Minimum spanning tree

Georgiana Rusu

September 8, 2018

Profesor: Costin Bădică
Profesor laborator: Alex Becheru

Abstract

Un arbore minim (MST) sau un arbore cu greutate minima este un subgrup al marginilor unui graf neorientat, care leaga toate varfurile, fara nici un ciclu i cu o greutate totala minima posibila. Adica, este un arbore al carui numar de greutati de margine este cat mai mic posibil.

Precizari:

- Specializarea : Calculatoare cu predare in limba Romana
- Anul : *I*
- Grupa : 1.3 B

Universitatea din Craiova,Facultatea de Automatica,Calculatoare si Electronica.

1 Cerinta problemei

Scopul acestui proiect este de a implementa doi algoritmi diferiti pentru aflarea arborelui minim al unui graf neorientat.Vor fi create functii pentru rezolvarea problemei folosing algoritmul Greedy.

2 Schema aplicatiei

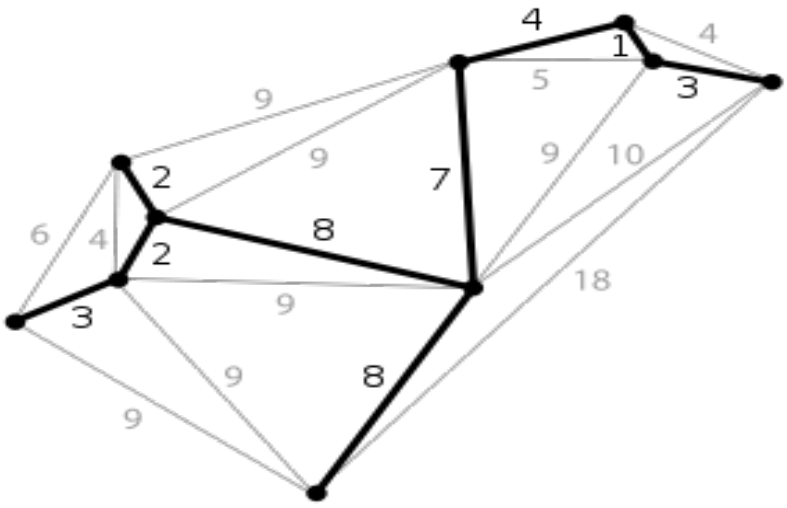

Figure 1: A planar graph and its minimum spanning tree. Each edge is labeled with its weight, which here is roughly proportional to its length.

3 Descrierea aplicatiei

Pentru a reprezenta un graf, ntr-un program, am folosit structuri de date ce reprezinta graful prin matricea sa de adiacenta. Problema se rezolva folosind algoritmul Greedy. Am creat functii pentru rezolvarea fiecarui algoritm. Pentru realizarea grafului , am folosit structurile:

i) **struct edge**,cu ajutorul careia am definit 2 noduri u si v si muchia dintre respectivele noduri (w),care semnifica costul distantei.

ii) **struct edgelist** , in care am definit un vector ce retine lista muchiilor (data[MAX]) si numarul lor (n).

In continuare,am definit urmatoarele functii ajutatoare :

i) Functia **sort()**,ce ordoneaza crescator lista muchiilor in funcie de costul fiecareia.

ii) Functia **find()**, ce cauta in vectorul belongs[] valoarea careia ii apartine respectivul cost

iii) Functia **union1()**, ce uneste varfurile cu muchia data.

iiii) Functia **print()**,afiseaza pe fiecare coloana varfurile si muchia dintre ele.In finalul functiei afisam costul minim.

Descrierea functiilor principale ale acestui proiect:

1. Functia **int prims ()** ce creaza matricele cost[][] si spanning[][].Initializam vectorii visited[],distance[] si from[],costul minim si numarul de muchii.Cautam varful aflat la distanta minima fata de arbore,apoi adaugam muchiile in arborele minim.

2. Functia **void kruskal** . Parcurgem lista muchiilor astfel incat sa retinem fiecare 2 noduri si muchia dintre ele,respectiv costul acesteia. Apoi sortam cu ajutorul functiei sort() nodurile in functie de cost. Retinem costurile intr-un vector (belongs[]). Cream lista muchiilor ce urmeaza sa fie adaugate in minimum spanning tree.Conectam nodurile cu muchiile date. Conectand nodurile,in cazul in care se formeaza un ciclu,eliminam muchia respectiva.Repetam pasii pana ajungem la sfarsitul listei de muchii.

6. Functia **int main()**,cuprinde declararea variabilelor: variabilele total-cost ,iterator-rows, iterator-columns,aux de tip int ; Citim din fisier matricea de adiacenta a grafului. Apelam functiile principale pentru rezolvarea problemei prin cei 2 algoritmi si afisam costul minim.

4 Concluzii

La prima vedere,sau mai bine spus,atunci cand citesti prima data cerinta problemei,nu pare deloc complicat,insa din punctul meu de vedere,cel mai greu a fost sa definesc functiile principale si cele ajutatoare ce rezolva problema prin 2 algoritmi.

5 Bibliografie

i) $The\,C\,Programming\,language, SecondEdition$,Bian W. Kernighan,Dennis M. Ritchie

ii) $Programming\,Techniques\,course$

iii) $The\,Crazy\,Programmer$