Marian Vlad

SUBARBORE DE COST MINIM

CRAIOVA
Martie, 2019

SUBARBORE DE COST MINIM

I Enuntul problemei

Se considera un graf G avand n varfuri si o multime A de muchii. Fiecare muchie are un anumit cost (numar real pozitiv). Se cere determinarea unui subarbore de cost minm al grafului G.

II Detalii teoretice

Pentru rezolvarea problemei se utilizeaza metoda Greedy si algoritmul lui Kruskal.

a) Metoda Greedy

Metoda Greedy se utilizeaza in situatia in care este data o multime A cu n elemente si se cere o submultime B a sa cu m elemente, astfel incat sa fie indeplinite anumite conditii (in multe aplicatii se cere sa fie indeplinit un anumit criteriu de optim).

Exemplu:

Se considera o multime de n numere reale. Sa se determine o submultime a sa, astfel incat suma elementelor submultimii sa fie maxima.

Descrierea metodei Greedy

- Se alege un anumit element din multimea A (dupa un mecanism specific fiecarei probleme in parte);
- Se testeaza daca elementul ales poate fi adaugat multimii B , initial vida;
- Procedeul continua repetitiv pana cand sunt alese toate elementele din B sau pana cand se constata ca nu se poate ajunge la solutie (au fost testate toate elementele multimii A si nu a fost gasita solutia). De aici provine si numele metodei (greedy = lacom).

Se pune problema de a afla mecanismul alegerii unui anumit element (alegerea nu se poate face la intamplare) si de a determina modul in care lementul ales poate fi sau nu adaugat multimii B.

Pentru rezolvare avem urmatoarele posibilitati:

- Dispunem de criterii care conduc la rezultatul dorit;
- Nu dispunem de astfel de criterii, dar avem la dispozitie altele care duc la un rezultat acceptabil (aproape de solutie). In acest caz avem o metoda Greedy euristica.

Metoda Greedy este caracterizata prin faptul ca odata ales un element nu se mai revine asupra alegerii ; poate fi privita ca o particularizare a metodei Backtracking, in care se renuntza la mecanismul de intoarcere.

b) Algoritmul lui Kruskal

Fie G=(X,U) un graf neorientat si conex cu n noduri dat prin vectorul muchiilor.

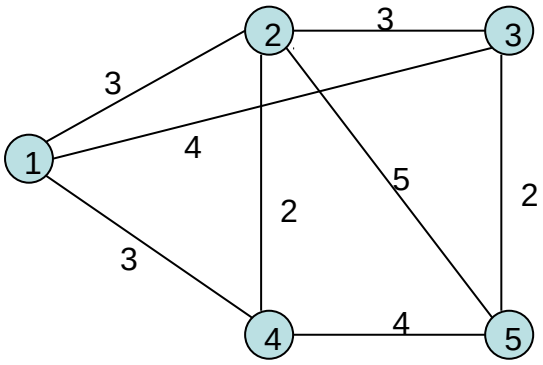
Algoritmul Kruskal determina arborele partial minim astfel:

Se pleaca initial de la n arbori disjuncti H_1 , H_2, ... , H_n , formati fiecare din cate un nod al grafului. La fiecare pas alegem muchia de cost minim neselectata anterior care are extremitatile in doi arbori distincti, in acest fel unificandu-se doi dintre arborii existenti.

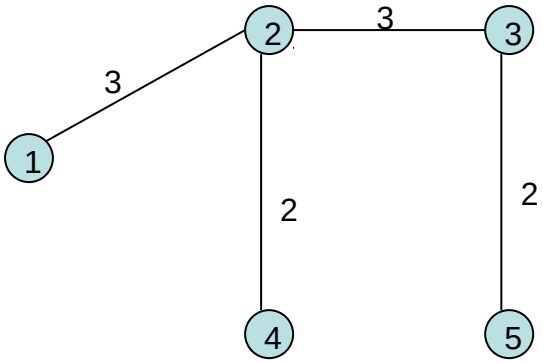
Extremitatile muchiei alese trebuie sa fie arbori diferiti pentru a nu determina aparitia unui ciclu in graful partial de cost minim care se contruieste.

La pasul k vom avea n-k arbori disjuncti. Prin unificarea a doi arbori cu muchia de cost minim, se vor obtine n-k-1 arbori disjuncti. Prin urmare, dupa n01 pasi (dupa alegerea a n-1 muchii) vom obtine un singur arbore (dealtfel, conform teoremei, orice arbore cu n noduri are n-1 muchii). Din modul de alegere al muchiei la fiecare pas, arborele obtinut are costul minim, deci am construit arborele partial de cost minim.

Exemplu:
Fie graful din figura:



Conform algoritmului Kruskal , se aleg in ordine muchiile: [2,4] , [3,5] , [1,2] , [2,3].
Arborele partial de cost minim este:



Costul arborelui este 10.

In implementarea algoritmului se utilizeaza urmatoarelel structuri de date:

- - vectorul v care retine muchiile arborelui si are componentele de tip record, cu urmatoarele campuri:n1 , n2 (extremitatile mcuhiei) si cost (costul muchiei);
- vectorul T care retine pentru fiecare nod arborele din care face parte; initial t[i]=i , pentru i=1,2, ... , n;
- vectorul s care retine pentru fiecare muchie daca a fost selectata (s[i]=1) sau nu (s[i]=0).

