INSTITUTO FEDERAL DE MINAS GERAIS

Bacharelado em Ciência da Computação

Disciplinas: Programação II e Algoritmos e Estrutura de Dados I

Trabalho Prático 1

Professores: Mário Luiz Rodrigues Oliveira e
Walace de Almeida Rodrigues

Alunos: Flávia Santos Ribeiro 0022651
Guilherme Cardoso Silva 0022645

Formiga-MG
17 de junho de 2016

Este trabalho tem como objetivo a representação de Grafos em uma TAD que irá gerenciar os seus dados.

A Estrutura utilizada para armazenar os dados é a seguinte:

```
typedef struct dados{
    int Vert1;
    int Vert2;
    int Aresta;
  }*Dados;


struct grafo{
    Dados PGrafo;
    int MaxVertices;
    int MaxArestas;
    int QtdVertices;
    int QtdArestas;
  };
```

A *struct grafo* contém 4 variáveis inteiras que armazenam: a quantidade de vértices, arestas, vértices máximas e arestas máximas que o grafo contem. Existe também uma variável *PGrafo* do tipo *Dados* que é um ponteiro de tamanho *QtdArestas* para uma estrutura que contém 3 variáveis: *Vert1*, *Vert2* e *Aresta*. Estas variáveis são responsáveis por armazenar as arestas do grafo. Para armazenar os nós do Grafo é verificado o valor em *QtdVertices*.

A imagem da estrutura pode ser visualizada na Figura 1.

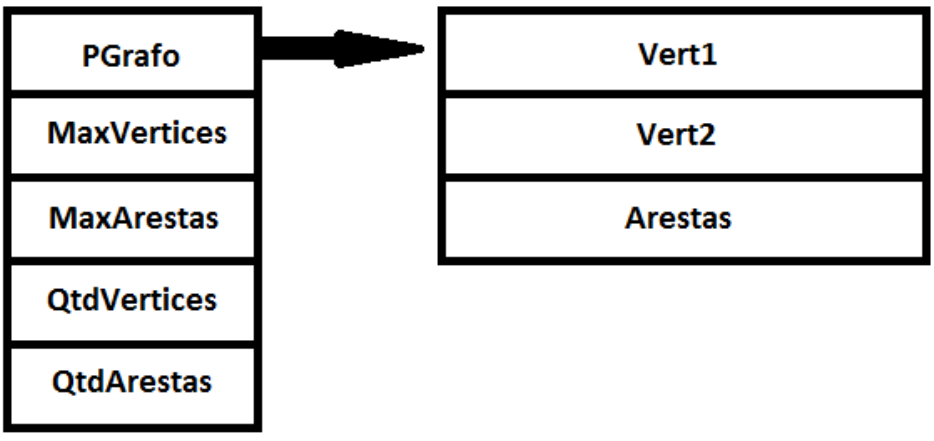

Figura 1 - Estrutura TAD Grafo

As arestas que são introduzidas no Grafo são armazenadas em um vetor que se inicia na posição 1.

A TAD grafo é composta pela estrutura acima junto de 31 funções que gerenciam todos os dados.

Além da TAD Grafo, foi implementado outra função externa chamada de Gcaminho, esta Função utiliza a primeira a TAD para fazer acesso aos dados de um determinado Grafo.

Esta função tem como proposito encontrar o caminho mais curto entre dois nós do Grafo. A metodologia utilizada para implementação desta função é conforme o algoritmo de “Dijkstra”, que funciona da seguinte forma para esta TAD:

É definido quatro vetores: Dist, Caminho, Resposta e Visitados, que irão armazenar respectivamente: a distância entre o início e os outros nós, o pai de cada nó no caminho percorrido, o caminho resposta invertido e os nós que já foram visitados.

Depois é definido os valores iniciais para cada vetor, o vetor Dist recebe INFINITO, o vetor Visitados recebe 0 e o vetor Caminho recebe 0.

A posição do vetor Dist correspondente ao início recebe 0 para que seja o primeiro a ser visitado.

Em seguida a função entra em um loop que só termina quando o fim for visitado.

Primeiro é pego o nó com a menor distância ainda não visitado, depois este nó é marcado como visitado, em seguida busco por todas as arestas que tem o nó que acaba de ser visitado como saída e calculo se a distância até este nó é menor que a já existente.

Este loop irá ser executado até a condição de parada, ou se o nó não visitado com distância mínima for INFINITO que significa que não existe um caminho entre o início e o fim.