Universidade Federal do Rio de Janeiro

COS242 - Teoria dos Grafos - 2010.2

**Prof.:** Daniel Ratton Figueiredo

**Alunos:** Bruno Tourinho Tomas e Jonathan Augusto da Silva

# Trabalho de Curso – Parte 1

# Relatório

1. Objetivo

Desenvolver uma biblioteca para manipular grafos, que seja capaz de representá-los, assim como desenvolver um conjunto de algoritmos em grafos. A biblioteca foi desenvolvida de forma que possa ser utilizada por outros programas.

1. Implementação

A biblioteca foi desenvolvida usando a linguagem de programação C++, orientada a objetos.

O objeto que representa o grafo em si é descrito pela classe Graph, que consiste de um conjunto de objetos da classe Edge e de um conjunto de objetos da classe Node.

A classe Node encapsula os elementos de um vértice, a saber: um identificador único (label), um vetor de ponteiros para arestas (edges) e um *flag* de modo a auxiliar nas funções de busca (flag).

Por sua vez, a classe Edge encapsula os elementos de uma aresta, que são: seu peso (weight), os nós conectados a ela (from e to) e um identificador de grafo direcionado (isDirected).

Edge

unsigned long int weight;

Node \*from, \*to;

bool isDirected;

Graph

vector<Node> g\_nodes;

vector<Edge> g\_edges;

Node

unsigned long int label;

vector<Edge\*> edges;

bitset<1> flag;

* 1. Principais observações sobre projeto/implementação
     1. Uso de vector<typename> em vez de *arrays* “puros”

O uso do *container* vector possibilita uma alocação dinâmica de memória para o *array*, permitindo expandi-lo ou contraí-lo quando necessário de modo prático – usando a função resize ou simplesmente adicionando um elemento no seu fim (push\_back).

Outras possíveis funções a serem utilizadas estão descritas em <http://www.cplusplus.com/reference/stl/vector/> .

* + 1. Matriz de adjacência

É sabido que variáveis do tipo bool não ocupam somente um *bit* em memória, e sim um *byte* – por questões de endereçamento de memória. Entretanto, o *container* vector<bool>, uma especialização de vector, usa somente um bit para cada elemento, além de ter a possibilidade de ser referenciado usando os colchetes (“[ ]”), como num *array*.

<http://www.cplusplus.com/reference/stl/vector/> .

1. Estudo de Caso 1

Consiste na análise de parte do grafo de colaboração dos pesquisadores (*collaboration\_graph.txt*), contendo 72000 vértices e 123379 arestas.

* 1. Quantidade de memória utilizada
  2. Tempo de execução
  3. Árvore geradora da BFS
  4. Distribuição empírica do grau dos vértices
  5. Componentes conexos

1. Estudo de Caso 2

Consiste em analisar parte do grafo das redes que formam a Internet (*as\_graph.txt*), que contém 32385 vértices e 46823 arestas.

1. Distribuição empírica do grau dos vértices
2. Componentes conexos
3. Árvore geradora de busca
4. Diâmetro da Internet