Ministerul Educatiei, Cercetarii si Tineretului
Grup Scolar „Gh. Asachi” Galati

Proiect pentru obtinerea certificatului de competente profesionale

Specializare : matematica-informatica

2006-2007

Tema proiectului: Grafuri hamiltoniene si euleriene

Scopul proiectului: Prezentarea teoriei si a aplicatiilor grafurilor hamiltoniene si euleriene

Absolvent:
Profesor indrumator:

CUPRINS

Grafuri euleriene

Adeseori suntem tentaţi să credem simplul fapt de a traversa străzi sau poduri nu implică nici o idee deosebită. Iată însă că există o celebră problemă de traversare în care singura idee implicată este aceea de "traversare", **problema celor şapte poduri din Königsberg**. Această banală şi totuşi foarte controversată problemă a dus la apariţia şi dezvoltarea teoriei grafurilor.

Problema se pune cam aşa:
Oraşul Königsberg era aşezat pe coasta Mării Baltice, la gurile râului Pregel. Pe râu erau două insule legate de ţărmuri şi între ele de şapte poduri ca în figura 1.

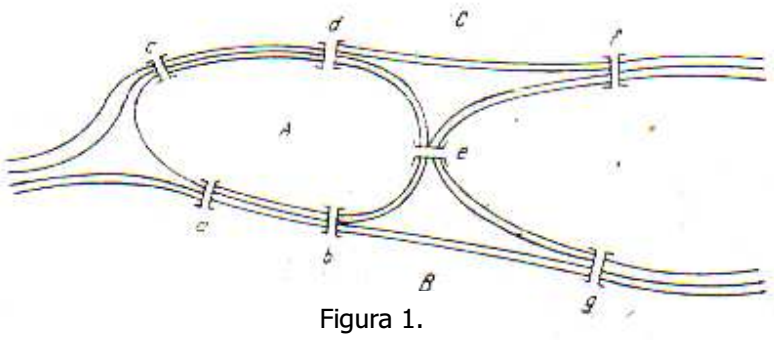
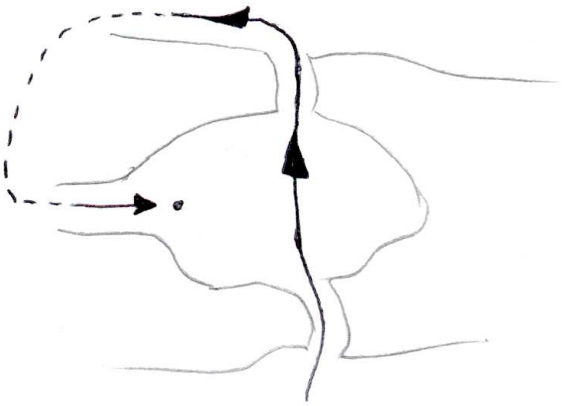


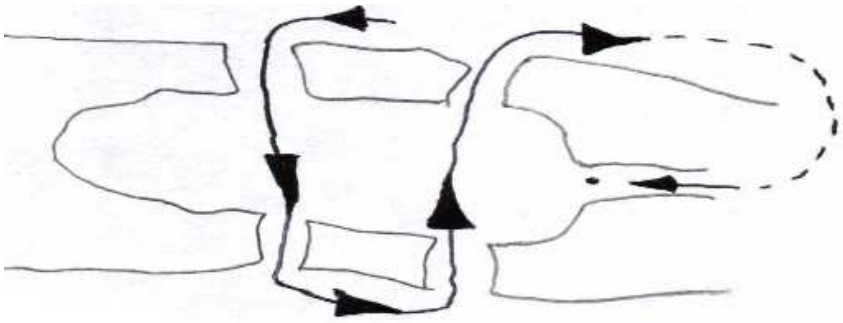
Figura 1.

Oamenii care cutreierau aceste insule au observat că dacă porneau de pe malul sudic al râului, nu puteau să-şi planifice plimbarea astfel încât să traverseze fiecare pod o singură dată. Se părea că ori trebuia să sară un pod ori să-l traverseze de două ori.
În anul 1735 Euler a descoperit că nu mai are rost să se încerce, propunând următoarea analiză a problemei, din punct de vedere matematic:
Să considerăm mai întâi *insula estică* (fig.2.):



sunt trei poduri care duc la ea. Deoarece se pleacă de pe malul sudic, înseamnă că se pleacă ***din afara*** insulei estice. Deoarece fiecare din cele trei traversări trebuie efectuate o singură dată, plimbarea trebuiesă se termine ***pe*** insula estică.

Să considerăm acum insula vestică:
sunt cinci poduri care duc pe ea, iar cinci este din nou număr impar. Aşadar plimbarea începe ***în afara***



insulei, şi deci trebuie să se termine ***pe*** insula vestică.
Aceasta înseamnă că plimbarea se termină în două locuri diferite simultan ceea ce e imposibil.
Soluţia dată de Euler este tipică pentru personalitatea şi ingeniozitatea sa. Tot el a scris în anul 1736 prima lucrare de teorie a grafurilor despre problema acestor şapte poduri.

