Arthur Herbert Silva Melo

TRABALHO PRÁTICO 2

ALGORITMOS I

Descrição

O Problema apresentado no TP2 foi o de encontrar a melhor maneira possível de conectar diversas cidades sem que a distância entre duas cidades superasse um limite D, e sem que houvessem ciclos. Nos foi dado um conjunto de cidades, e suas posições na terra (em latitude e longitude), deveríamos modelar um grafo G(V,A) com estas cidades e suas posições e minimizar, em O(n) o limite D. Tecnicamente falando, o problema pode ser interpretado como “devemos encontrar a árvore geradora mínima de menor bottleneck do grafo G(V,A)”.

Estrutura de dados e Algoritmos

Tipos (structs)

Para ajudar na manipulação e abstração dos dados, foram criados alguns tipos, são eles:

```c
typedef int bool;
#define true 1
#define false 0

typedef struct Edge Edge;

struct Edge {
  int src;
  int dest;
  int weight;
  Edge *next;
};

typedef struct {
  int id;
  bool visited;
  Edge *head;
} Vertex;

typedef struct {
  int V;
  Vertex *array;
} Graph;
```

bool: Tipo usado para abstrair tomada de decisões com *true* e *false*

Edge: Tipo usado para representar a Aresta. Contém o *src*, origem; o *dest*, destino**;** o *weight*, peso; e um ponteiro *next*, para apontando para a próxima aresta da lista de adjacência.

Vertex: Tipo usado para representar um vértice. Contém o *id* do vértice; uma flag *visited*, usada no DFS; e um ponteiro que representa o início da lista de adjacência;

Graph: Tipo usado para representar um grafo. Contém um inteiro V, para representar a quantidade de vértices; e um array Vertex, que representa cada vértice do grafo.

Algoritmo

O algoritmo principal para encontrar a MBST, árvore geradora mínima com o menor bottleneck, foi o algoritmo de Camerini. O pseudo-código se encontra abaixo:

```
function MBST(V, E)
    if |E|
        return E;
    else
        A = Conjunto de arestas maiores que a mediana dos pesos
        B = Conjunto de arestas menores que a mediana dos pesos
        F = Componentes conectados em B
        if |F| = 1 e F contém todos os vértices V
            return MBST(V, B);
        else
            V1 = Um vértice para cada componente conectado em F + Os vértices que não estão em B
            B1 = Arestas que conectam vértices de A + arestas que conectam A aos componentes de F
            return F + MBST(V1, B1);
        end if
    end if
end function
```

Este algoritmo consiste em dividir as arestas do Grafo em duas partes, uma contendo todas as arestas maiores que a mediana dos pesos, e outra com as menores ou iguais à mediana dos pesos. Após a divisão, deve-se buscar os componentes conectados de B (parte menor ou igual à mediana). Se esta busca retornar apenas um componente conectado, e este englobar todos os vértices do Grafo G, chamamos o MBST novamente com este componente, caso contrário, chamamos o MBST com um novo grafo cujos vértices são compostos pelos vértices que não estão em B mais um “super-vértice” para cada componente de F.

Análise de complexidade do algoritmo

Seja a o número de arestas na primeira chamada de MBST() , i a *i-ésima* iteração da função MBST() e v o número de vértices.

Para calcular a mediana e separar as arestas em dois vetores, é necessário O(a/2^i), pois basta ordenarmos as arestas pelo peso e após isso percorrê-las, guardando as maiores e menores que a mediana.

Para encontrarmos os componentes conectados, é necessário, no pior caso, O(|a/2^i||v/2^i|), pois chamaremos um DFS e armazenaremos seu resultado para cada vértice ainda não visitado do Grafo.

Para montar V1 e B1, precisamos somente construir as lista de adjacência, o que pode ser feito em O(a/2^i).

Portanto o algoritmo roda em O(a+ a/2 + a/4 + ... + 1) = O(a).

Prova de Corretude

Considere os dois caminhos do algoritmo

(a) Se F é uma árvore geradora de G, então a MBST é dada pela MBST de B.
(b) Se F não é uma árvore geradora de G, então a MBST de G é dada por F U MBST de um grafo G2. Onde G2 é o grafo cujos vértices são os vértices que não estão em B mais um para cada componente conectado de F.

Como qualquer aresta de B é menor que qualquer aresta de A, por (a) podemos afirmar que a MBST está no grafo composto pelas arestas de B. Analisando (b), considere S como uma MBST de G. F = S - A, pois F contém todos os componentes conectados das menores arestas de G, desta forma S = F U G’. Onde G’ contém as arestas que interligam F a A.