Programul Pascal corespunzator este urmatorul:

```
program kruskal;
Type muchie=record
      n1,n2:integer;
      cost:integer;
           end;
var f:text;
       s,t:array[1..100] of integer;
       v:array[1..100] of muchie;
       n,m:byte;
procedure citire;
var i:byte;
begin
  assign(f,'cost.in');reset(f);
  readln(f,n,m);
  for i:=1 to m do
    readln(f,v[i].n1,v[i].n2, v[i].cost);
```

```
 close(f);
end;

procedure minim;
var i,j,nr,p:byte;
begin
  for i:=1 to n do t[i]:=I;
  nr:=1;
  j:=1;
while nr<=n do begin
  while t[v[j].n1]=t[v[j].n2] do
     inc(j);
  p:=t[v[j].n1];
  for i:=1 to n do
    if p=t[i] then t[i]:=t[v[j].n2];
   s[j]:=1;
   inc(nr);
  end;
end;

procedure sort;
var I,tmp:byte;
begin
 for i:=1 to m do
   if v[i].n1>v[i].n2 then begin
     tmp:=v[i].n1;
     v[i].n1:=v[i].n2;
     v[i].n2:=tmp;
   end;
end;

procedure ordonare_cost;
var i,j:integer; tmp:muchie;
begin
 for i:=1 to m-1 do
  for j:=1+I to m do
   if v[i].cost>v[j].cost then begin
       tmp:=v[i];
       v[i]:=v[j];
       v[j]:=tmp;
     end;
end;

procedure afisare;
var i:byte;
begin
 for i:=1 to m do
if s[i]=1 then
 writeln(v[i].n1,' ', v[i].n2,' ', v[i].cost);
end;

begin {main}
  citire;
  sort;
  ordonare_cost;
  minim;
  afisare;
end.
```

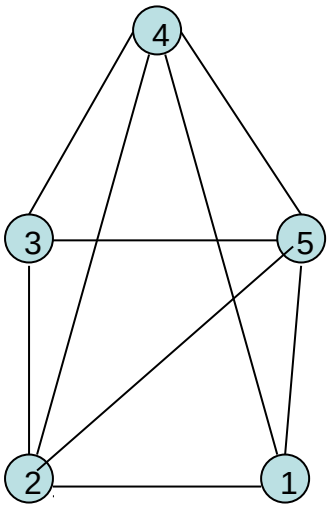
III Detalii de implementare

Algoritmul construieste succesiv multimile de muchii $B_1 \subseteq B_2 \subseteq . . . \subseteq B_{n-1}$, incluse toate in multimea A de muchii ale lui G. B_1 contine o singura mucchie din A si anume cea de cost minim. In continuare, pentru orice numar $i \geq 2$, B_i se obtine adaugand la B_{i-1} acea muchie din A care sete de cost minim printre muchiile neadaugate si care da nastere la circuite prin adaugare. B_{n-1} va contine exact muchiile reprezentand subreteaua de modernizat.

La implementarea algorimului, principala problema este cea a alegerii muchiei de adaugat la B_{i-1} si in special a verificarii conditiei de a nu introduce circuite. Se observa ca muchiile din B_{i-1} compun un graf de tip “padure”, si in sensul definitiei de mai sus. Pentru a nu introduce circuite, muchia de adaguat trebuie sa uneasca noduri ce fac parte din for subarbori diferiti ai “padurii” cu muchiile in B_{i-1}.

Baza de date a programului este conceputa pentru a exploata aceasta observatie. Fiecare arbore din “padure” are un reprezentant si anume nodul de indice minim. Elementul REP[i] stabileste corespondenta intre nodul de indice i si reprezentantul subarborelui din care face parte acest nod. Astfel, o muchie (i,j) nu introduce circuite daca REP[i] este diferit de REP[j]. Vectorul V contine informatii despre muchiile din A, dar este folosit si pentru a reprezenta multimile B_i. Astfel, la etapa i a algoritmului, V[k].a are valoarea TRUE numai daca muchia k face parte din B_i.

Este prevazuta si o optiune de reprezentare grafica a modului de lucru al algoritmului (in cazul in care numarul de noduri ale grafului nu depaseste 9) :



