

Metoda Trierii

Cuprins

1. Aspecte teoretice	3
Definitii	3
Schema	3
2. Tehnica Greedy	3
Schema generala	4
3. Probleme din cotidian care pot fi rezolvate utilizând această metodă	4
4. Probleme rezolvate	5
Problema 1	5
Problema 2	6
Problema 3 (Tehnica Greedy)	7
5. Concluzie	8
6. Bibliografie	9

1. ASPECTE TEORETICE

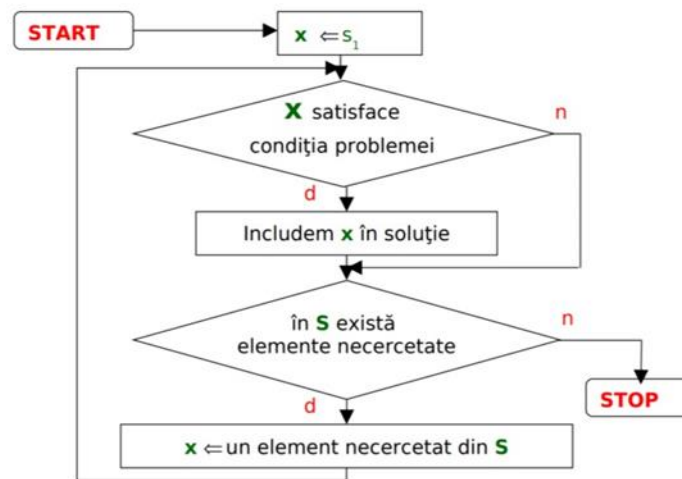
DEFINIȚII

Se numește metoda trierii o metodă ce indentifică toate soluțiile unei probleme în dependență de mulțimea soluțiilor posibile. Toate soluțiile se identifică prin valori, ce aparțin tipurilor de date studiate: integer, boolean, enumerare, char, subdomeniu, tablouri unidimensionale.

SCHEMA

Schema generală a unui algoritm bazat pe metoda trierii poate fi redată cu ajutorul unui ciclu:
For $i:=1$ to k do If SolutiePosibila(S_i) then PrelucrareaSolutiei(S_i)

Unde SoluțiaPosibilă este o funcție booleană care returnează valoarea true dacă elementul S_i satisface condițiile problemei și false în caz contrar, iar PrelucrareaSolutiei este o procedura care efectuează prelucrarea elementului selectat. De obicei, această procedura soluția S_i este afișată pe ecran.



2. Tehnica Greedy

Această metoda presupune ca problemele pe care trebuie să le rezolvăm au următoarea

structura: - se da o mulțime $A = \{a_1, a_2, \dots, a_n\}$ formată din n elemente; - se cere să determinăm

o submulțime B , $B \in A$, care îndeplinește anumite condiții pentru a fi acceptată ca soluție. În principiu, problemele de acest tip pot fi rezolvate prin metoda trierii, generînd consecutiv cele

$2^n(n)$ submulțimi A_i ale mulțimii A . Dezavantajul metodei trierii constă în faptul că timpul cerut

de algoritmi respectivi este foarte mare. Pentru a evita trierea tuturor submulțimilor A_i , $A_i \in A$,

în metoda Greedy se utilizează un criteriu (o regula) care asigură alegerea directă a elementelor necesare în mulțimea A . De obicei, criteriile sau regulile de selecție nu sînt indicate explicit în enunțul problemei și formularea lor cade în sarcina programatorului. Evident, în absența unor astfel de criterii metoda Greedy nu poate fi aplicată.

SCHEMA GENERALA

Schema generală a unui algoritm bazat pe metoda Greedy poate fi redată cu ajutorul unui ciclu: While ExistaElemente do Begin AlegeUnElement (x); IncludeElementul (x); End. Funcția ExistaElemente returnează valoarea true dacă în mulțimea A există elemente care satisfac criteriul (regula) de selecție. Procedura AlegeUnElement extrage din mulțimea A un astfel de element x , iar procedura IncludeElementul înscrie elementul selectat în submulțimea B . Inițial B este o mulțime vidă. După cum se vede, în metoda Greedy soluția problemei se caută prin testarea consecutivă a elementelor din mulțimea A și prin includerea unora din ele în submulțimea B . Într-un limbaj plastic, submulțimea B încearcă să "înghită elementele "gustoase" din mulțimea A , de unde provine și denumirea metodei (greedy- lacom, hrăpăreț).

3. Probleme din cotidian care pot fi rezolvate utilizând această metodă:

- aflarea numărului minim de monede care pot fi date drept plată sau rest;
- medicii deseori se confruntă cu necesitatea aplicării metodei trierii cazurilor, când numărul răniților sau bolnavilor este foarte mare, medicul fiind suprasolicitat, în cazul unui război, sau când își periclitează propria viață în cazul unei epidemii periculoase;
- aflarea ariei maxime a unui lot de teren, avînd la dispoziție o anumită lungime de sîrmă ghimpată, spre exemplu (ca perimetru dat);
- generarea submulțimilor unei mulțimi (aflarea tuturor combinațiilor posibile), ceea ce ne poate fi foarte util în viața de zi cu zi;
- afișarea coordonatelor a două puncte date ce au distanță minimă sau maximă, ceea ce va fi foarte folositor dacă plănuim o călătorie;
- calcularea șanselor de a lua premiul mare la loterie etc.

4. Probleme rezolvate

PROBLEMA 1

Se consideră numerele naturale din mulțimea $\{0, 1, 2, \dots, n\}$. Elaborați un program care determină pentru câte numere K din această mulțime suma cifrelor fiecărui număr este egală cu

m . În particular, pentru $n=100$ și $m=2$, în mulțimea $\{0, 1, 2, \dots, 100\}$ există 3 numere care satisfac condițiile problemei: 2, 11 și 20. Prin urmare, $K=3$.

