Projeto de Programação - Distância entre Vértices e Conexidade

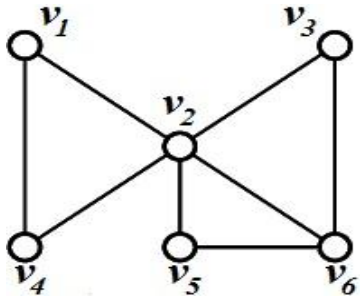
Teoria dos Grafos - 2020

Prof. Roberto C. de Araujo

Este projeto consiste em implementar um algoritmo simples para calcular a distância entre dois determinados vértices de um grafo. Além disto, será feito um ajuste no algoritmo obtido para criar um algoritmo para verificar se um grafo é conexo ou não.

Distância entre dois vértices

Sejam G um grafo e $u,v \in$ VG. A distância de u até *v* em G é definida como o **comprimento de um caminho mais curto** de *u* até *v*. Por exemplo, no grafo abaixo, um caminho mais curto de v_5 até v_4 é o C=(v_5, v_5v_2, v_2, v_2v_4,v_4). Como tal caminho passa por 2 arestas, a distância de v_5 até v_4 é 2.



Apesar de existir um algoritmo eficiente e elegante para resolver este problema (e que será tratado posteriormente no curso), nesta tarefa faremos um algoritmo mais básico:

```
/* Calcula as distancia de Vi até os demais vértices */
calculaDistancia(G, ordemG, Vi)
1. Para cada v∈VG, crie um atributo dist[v]=-1
   (indicando indefinida); como Vi é o vértice inicial
   do caminho, inicialize dist[Vi]=0.
2. Crie uma variável d inicializada com 0;
3. Repita
   a) todo vértice v com dist[v]=-1 que seja
      adjacente a um vértice j com dist[j]=d
      será atualizado com dist[v]=d+1.
   b) incremente d.
   até que o passo a) não seja executado nenhuma
            vez em uma iteração.
```

A ideia do Passo 3 é, inicialmente, "marcar", todos os vértices marcados com **`dist=-1`** que são adjacentes a um vértice com **`dist`**=0, esses terão **`dist`** atualizado como 1; na iteração seguinte, todos os vértices marcados com com **`dist=-1`** que são adjacentes a um vértice com dist=1 atualizarão **`dist`** valendo 2, etc.

Ao terminar a execução do Passo 3, todos os vértices do grafo terão anotadas as distâncias entre eles e o vértice inicial **`Vi`**. No caso de um vértice para o qual não exista nenhum caminho a partir de **Vi**, seu atributo **`dist`** terminará valendo -1

Conexidade

O algoritmo apresentado também pode ser usado, com poucas adaptações, para decidir se um grafo é conexo ou não. Basta considerar um vértice inicial qualquer Vi e marcar o atributo **`dist`** para todos os vértices encontrados a partir dele. Se, após a excução da função, sobrar algum vértice com o atributo dist valendo -1 então o grafo é desconexo.

```
4. Se sobrou algum vértice com dist[Vj]=-1 então
   o grafo é desconexo; caso contrário, o grafo é
   conexo
```

Descrição do projeto

Os alunos deverão usar como base o arquivo **`Grafo.c`**, versão divulgada no início de setembro de 2020, e completá-lo com as duas funções solicitadas neste texto:

- uma para calcular as distância a partir de um vértice, e
- a outra para decidir se um grafo é conexo ou não.

Os algoritmos elaborados deverão, obrigatoriamente, seguir a estratégia descrita neste enunciado.

As estruturas de dados propostas devem ser mantidas, podendo, apenas, ser acrescentados certos detalhes úteis para implementar os algoritmos.

O programa deverá ter uma função **`main`** que **inicializa um grafo com um valor fixo**, executa as funções solicicitadas e, depois, imprime os resultados obtidos.

Observações Complementares

1. O trabalho pode ser feito por grupos de até 4 alunos.
2. Um único aluno do grupo deverá publicar o trabalho no Moodle (penalidade: 2,0 pontos).
3. Deverá ser entregue um único arquivo, com extensão .c, com todo o código na linguagem C (padrão ANSI C).
4. O arquivo deverá conter um cabeçalho (comentário) com as identificações completas de todos os membros do grupo (penalidade: 2,0 pontos).
5. Documentar adequadamente o programa e incluir comentários úteis e informativos.
6. Não usar variável global no programa (penalidade: 2,0 pontos).
7. Seu programa será testado no DEV para Windows.
8. Entrega até as 18:00 horas do dia 5 de outubro de 2020.