IV Programul Pascal

```
Program ARBSOSTMIN;
uses graph, crt;
type muchie=record {date referitoare la o muchie}
     e1,e2,cost:integer; {e1,e2- indicii nodurilor unite de muchie cost-
costul muchiei}
     a:boolean; {a=true- muchia a fost adaugata in procesul de generare
a subretelei}
end;

var carte, modegraph:integer;
     m, k0,x0,n,k,i,j,x1,x2,y1,y2 : integer ;
     rasp :char ;
     x,y,rep :array[1..35] of integer ;
     v:array[1..40] of muchie; c:array[1..9] of string;
        fg:boolean;
```

```
begin {corp program}
        {introducere date}
        write('Introduceti nr. de noduri:'); read(n);
      write('Introduceti nr. de muchii:'); read(k);
        for j:=1  to k do
          begin
           write('Introduceti o extrem. a muchiei ', j,': ');
           read(v[j].e1);
           write('Introduceti cealalta extrem. a muchiei',j,': ');
           read(v[j].e2);
           write('Introduceti costul muchiei',j,': ');
           read(v[j].cost);
           v[j].a:=false;
          end;  {s-au introdus datele si s-a initializat v}
        for i:=1 to n do
             rep[i]:=I; {fiecare subarbore are initial un singur nod}
        k0:=0;
  {      write('Doriti o reprezentare grafica?(d/n):');
        readln; read(rasp);
        If rasp='d' then fg:=true
                    Else fg:=false;
        If fg then
           Begin {desenare graf dat}
  {    Carte:=cga;
            Modegraphe:=cgac1;
         Initgraph(carte,modegraph,' ');

         For i:=1 to 9 do
          C[i]:=chr(48+i);
         For i:=1 to n do
          Begin
           X[i]:=160+trunc(80*cos(2*i*pi/n));
           Y[i]:=100+trunc(80*sin(2*i*pi/n));
           Circle(x[i],y[i],8);
           {floodfill(x[i],y[i],6); }
   {   settextstyle(2,0,4); {inscrierea numarului nodului }
   {   outtextxy(x[i]-4,y[i]-8,c[i])
           end;
            setcolor(2);
        for j:=1 to k do {se traseaza muchiile}
    {       begin
             x1:=x[v[j].e1];y1:=y[v[j].e1];
             x2:=x[v[j].e2];y2:=y[v[j].e2];
             line(x1,y1,x2,y2);
            end;
        setcolor(1);
    end; delay(1000);
      {se genereaza arborele de cost minim}
       repeat
       k0:=0;
       m:=maxint;{se alege muchia de adaugat}
      for j:=1 to k do
         if (rep[v[j].e1] <> rep[v[j].e2])
         and (not (v[j].a)) and (m>v[j].cost) then
       begin
        m:=v[j].cost;
        k0:=j
       end;
```

```
          if k0 <> 0 then
        begin {se adauga o noua muchie}
          v[k0].a:=true;
          if rep[v[k0].e1] < rep[v[k0].e2] then
             begin  {actualizarea reprezantarilor nodurilor}
              for i:=1 to k do
                If  rep[i]=rep[v[k0].e2] then  rep[i]:=rep[v[k0].e1]
             End
           end
           Else
         For i:=1 to k do
           If rep[i]=rep[v[k0].e1] then Rep[i]:=rep[v[k0].e2];
  {         If fg then {se deseneaza muchia adaugata}
  {       Begin
           X1:=x[v[k0].e1];y1:=y[v[k0].e1];
           X2:=x[v[k0].e2];y2:=y[v[k0].e2];
           Delay(1500);
           Line(x1,y1,x2,y2)
         End}
           Until k0=0;
{        end;}

       If not fg then { in mod neografic se listeaza muchiile arborelui
de cost minim}
         Begin
         Writeln;
         Writeln('Din arborele de cost minim fac parte muchiile');
         For j:=1 to k do
                 If v[j].a then writeln(j,' intre nodurile ', v[j].e1,' si
',v[j].e2);
                 End;
 {        Else
         Begin
         Rasp:=readkey;
         Closegraph inchiderea modului grafic
         End;                                    }
End.
```

V Exemplu de executie

Introduceti nr. de noduri: 5
Introduceti nr. de muchii: 9
Introduceti o extrem. a muchiei 1:1
Introduceti cealalta extremitate a muchiei 1:2
Introduceti costul muchiei 1:10
Introduceti o extrem. a muchiei 2:1
Introduceti cealalta extremitate a muchiei 2:5
Introduceti costul muchiei 2:7
Introduceti o extrem. a muchiei 3:2
Introduceti cealalta extremitate a muchiei 3:3
Introduceti costul muchiei 3:8
Introduceti o extrem. a muchiei 4:2
Introduceti cealalta extremitate a muchiei 4:4
Introduceti costul muchiei 4:3
Introduceti o extrem. a muchiei 5:2
Introduceti cealalta extremitate a muchiei 5:5
Introduceti costul muchiei 5:11
Introduceti o extrem. a muchiei 6:3
Introduceti cealalta extremitate a muchiei 6:4
Introduceti costul muchiei 6:5
Introduceti o extrem. a muchiei 7:3
Introduceti cealalta extremitate a muchiei 7:5
Introduceti costul muchiei 7:14
Introduceti o extrem. a muchiei 8:4
Introduceti cealalta extremitate a muchiei 8:1
Introduceti costul muchiei 8:7
Introduceti o extrem. a muchiei 9:4
Introduceti cealalta extremitate a muchiei 9:5
Introduceti costul muchiei 9:3

Doriti o reprezentare grafica? (d/n):n

Din arborele de cost minim fac parte muchiile
2 intre nodurile 1 si 5
4 intre nodurile 2 si 4
6 intre nodurile 3 si 4
9 intre nodurile 4 si 5