Program P151;

Type Natural=0..MaxInt; Var l, k, m, n : Natural;

Function SumaCifrelor(i:Natural): Natural; Var suma: Natural;

Begin Suma:=0; Repeat

Suma:=suma+(l mod 10); i:=i div 10;

until i=0; SumaCifrelor:=suma; End;

Function SolutiePosibila(i:Natural):Boolean; Begin

If SumaCifrelor(i)=m then SolutiaPosibila:=true

Else SolutiePosibila:=false; End;

Procedure PrelucrareaSolutiei(i:Natural); Begin

Writeln('i=', i); K:=k+1;

End; Begin

Write('Dati n='); readln(n); Write('Dati m='); readln(m); K:=0;

For i:=0 to n do

If SolutiePosibila(i) then PrelucrareaSolutiei(i);

Writeln('K=', K); Readln;

End.

PROBLEMA 2

Se consideră mulțimea $P=\{P_1, P_2, \dots, P_n\}$ formată din n puncte ($2 \leq n \leq 30$) pe un plan Euclidian. Fiecare punct P_j este definit prin coordonatele sale X_j, Y_j . Elaborați un program care afișează la ecran coordonatele punctelor P_a, P_b distanța dintre care este maximă. Rezolvare. Mulțimea soluțiilor posibile $S=P \times P$. Elementele (P_j, P_m) ale produsului cartezian $P \times P$ pot fi generate cu ajutorul a doua cicluri imbricate:

```

Program P152;

Const nmax=30; Type Punct = record
  X, y: real; End;

Indice = 1..nmax;

Var P:array[Indice] of Punct; J, m, n:Indice;
Dmax:real; PA, PB: Punct;

Function Distanta(A, B: Punct): real; Begin
  Distanta:=sqrt(sqr(A.x-B.x)+sqr(A.y-B.y)); End;

Function SolutiePosibila(j, m:Indice):Boolean; Begin
  If j<>m then SolutiePosibila:=true
  Else SolutiePosibila:=false; End;

Procedure PrelucrareaSolutiei(A, B: Punct); Begin
  If Distanta(A,B)>dmax then Begin
    PA:=A; PB:=B;
    Dmax:=Distanta(A,B); End;
  End;

Begin
  Write('Dati n='); readln(n);
  Writeln('Dati coordonatele x, y ale punctelor'); For j:=1 to n do
  Begin
    Write('P[', j, ']: '); readln(P[j].x, P[j].y); End;
  Dmax:=0;
  For j:=1 to n do For m:=1 to n do
  If SolutiePosibila(j, m) then
    PrelucrareaSolutiei(P[j], P[m]);
  Writeln('Solutia: PA=(', PA.x:5:2, ',', PA.y:5:2, ')');

```

```

Writeln('Solutia: PB=(', PB.x:5:2, ',', PB.y:5:2, ')'); Readln;

End.

```

PROBLEMA 3 (TEHNICA GREEDY)

Se consideră mulțimea $A = \{a_1, a_2, \dots, a_i, \dots, a_n\}$ elementele căreia sînt numere reale, iar cel puțin unul din ele satisface condiția $a_i > 0$. Elaborați un program care determină o submulțime B , $B \subseteq A$, astfel încît suma elementelor din B să fie maximă. De exemplu, pentru $A = \{21,5; -3,4; 0; -12,3; 83,6\}$ avem $B = \{21,5; 83,6\}$.

```

Program P153; Const nmax=1000;

Var A : array [1..nmax] of real; N : 1..nmax;

B : array [1..nmax] of real; M : 0..nmax;

X : real;

I : 1..nmax;

Function ExistaElemente : Boolean; Var i: integer;

Begin ExistaElemente:=false; For i:=1 to n do

If A[i]>0 then ExistaElemente:=true; End;

Procedure AlegeUnElement(var x:real); Var i: integer;

Begin I:=1;

While A[I]<=0 do i:=i+1; X:=A[i];

A[i]:=0;

End;

Procedure IncludeElementul (x:real); Begin

M:=m+1; B[m]:=x;

End; Begin

Write('Dati n='); readln(n); Writeln('Dati elementele multimii A:'); For i:=1 to n do read(a[i]);

Writeln; M:=0;

While ExistaElemente do Begin

```

```
AlegeUnElement (x); IncludeElementul (x); End;  
  
Writeln('Elementele multimii B:'); For i:=1 to m do writeln(B[i]); Readln;  
  
End.
```

5. Concluzie

Avantajul principal al algoritmilor bazați pe metoda trierii constă în faptul că programele respective sînt relative simple, iar depanarea lor nu necesita teste sofisticate. Complexitatea temporală a acestor algoritmi este determinată de numărul de elemente k din mulțimea soluțiilor posibile S . În majoritatea problemelor de o reală importanță practică metoda trierii

conduce la algoritmi exponențiali. Întrucît algoritmi exponențiali sînt inacceptabili în cazul datelor de intrare foarte mari, metoda trierii este aplicată numai în scopuri didactice sau pentru elaborarea unor programe al căror timp de execuție nu este critic.

Prin metoda trierii, programele elaborate afișează pe ecran soluțiile găsite după ce este analizat fiecare element. Deși soluțiile problemei sînt de tip ordinal, ele mai pot fi prezentate sub formă de tablou sau multime. Nu toate problemele pot fi rezolvate prin această metodă, dar anume cele care implică enumerarea după calcul, alegerea între elemente.

6. Bibliografie

<https://www.mindmeister.com/689967199/metoda-trierii>

<http://caterinamacovenco.blogspot.com/p/tehnici-de-programare.html>

<https://www.scribd.com/doc/60874739/Proiect-la-informatica>