Un ciclu al unui graf G care conţine toate muchiile lui G se numeşte *ciclu eulerian*. Un graf G care are un ciclu eulerian se numeşte *graf eulerian.*
Un graf G fără vârfuri izolate este eulerian dacă şi numai dacă este conex şi gradele tuturor vârfurilor sale sunt numere pare.
Din punct de vedere al teoriei grafurilor, problema se pune cam aşa: cele patru regiuni (insule şi maluri) A,B,C,D şi cele şapte poduri le reprezentăm în graful următor (fig.3.):

Muchiile grafului reprezentând posi-bilităţile de trecere de pe un mal pe un pod şi reciproc.

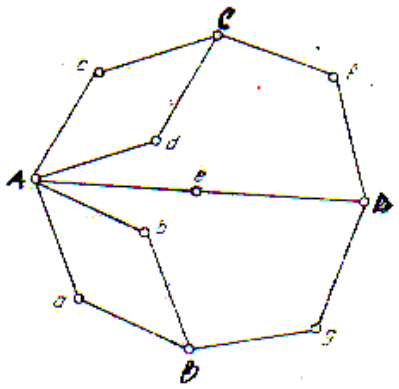


Fig.3

Problema are soluţie dacă acest graf conţine un ciclu eulerian. Un astfel de ciclu, utilizează la fiecare trecere printr-un vârf două muchii ce nu mai pot fi folosite pentru o nouă trecere. Cum fiecare dintre cele patru vârfuri (A,B,C,D) au grade impare, rezultă că ultima muchie va rămâne nefolosită sau va fi folosită pentru a face trecerea de final (pentru a încheia plimbarea). Aceasta ar însemna că ori va rămâne la unul din vârfuri, o muchie nefolosită (fapt ce demonstrează că nu avem un ciclu eulerian) ori plimbarea ar trebui să se termine în mai multe locuri simultan ceea ce e iarăşi imposibil.

Ciclu eulerian*:* Fiind dat un graf neorientat, să se verifice dacă este graf eulerian şi în caz afirmativ, să se determine un ciclu eulerian al său.

Explicaţia programului:

Pornim dintr-un varf neizolat reţinut cu ajutorul variabilei **prim**, în cadrul procedurii de citire, apoi căutăm un ciclu eulerian al grafului printr-un algoritm **backtracking**. Vom folosi pentru reţinerea ordinii vârfului în ciclul eulerian un şir **s**.

În cadrul procedurii de cititre a matricii de adiacenţă, numărăm şi muchiile grafului cu ajutorul variabilei **m**.

Funcţia **valid** verifică dacă vârful **k** aparţine ciclului eulerian (dacă este adiacent cu vârful anterior determinat, iar în cazul în care este ultimul vârf **k=m** dacă este adiacent cu primul vâfr al ciclului).

Procedura **back** caută succesiv, autoapelându-se, vârfuri adiacente cu vârful anterior determinat până când se ajunge la ultimul vârf al ciclului (**k=m**); în acest caz, variabila booleană **găsit** care a fost iniţializată pe **false**, ia valoarea **true** şi este tipărit şirul **s** încheindu-l cu primul vârf al său (pentru a arăta că este un ciclu).

Dacă nu a fost găsit nici un ciclu eulerian al grafului dat (**găsit=false**), atunci graful nu este eulerian şi se tipăreşte mesaj.

Acelaşi lucru se întâmplă şi dacă graful nu are vârfuri neizolate (**prim=0**).

```
program ciclu_eulerian;
uses crt;
type mat=array [1..20,1..20] of integer;
     sir=array [1..20] of integer;
var a:mat;s:sir;
    n,m,i,j,k,prim:integer;
    gasit:boolean;
procedure cit;
begin
 write('n=');readln(n);
 m:=0;prim:=0;
 for i:=1 to n-1 do
  for j:=i+1 to n do
   begin
    write('a[',i,',',j,']=');readln(a[i,j]);
    if a[i,j]=1 then
            begin
             m:=m+1;
             prim:=i;
            end;
    a[j,i]:=a[i,j];
   end;
end;
function valid(k:integer):boolean;
var i:integer;
begin
 valid:=true;
 if a[s[k],s[k-1]]=0 then valid:=false;
 if (k=m)and(a[s[k],s[1]]=0)then valid:=false;
end;
procedure back(k:integer);
var i,j:integer;
begin
 i:=1;
 while (i<=n)and(not gasit) do
```

```
 begin
  s[k]:=i;
  if valid(k) then
   begin
    a[s[k],s[k-1]]:=0;
    a[s[k-1],s[k]]:=0;
    if k=m then
       begin
        gasit:=true;
        writeln('Ciclul eulerian este:');
        for j:=1 to m do write(s[j],' ');
        writeln(s[1]);
       end
          else back(k+1);
    a[s[k],s[k-1]]:=1;
    a[s[k-1],s[k]]:=1;
   end;
  i:=i+1;
 end;
end;
procedure tip;
begin
 clrscr;
 writeln('Matricea de adiacenta:');
 for i:=1 to n do
 begin
  for j:=1 to n do write(a[i,j],' ');
  writeln;
 end;
 writeln;
end;
begin{PP}
clrscr;
cit;
tip;
if prim<>0 then
 begin
  s[1]:=prim;
  gasit:=false;
  back(2);
```

```
  if not gasit then write('Graful nu este eulerian.');
 end
         else write('Graful nu este eulerian.');
 readkey;
end.
```

