Trabalho Prático Final - Fase 1

Professor Mayron Moreira
Universidade Federal de Lavras
Departamento de Ciência da Computação
GCC218 - Algoritmos em Grafos
Valor máximo: 5 pts

23 de abril de 2018

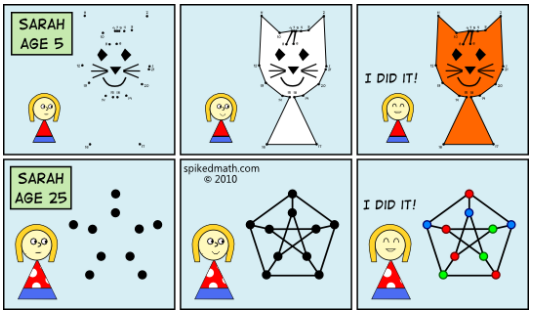

Figura 1: Fonte - `http://spikedmath.com/324.html`

1 Introdução

A resolução eficiente de um problema modelado através de um grafo passa, inicialmente, pela escolha de uma boa estrutura de dados. Em nosso curso, vimos três estruturas elementares:

- Matriz de adjacências;
- Lista de adjacências;
- Matriz de incidências.

Na Fase 1 de nosso projeto, vamos focar na representação de um grafo através dessas estruturas. Para tanto, como não sabemos ainda qual problema será resolvido, implementaremos todas as estruturas disponíveis. Posteriormente, escolheremos a mais adequada ao futuro problema estudado (próximas fases do Trabalho Prático).

2 Detalhamento da proposta

Cada grupo deverá implementar as três estruturas de dados mencionadas na seção anterior. No entanto, durante a execução do programa, apenas uma das representações (matriz de adjacência, lista de adjacência e matriz de incidência) estará com espaço alocado na memória. Assim, ao criar o grafo, o usuário deverá informar qual representação deseja utilizar.

Durante a execução de um algoritmo, podem haver casos em que seja mais vantajoso a utilização de uma estrutura em detrimento de outra. Logo, seu projeto deverá fornecer as seguintes funções:

- Converte matriz de adjacência em lista de adjacência;
- Converte matriz de adjacência em matriz de incidência;
- Converte lista de adjacência em matriz de adjacência;
- Converte lista de adjacência em matriz de incidência;
- Converte matriz de incidência em lista de adjacência;
- Converte matriz de incidência em matriz de adjacência.

Para testar a eficiência das estruturas, implemente os métodos mostrados na sequência e teste para os grafos contidos no arquivo instances.zip (disponível no Campus Virtual).

- obtemVizinhos: recebe um vértice u como parâmetro e retorna um conjunto de vértices vizinhos a u;
- obtemPred: recebe um vértice u como parâmetro e retorna o conjunto de predecessores do vértice em questão (para grafos direcionados);
- obtemSuc: recebe um vértice u como parâmetro e retorna o conjunto de sucessores do vértice em questão (para grafos direcionados);
- ehVizinho: recebe dois vértices u e v como parâmetros e retorna *true* se os mesmos são vizinhos;
- ehPredecessor: recebe dois vértices u e v como parâmetros e retorna *true* se v é predecessor de u (para grafos direcionados);
- ehSucessor: recebe dois vértices u e v como parâmetros e retorna *true* se v é sucessor de u (para grafos direcionados);
- delVertice: deleta um vértice do grafo e as arestas incidentes a ele (por consequência);
- delAresta: deleta a aresta (u, v) passada como parâmetro;
- geraSubgrafoIV: gera um subgrafo induzido por vértices, dado um conjunto de vértices passados como parâmetro;
- geraSubgrafoIA: gera um subgrafo induzido por arestas, dado um conjunto de arestas passadas como parâmetro.

3 Regras para a entrega

- Os alunos farão os trabalhos em grupos de no **máximo 3 pessoas.**
- **Não serão permitidos trabalhos individuais.**
- O trabalho deverá ser implementado em C++ ou Python.
- Cada grupo deverá implementar a estrutura ou classe Grafo.
- Os grafos serão orientados ou não-orientados.
- **Data e hora de entrega: 11/05/2018, até às 23h55.**
- **Qualquer constatação de cópia ou plágio de trabalhos acarretará em nulidade das notas de todos os membros dos grupos envolvidos.**
- O *upload* do código fonte referente a este trabalho deve ser feito no Campus Virtual, na respectiva sala da disciplina dos membros do grupo, em local devidamente especificado.
- Apenas um dos alunos do grupo deve depositar o conteúdo do trabalho do sistema. Para tanto, a pasta compactada com todos os arquivos deve ter o seguinte formato:

  matriculaAluno1_matriculaAluno2_matriculaAluno3.zip (ou tar.gz)

- Os grupos que fizerem o trabalho em **C++** deverão **fornecer um Makefile** para compilação do código.
- Será descontado 1 ponto dos membros de cada grupo cujo trabalho não seguir as especificações estabecidas neste documento.
- Trabalhos entregues após a data limite terão nota nula;
- Clareza e organização do código fonte serão itens levados em conta na avaliação.
- O docente poderá solicitar a presença dos grupos, pessoalmente, para esclarecimentos sobre o trabalho implementado.