Relatório do 1º projecto de Análise e Síntese de Algoritmos

Henrique Caldeira, 75838
Pedro Bucho, 69537

Introdução:
O problema deste projecto é um caso de uma aplicação real de grafos não dirigidos.
Cada um dos autores é um dos vértices do grafo, e as relações de colaboração são representadas pelos arcos(ou arestas).

Solução:
Decidimos usar uma representação do grafo por lista de adjecencias por ser aquela em que é mais facil inicializar e guardar o grafo(O(V+E), contra a complexidade O(V^2) para as matrizes).

Como estrutura de dados para representar o grafo, usámos um vector de objectos da nossa classe em C++ 'Node', em que cada um dos quais contém um ponteiro para uma lista ligada com todos os seus vértices adjecentes.

A inserção de um arco no grafo consiste na inserção de um elemento no inicio da lista ligada do respectivo vértice, o que é feito em O(1).

Algortimo:
Para resolver o problema em questão, decidimos usar uma BFS, pois o que se precisa é a distância a que todos os vértices estão da origem.

A implementação deste algoritmo implicou também o uso de uma outra classe em C++ a que chamámos Queue, para representar a fila necessária ao algoritmo. A implementação foi feita de modo a que as operaçoes “enqueue” e “dequeue” tenham complexidade O(1).

Avaliação teórica:

Dado um número V de vértices e E de arcos:

- Na fase de inicialização do grafo, é necessário inserir os vértices e os arcos.
  A inserção dos vértices tem complexidade O(V).
- Já para a inserção dos arcos, como o grafo é 'não dirigido', é necessário inserir cada arco nas duas direcções. Logo a complexidade é O(2E).

  T(inicialização) = t(ins.vertices) + t(ins.arcos) =
  = O(V) + O(2E) =
  = O(V+2E)

- Para o algoritmo, como implementámos a partir do pseudo-código, a complexidade é O(V+E).

  T(bfs) = O(V+E)

Logo, a complexidade total do programa é O(2V+3E) (?)