# THE ALGORITHM DANTZING

**Constantin BOGDAN - .PhD.,** Machines and Installation Departement, University of Petroşani , ROMANIA, tica1234ticabogdt@yahoo.com
**Olimpia PECINGINA - Assistant Professor**, "University of Constantin Brancusi", Tg-jiu ,, ROMANIA, pecinginaolimpia@yahoo.com
**Dragos Vasile VÎGA PhD**, DEPARTAMENT OF MACHINES AND INSTALLATIONS, UNIVERSITY OF PETROSANI, ROMANIA
(e-mail: dragosviga71@gmail.com)

**Abstract**: In this article are made an implementation of an optimum transport networks using methods very useful Dantzing type especially as execution times are good and the algorithm adapts very easily to practical situations.

**Key words**: transport, optimizations, mechanics

## 1. Introduction

Get a graph-oriented G = (V, M) and a function c:M x R → r. Give a nod to go. Require minimal roads from p to any node node i.

EXAMPLE: NUMBER OF NODES =6

| NOD1 | NOD2 | COST |
|---|---|---|
| 1 | 2 | 2 |
| 1 | 3 | 2 |
| 2 | 3 | 4 |
| 2 | 4 | 2 |
| 2 | 5 | 1 |
| 3 | 5 | 1 |
| 4 | 6 | 2 |
| 5 | 4 | 1 |
| 5 | 6 | 7 |

Figure 1

STARTING NODE P = 1 = > node 6 minimum cost is 6

STAGE I:

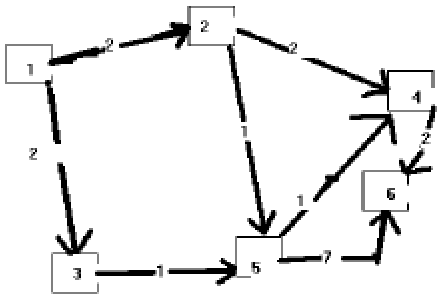


Figure 2

Determines a graph of partial or which may contain nodes 1 and 2 for as little as the cost of going from 1 is 2 so we d[2 ]=cost(1,2)=2     so do I mark that I have gone through the node 2 .

Determines a graph of partial or which may contain nodes 1 and 2 for as little as the cost of going from 1 is 2 so we d[2 ]=cost(1,2)=2     so do I mark that I have gone through the node 2 .

STAGE II

Another possibility is partially composed of tree nodes 1 and 3 I mean d[3]= 2

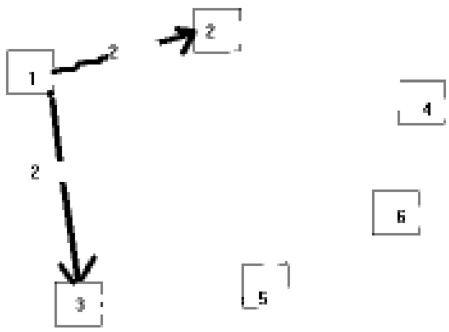


Figure 3

STAGE III

How can I get to node 5?        The loins above can partially add an archive which contains nodes 3 and 5, or to add an arc that encompasses the vertices 2 and 5   adica d[5]=3= cost(1,2)+cost(2,5)= 1+ 2

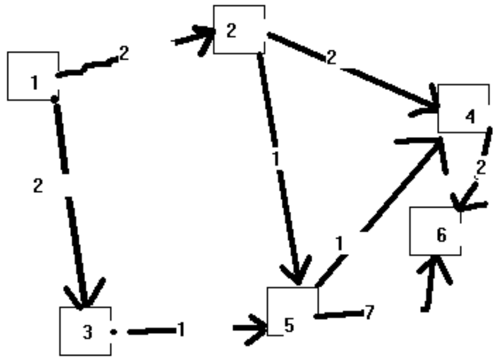


Figure 4

STAGE IV

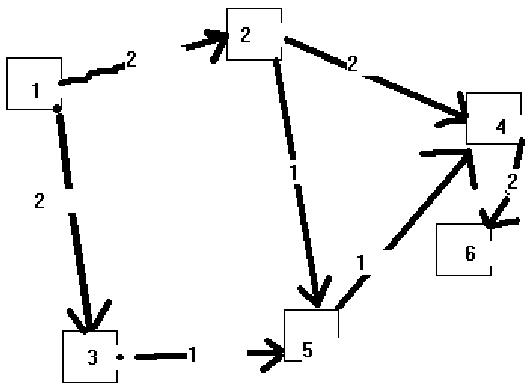


Figure 5

How can I get to the 4?

