PROIECT IDIVIDUAL

LA INFORMATICĂ

TEMA: TEHNICA RELUĂRII

(BACKTRACKING)

Subtitlu

A realizat: Felicia Cazacu, clasa XI-a “C”

A verificat: Maria Guțu, IPLT “Spiru Haret”

Cuprins

Accentuare Titlu 1 Metoda reluĂRii 1

Titlu 2 Avantajele acesteia 1

Titlu 3 Dezavantajele acesteia 2

Titlu 4 Utilitatea.......................................................................3

Titlul 5 Exemple.......................................................................4

Bibliografie.............................................................................5

# Accentuare Metoda reluĂrii

Metoda Reluării se mai numește Metoda Backtracking. Aceasta este o metodă de elaborare a algoritmilor. Ea se aplică problemelor în care soluţia se poate reprezenta sub forma unui vector, X=(x1,x2,...xm), care aparţine lui S=S1xS2x...Sm

S=S1xS2x...Sm se numeşte spaţiul soluţiilor posibile

Pentru fiecare problemă în parte se dau anumite condiţii între componentele vectorului soluţie care se numesc condiţii interne

Soluţiile posibile care verifică condiţiile interne se numesc soluţii rezultat

Metoda Backtracking îşi propune să genereze toate soluţiile rezultat

O metodă simplă de a genera soluţiile rezultat constă în a genera într-un mod oarecare toate soluţiile posibile şi de a alege dintre acestea doar pe cele care verifică condiţiile interne. Dezavantajul constă în faptul că timpul cerut este foarte mare.

Metoda Backtracking urmăreşte să evite generarea tuturor soluţiilor posibile. Pentru aceasta elementele vectorului x primesc pe rând valori în sensul că lui x[k] i se atribuie o valoare doar dacă componentele din faţa sa x1, x2,...x[k-1] au primit valori.

Dacă lui x[k] i s-a atribuit o valoare, nu se trece direct la atribuirea de valori lui x[k+1], ci se verifică nişte condiţii de continuare, referitoare la x1, x2,...x[k-1], x[k]. Dacă condiţiile de continuare au fost satisfăcute, se trece la calculul lui x[k+1]. Neîndeplinirea lor exprimă faptul că oricum s-ar alege x[k+1],...,x[n], nu se va ajunge la o soluţie rezultat. Evident, ca în cazul neîndeplinirii condiţiilor de continuare va trebui să se facă o altă alegere pentru x[k]. Sau dacă S[k] a fost epuizat, să se micşoreze k cu o unitate, încercând să se facă o nouă alegere pentru xk.

## AVantajele Și importanța

-Utilizează puțin timp

-Utilizează puțină memorie

Metoda Reluării trebuie să fie studiată, deoarece reprezintă o metodă eficientă de rezolvare a problemelor. Împreună cu Metoda Trierii și Metoda Greedy aceasta reprezintă o metodă utilă în informatică și în limbaje de programare ca TurboPascal.

## Dezavantajele

-Necesită condiții bune de continuare, nu există condiții universal

Metoda Reluării sau așa numita Metoda Backtracking nu este întocmai cea mai folosită metodă dar nici cea mai bună metodă de rezolvare a problemelor. Această metodă este una dintre cele mai costisitoare metode ale timpului de execuție, din acest motiv nu este utilizată pe larg în Europa.

**Exemple**

Program Regina;

var x:array[1..100] of byte;

n:byte;

nrsol:word;

procedure scriesol;

var i,j:byte;

begin

inc(nrsol);

writeln('Solutia a',nrsol,'este');

for i:=1 to n do begin

writeln;

for j:=1 to n do

if x[j]=i then write('X',' ')

else write('O',' ');

end;

end;

function potcont(k:byte):boolean;

var i:byte;

atac:boolean;

begin

atac:=false;

for i:=1 to k-1 do

if(x[i]=x[k]) or (k-i=abs(x[k]-x[i])) then atac:=true;

potcont:=not atac;

end;

procedure back(k:byte);

var i:byte;

begin

for i:=1 to n do

begin

x[k]:=i;

if potcont(k) then

if k=n then scriesol

else back(k+1);

end;

end;

begin

read(n);

nrsol:=0;

back(1);

writeln(nrsol,'solutii');

end.

Program Backtracking;

uses crt;

type poz=record x:byte; y:byte; end; TSolutie=array[0..20] of pozitie;

var v:TSolutie; solutie:TSolutie; lungime:byte; n:Byte;

m:Byte;

a:array[1..n,1..m] of byte; ix:Byte;

iy:Byte;

fx:Byte;

fy:Byte;

min:integer;

procedure Citire; var f:text;

i1, i2:byte;

begin

assign(f,'labirint.in');

reset(f);

readln(f,n);

readln(f,m);

for i1:=1 to n do for i2:=1 to m do

readln(f,a[i1,i2]);

readln(f,ix);

readln(f,iy);

readln(f,fx);

readln(f,fy);

Close(f);

end;

procedure Afisare; var i,k:byte;

begin

for i:=1 to min do begin

write('(');

write(solutie[i].x,',');

write(solutie[i].y,',');

writeln(')'); end;

end;

function UltimaPozitie(k:byte):boolean; begin

UltimaPozitie:=(v[k].x=fx) and(v[k].y=fy); end;

function final(k:byte):boolean; begin

final:= (a[v[k].x,v[k].y]=0) and UltimaPozitie(k); end;

22

function Continuare(pozitie:integer; element:pozitie):boolean; var c:byte;

begin

Continuare:=true;

for c:=1 to pozitie-1 do

if (v[c].x=element.x) and(v[c].y=element.y) then begin

end;

end end;

Continuare:=false;

exit; end;

procedure back(k:integer); var element:pozitie;

begin

if final(k-1) then

if k<min

then begin

min:=k-1;

solutie:=v;

end else

else begin

element.x:=v[k-1].x+1; element.y:=v[k-1].y+1; if Continuare(k,element) then begin

v[k]:=element;

back(k+1); end;

element.x:=v[k-1].x+1; element.y:=v[k-1].y-1; if Continuare(k,element) then begin

v[k]:=element;

back(k+1); end;

element.x:=v[k-1].x-1; element.y:=v[k-1].y+1; if Continuare(k,element) then begin

v[k]:=element;

back(k+1); end;

element.x:=v[k-1].x-1; element.y:=v[k-1].y-1; if Continuare(k,element) then begin

v[k]:=element;

back(k+1); end;

procedure Initializeaza; begin

v[1].x:=ix;

v[1].y:=iy;

min:=10000;

23

end;

begin

Citire;

Initializeaza;

back(2);

Afisare;

end.

**BIBLIOGRAFIE**

[**https://www.tcreate.org/generarea\_automata\_a\_programelor\_de\_calculator.pdf**](https://www.tcreate.org/generarea_automata_a_programelor_de_calculator.pdf)

**http://timofti7.simplesite.com/435052643**