În viaţa de zi cu zi rezolvăm adesea, fără să ne dăm seama probleme de grafuri euleriene, de exemplu când vrem să mergem cu trenul în circuit şi vrem să plătim mai puţin, calculăm în aşa fel încât să trecem peste tot şi să plătim mai puţin. Dar aceasta nu o facem numai noi şi poştaşii, ci grafurile se utilizează la calcularea poziţilor optime de amplasare a sateliţilor de comunicaţie pentru ca informaţia transmisă să folosească puţin timp, căci în noua eră :,, time is money.

Grafuri hamiltoniene

Se numeste ciclu hamiltonian intr-un graf, un ciclu elementar care contine toate varfurile grafului.
Un graf care contine un ciclu hamiltonian se numeste graf hamiltonian.
Un lant elementar care contine toate varfurile grafului se numeste lant hamiltonian.

* Un graf hamiltonian are cel putin trei varfuri.
* Graful complet cu n varfuri este un graf hamiltonian.

Teorema:

Fie G=(X,U), cu n>=3 varfuri, daca oricare ar fi x un nod al grafului si d(x)>=n/2, atunci graful este hamiltonian.

drum hamiltonian: un drum elementar care trece prin toate nodurile grafului;
circuit hamiltonian: un circuit care trece prin toate nodurile grafului;

În timp, s-au evidenţiat o multitudine de probleme reductibile la găsirea unui drum (sau circuit) hamiltonian într-un graf, cum ar fi:

1. Problema poştaşului (găsirea traseului cel mai scurt care trece pe la toate locuinţele ce aparţin de oficiul poştal la care lucrează acesta);
2. Problema adunării deşeurilor (cel mai scurt drum care trece pe la toate punctele de depozitate a deşeurilor);
3. Problema succesiunii operaţiilor (executarea mai multor operaţii pe o maşină în acea ordine în care suma timpilor consumaţi cu pregătirea maşinii pentru trecerea de la o operaţie la următoarea să fie minim)
4. Ordinea lipirii unor componente electronice pe o placă, etc;

Determinarea drumurilor hamiltoniene
Problema determinării drumului (circuitului) hamiltonian de valoare optimă s-a dovedit deosebit de dificilă, neexistând nici acum un algoritm care să rezolve problema în timp polinomial şi nici măcar o metodă simplă prin care să se decidă dacă într-un graf dat există sau nu drumuri hamiltoniene. Există însă mai mulţi algoritmi, unii exacţi alţii heuristici, care reuşesc, într-un caz sau altul, să rezolve problema satisfăcător şi în timp util.

Teorema Dacă într-un graf orientat fără circuite există un drum hamiltonian atunci acesta este unic.
Demonstraţie Deoarece un drum hamiltonian se identifică cu o permutare a nodurilor grafului, existenţa a două drumuri hamiltoniene implică existenţa a două permutări distincte a nodurilor grafului şi cum două permutări distincte diferă prin cel puţin o inversiune vor exista două noduri x_i şi x_j în ordinea $x_i \rightarrow x_j$ pe un drum şi invers pe celălalt, existând deci un drum atât de la x_i la x_j cât şi de la x_j la x_i, cele două formând împreună un circuit, în contradicţie cu ipoteza.

Graful din figura 1. Este hamiltonian, deoarece ciclul C=[1,2,3,5,4,1] este elementar (pleaca din varful 1 si se incheie tot in 1, iar muchiile [1,2], [2,3], [3,5], [5,4] si [4,1] sunt distincte doua cate doua) si in plus contine toate varfurile.

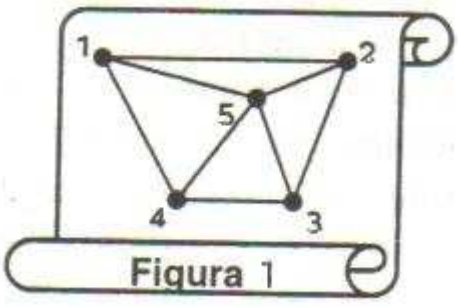


Figura 1

BIBLIOGRAFIE

1. Manual de informatica pentru clasa a XI-a , George Daniel Mateescu / Pavel Florin Moraru, Editura **Niculescu** , 2004

2. Metoda backtracking cu exemple in limbajul Pascal , Tiberiu Socaciu , Editura **Edusoft** , 2005