Relatório para o 1º Projeto - ASA

Alexandre Caldeira nº 85254

Tiago Miguel Soares Dias nº 84772

1. Introdução

I. Problema:

O Sr. João Caracol tem uma cadeia de supermercados que foi crescendo com o tempo. Atualmente, essa cadeia é enorme e inclui uma rede de distribuição que permite todos os supermercados terem sempre todos os produtos num curto espaço de tempo. O Sr. João Caracol pretende dividir a rede de distribuição em sub-redes regionais de forma a que numa região seja possível qualquer ponto de distribuição enviar produtos para qualquer outro ponto da rede regional. Se um ponto u da rede de distribuição tem uma rota para um ponto v e do ponto v também existe uma rota para o ponto u, então ambos os pontos fazem parte da mesma sub-rede regional. Considerando as atuais rotas de distribuição de produtos, o seu objetivo é ajudar o Sr. João Caracol a identificar as sub-redes regionais.

Input:

O ficheiro de entrada descreve a rede de distribuição da empresa do Sr. João Caracol. O input é definido da seguinte forma:

- Uma linha com o número de pontos de distribuição da rede N (N ≥ 2).
- Uma linha com o número de ligações na rede de distribuição M (M ≥ 1).
- Uma lista de M linhas, em que cada linha contém dois inteiros u e v (separados por um espaço em branco) indicando que os produtos podem ser transportados do ponto u para o ponto v da rede de distribuição.

Assume-se que a identificação dos pontos de distribuição é um inteiro entre 1 e N.

II. Output:

No output, uma sub-rede regional é identificada pelo ponto de distribuição com menor identificador que pertence à sub-rede. O seu programa deverá escrever no output a seguinte informação:

- Uma linha com um inteiro R que denota o número de sub-redes regionais.
- Uma linha com um inteiro L que denota o número de ligações entre sub-redes regionais. Note que entre duas sub-redes regionais, apenas pode existir no máximo uma ligação. Não deve considerar duplicações de ligações entre duas sub-redes.

- Uma sequência ordenada de L linhas correspondendo às ligações entre sub-redes. A sequência deve estar ordenada de forma não decrescente primeiro pelo identificador de origem da sub-rede e depois pelo identificador da sub-rede destino.

2. Descrição da solução

I. Linguagem de programação

A linguagem utilizada para implementação deste problema foi o C. Java foi descartado logo à partida por ser uma linguagem pouco eficiente em termos temporais, já que só o facto de correr sobre uma máquina virtual limita severamente os limites de tempo impostos. C++ não foi utilizado por nenhum membro do grupo se sentir à vontade com esta linguagem apesar de o suporte desta linguagem em relação aos algoritmos estudados ser muito maior e muito mais vasto. Portanto C foi a opção escolhida por já ter sido utilizado em cadeiras anteriores, e ser a linguagem com que ambos os membros se sentem mais à vontade.

II. Estruturas de dados

No nosso programa existem várias estruturas de dados que vão ser enumeradas descritas de seguida:

Stack : Contém um ponteiro para a pilha(de nome array na estrutura de dados) que será depois alocada na função initStack, um inteiro sp (stack pointer), que representa a posição do último elemento adicionado á lista e o inteiro size que contém o tamanho da pilha.

Edge : Representa um arco .

Edge_list : Represento um vetor de arcos

Edge_linked_list_node : Representa uma lista ligada de arcos.

Graph : Contém toda a informação associada a um grafo (número de vértices e arcos) .

Scc_data : Resultado da aplicação do algoritmo tarjan a um grafo.

Tarjan_data : Dados auxiliares para a execução do algoritmo tarjan.

III. Solução

Ao analisar o enunciado, a solução que imediatamente nos ocorreu foi implementar o algoritmo de Tarjan. A estrutura de dados inicial a utilizar foi alvo de debate dentro do grupo, sendo que foi debatido ser uma matriz de adjacências, mas por o tempo de procura nesta estrutura ser mais elevado do que o necessário, concluímos que a lista de adjacências seria o melhor. Foi assim decidido que a melhor solução seria um grafo orientado em que cada vértice corresponde a um ponto da rede de distribuição, e os seus arcos a uma rota de distribuição, isto é se u tem um arco para v, existe uma rota em que o produtos podem ser transportados de u para v.
A seguir bastaria aplicar o Tarjan em todos os vértices para saber o número de sub-redes regionais.

