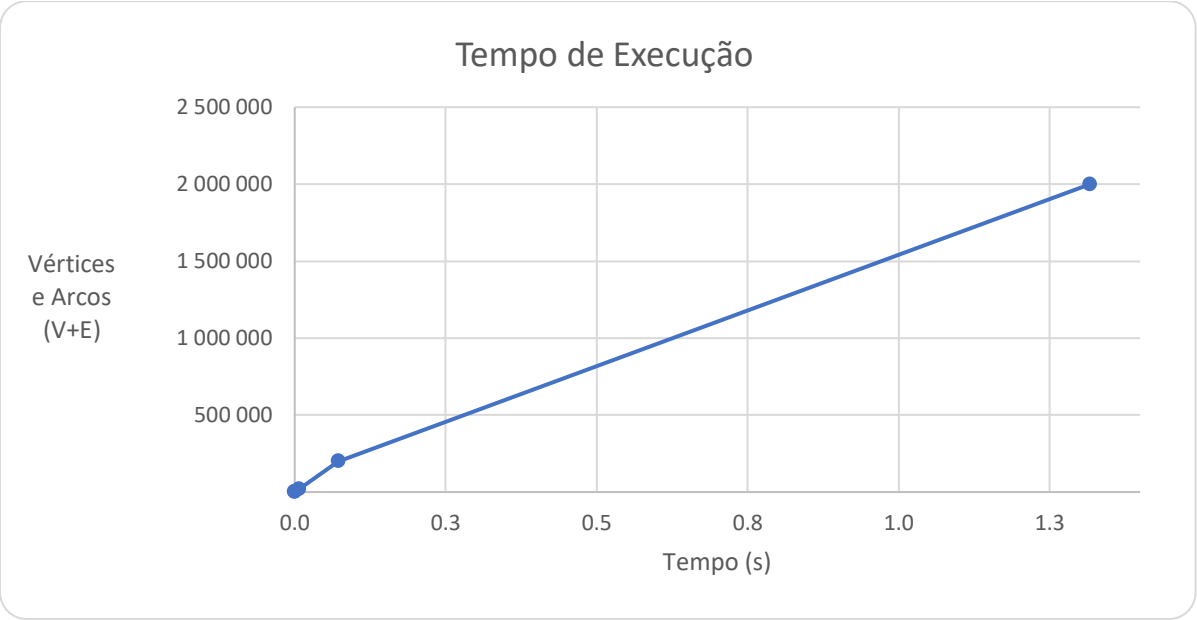
Implementação e Resultados

Implementámos o nosso programa em C pela familiaridade com a linguagem, por ser suficiente para a estrutura de dados que queríamos implementar e por ser *extremamente* eficiente. No sistema Mooshak passámos a todos os testes, obtendo os 16 valores totais.

Apresentamos de seguida cinco testes automaticamente gerados pelo programa fornecido na página da cadeira e aplicado ao nosso programa (testados o Sigma):

V	E	SCC	T(Criação)	T(Algoritmo)	T(Ordenação)	T(Total)
100	100	300	0,000002	0,000035	0,000005	0,000042000
1 000	1 000	3 000	0,000015	0,000715	0,000379	0,001109000
10 000	10 000	30 000	0,000029	0,006724	0,000611	0,007364000
100 000	100 000	300 000	0,000742	0,062247	0,009021	0,072010000
1 000 000	1 000 000	3 000 000	0, 000040	1.196676	0,120406	1,317122000

O gráfico correspondente apresenta-se da seguinte forma:



Como podemos observar, o tempo demorado para criar o grafo é geralmente superior à aplicação do algoritmo de Tarjan. Podemos igualmente observar que os tempos obtidos experimentalmente formam uma reta que corresponde à complexidade teórica esperada de O(V log(V)+E).

Referências

- Introduction to Algorithms (3rd ed.), MIT Press and McGraw-Hill, ISBN 0-262-03293-7.

- Grafos: Slides Introdução Algoritmos e Estruturas de Dados, Profº Francisco Santos IST