As D[4]=4=COST(1,2)+COST(2,5)+COST(5,4) =2+1 +1 or

=cost(1,2)+cost(2,4)=2+2

STAGE V

We add the arc formed by the vertices of 4 and 6 So d[6]=6=cost(1,2)+cost(2,4)+cost(4,6)

## 2. Conclusions

Conclusion:
Routes 1 2 4 6
1 2 5 4 6
1 3 5 4 6
care sunt de cost minim 6.
{(The ALGORITHM Dantzig)
Let a graph-oriented G = (V, E) and a cost function: ExR-> R. E contains The n vertices. It means the top p departure. For every top i belongs to V is required to determine all the roads to p at i have minimal cost.
EX:
6
1 2 2
1 3 2
2 3 4
2 4 2
2 5 1
3 5 1
4 6 2
5 4 1
5 6 7
STARTING NODE P = 1 for the node > = 6 minimum cost e de 6}

```
const v=10000;

type ma=array[0..200,0..200] of boolean;
sir=array[0..200 ] of byte;
sirb=array[0..200] of boolean;
interval=0..200;
mc=array[0..100,0..100] of integer;
var a,b:mc;
i,j,n,m,min,k,sos,p:integer;
d,st:sir; sel:sirb;f:text;
procedure minim;
var i,j:integer;
begin
min:=v;
for i:=1 to n do
for j:=1 to n do
```

```
if(sel[i]) and (not sel[j]) then
if (min>d[i]+a[i,j]) then
begin
min:=d[i]+a[i,j];
k:=j;
end;
end;
procedure dantzig;
var i,j:integer;
begin
for i:=1 to n-1 do begin
minim;
if min<>v then begin
sel[k]:=true;
d[k]:=min;
for j:=1 to n do
if sel[j] and (d[j]+a[j,k]=min) then b[j,k]:=a[j,k];
end;
end;
end;
procedure drum(k:integer);
var i,j:integer;
begin
for i:=1 to n do
if (b[st[k-1],i]<v) and not sel[i] then begin
if i<>sos then begin
sel[i]:=true;
st[k]:=i;
drum(k+1);
sel[i]:=false;
end
else begin
for j:=1 to k-1 do
write(st[j],' ');
write(sos,'  de cost');
writeln(d[sos]);
end;
end;
end;
procedure load_graf_cost(var a:mc;var n,m:integer;s:string;digraf:boolean;v:integer);
var i,j,co:integer;f:text;
begin
assign(f,s);
reset(f);
```

```
readln(f,n);
m:=0;
for i:=1 to n do
for j:=1 to n do
a[i,j]:=v;
while not seekeof(f) do begin
readln(f,i,j,co);a[i,j]:=co;
inc(m);
end;
close(f);
end;

begin
load_graf_cost(a,n,m,'in12.pas',FALSE,v);
for i:=1 to n do
for j:=1 to n do b[i,j]:=v;
for i:=1 to n do b[i,i]:=0;
fillchar(sel,sizeof(sel),FALSE);
writeln(' nodul de plecare(DE REGULA SE DA NODUL P=1)   p=');
readln(p); sel[p]:=true; d[p]:=0;
dantzig;
for sos:=1 to n do
if p<>sos then begin
for j:=1 to n do sel[j]:=false;
sel[p]:=true;
st[1]:=p;
drum(2);
end;
readln;
end.
```

## 3. Bibliography


1. BERCEA, N. *Optimizarea capacităţii de excavare a excavatoarelor cu roată cu cupe prin modernizarea acţionărilor electrice*. Revista minelor nr. 12/2002

2. DAFINOIU, M. Analysis of the specific forces and strugths born during the sterile rocks *cutting performed at Roşiuţa open pif.* Annals of the University of Petroşani, Mechanical engineering, vol. 6 (XXXIII), 2004, ISSN 1454-9166

3.FODOR, D. *Exploatări miniere la zi*. Editura Didactică şi Pedagogică, Bucureşti, 1980

4. KOVACS, I., NAN, M.S., ANDRAŞ, I. JULA, D.
*Stabilirea regimului extrem de funcţionare a excavatoarelor cu roată portcupe*. Lucrările ştiinţifice ale Simpozionului Internaţional „Universitaria ROPET 2002” 17-19 oct.2002 Petroşani – Inginerie mecanică

5. KOVACS, I., ILIAŞ, N., NAN, M.S. Regimul de lucru al combinelor miniere.
Editura Universitas, Petroşani, 2000.
6. NAN, M.S., KOVACS, I., ANDRAŞ, I., MAGYARI, A.
Despre re*zultatele încercărilor experimentale privind determinarea unghiului de rupere a lignitului la aşchierea acestuia*. Conferinţa XXXVII-a de Maşini şi Electrotehnică Minieră. Balatongyörök-2004-Ungaria
7. OZDEMIR, L. Comparison of Cutting Efficiencies of Single-Disc, Multi-Disc and Carbide Cutters for Microtunneling Applications, în NO-DIG ENGINEERING Vol. 2, No. 3, 2002
8. PECINGINĂ, GH. *Îmbunătăţirea construcţiei organului de tăiere la excavatoarele folosite* în minele de lignit. Teză de doctorat. Universitatea din Petroşani, 2004.
9. POPESCU, FL. *Programarea şi utilizarea calculatoarelor*. Editura SIGMA PLUS, Deva, 2002
10. RIDZI, M.C. *Analiza experimentală a tensiunilor*. Editura UNIVERSITAS Petroşani, 2004
11. RUMŞISKI, L.Z. *Prelucrarea matematică a datelor experimentale. Traducere din limba rusă*. Editura Tehnică – Bucureşti, 1974