UVA 10034, 11710 e 11733

COMPLEXIDADE

Os algorítmos são construídos de forma semelhante. Sempre ordenando as arestas, sendo ‘a’ o número de arestas e ‘n’ o número de nós. O (a log a) para ordenar as arestas, O(n) para falar que cada pai[i]=i, no pior caso O ((2a + n − 1) lg∗ n), que no meu caso é esse trecho de código :

```
for(int i = 1;i <= m;i++){

        if( procurar(aresta[i].x) != procurar(aresta[i].y) ){


                join(aresta[i].x, aresta[i].y);

                mst[++tam] = aresta[i];

                contador+=mst[tam].dis;

                        aresta[i].x = aresta[i].y;
            }

}
```

Sendo :

```
int procurar(int x){
   if(pai[x] == x) return x;
   return pai[x] = procurar(pai[x]);
}
```

CORRETUDE

Usei o algoritmo de Kruskal nas três questões. O algorítmo de Kruskal, determina uma árvore geradora A, de custo mínimo, conectando com custos reais as arestas.

Assim, vamos provar. Uma árvore geradora de custo mínimo pode não ser única.

Seja B uma árvore geradora de custo mínimo, proxima de A, mostrando-se que A = B.

- Seja e_1, e_2,...,e_{n-1} a sequência das arestas de A.
- Suponha que A!=B. E ‘i’ é o menor índice tal que {e1,e2,...,ei} contido em E(B) e e_{i+1} não pertence a E(B).
- A inclusão de e_{i+1} em B forma um circuito que deve conter essa aresta ‘x’ que não pertence a E(B), senão teríamos um circuito em B.
- Como x, e1,e2…,e_i pertecem à B e o algoritmo consulta as arestas por ordem de custo, conclui-se que peso(x) >= peso(e_i+1), senão x teria sido escolhido no lugar de ei+1. Consideramos a árvore geradora B’= (B - {x}) U {e_{i+1}}.
- Portanto, se peso(x) > peso(ei+1), B’ é uma árvore geradora de custo menor do que B, o que é uma contradição.

- Se peso(x) = peso(e_{i+1}), M’ é uma árvore geradora de custo mínimo, mas B’ contém as arestas e1,e2...ei,e_{i+1}, contradizendo a escolha de B.
- Portanto A=B.

TIAGO LIMA MARINHO