IV. Algoritmo

Para encontrar os número de scc's executamos o algoritmo tarjan, ao fechar um scc's guardamos o menor vértice associado a esse scc(scc → vértice) e para cada vértice associado a esse scc guardamos a que scc ele pertence(vértice → scc). Para remover os arcos duplicados entre scc's criamos um novo objeto graph em que cada scc está associado a um vértice e uma matrix de adjacências associada a esse grafo. Para obter um vetor dos arcos entre scc's distintos percorremos a lista original de arcos e verificamos se os sccs a que os vértices de origem e destino estão associados são diferentes, caso o sejam, modificamos a entrada da mda para TRUE. Ao percorrer um arco verificamos se esta entrada é TRUE, se tal acontecer não é necessário guardar um novo arco associado a estes 2 sccs na novo vetor. Por final basta ordenar o novo vetor de arcos e imprimir cada todos os arcos no terminal.

3. Análise Teórica

1. Função main->O(V+E)

    a) Leitura do input -> Θ (E)

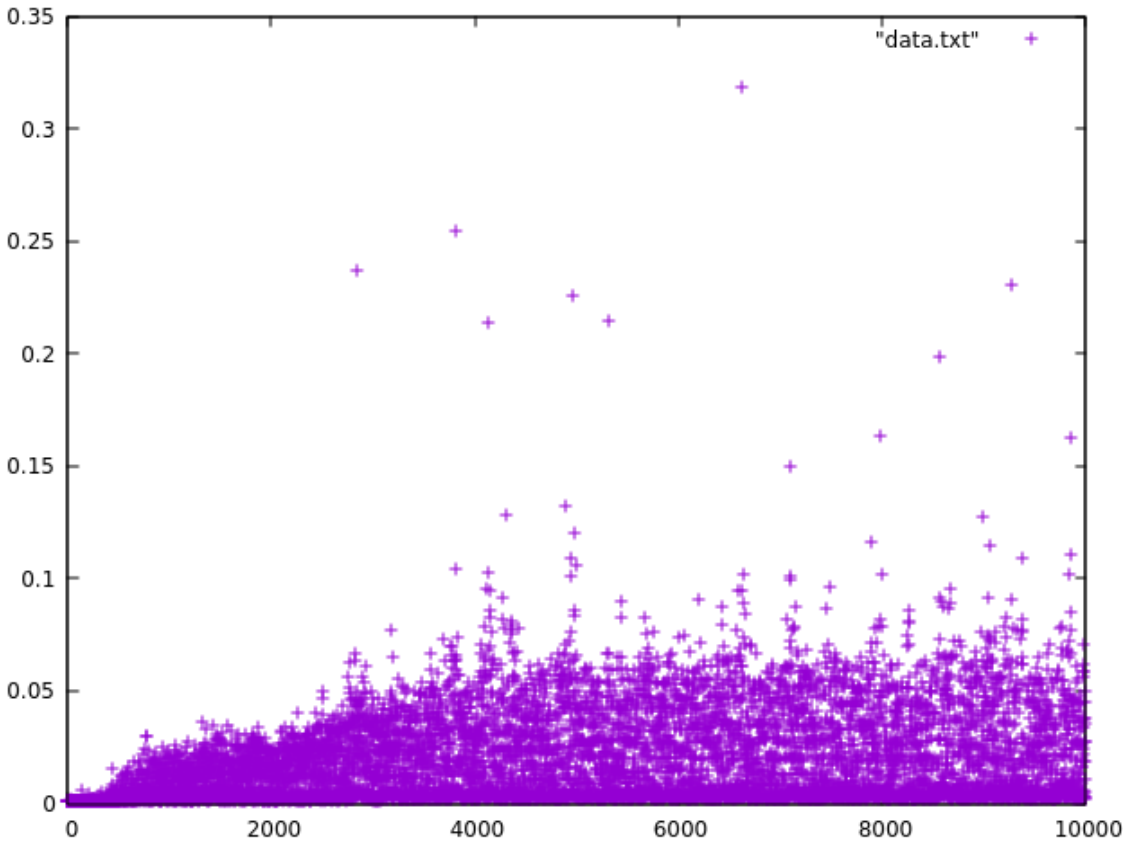
    b)Tarjan -> O(V+E)

    c)HeapSort -> O(V * log(V))

Complexidade do nosso programa é igual à função main, ou seja, **O (V*log(V)).**

4. Avaliação Experimental

O nosso projeto passou com sucesso em todos os testes fornecidos do sistema de testes automáticos.



O gráfico seguinte representa o tempo (de 0 a 0.35) por número de arcos com vértices (de 0 a 10000)

Depois da análise, concluímos que o nosso algoritmo não é linear, sendo que não podemos afirmar que o tempo de execução não aumenta à medida que número de arcos com vértice aumenta nem o inverso.