Tehnica Greedy

Ciumașu Galina
cl. a XI-a ”D”
IPLT ”Spiru Haret”

Cuprins

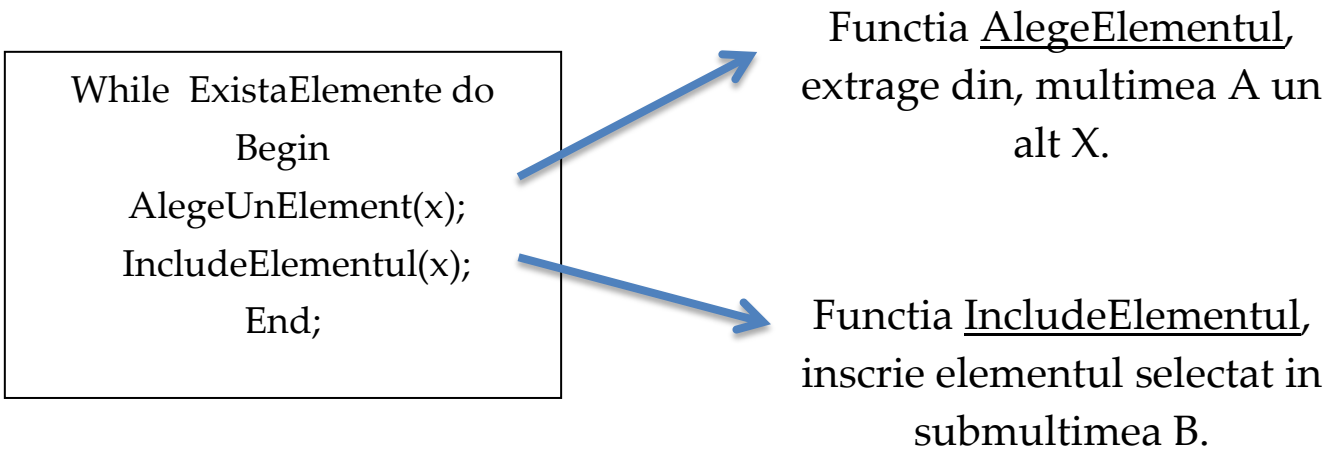
Definiție [1]

Această metodă presupune că problemele pe care trebuie să le rezolvăm au următoarea structură:

- se dă o mulţime $A=\{a_1, a_2, ..., a_n\}$ formată din n elemente;
- se cere să determinăm o submulţime B, $B \subset A$, care îndeplineşte anumite condiţii pentru a fi acceptată ca soluţie.

În principiu, problemele de acest tip pot fi rezolvate prin metoda trierii, generînd consecutiv cele 2^n submulţimi A_i ale mulţimii A. Pentru a evita trierea tuturor submulţimilor A_i, $A_i \subset A$, în metoda *Greedy* se utilizează un **criteriu (o regulă)** care asigură alegerea directă a elementelor necesare din mulţimea A. De obicei, criteriile sau regulile de selecţie nu sînt indicate explicit în enunţul problemei şi formularea lor cade în sarcina programatorului. Evident, în absenţa unor astfel de criterii metoda *Greedy* nu poate fi aplicată. Schema generală a unui algoritm bazat pe metoda *Greedy* poate fi redată cu ajutorul unui ciclu:

Schema generală a unui algoritm bazat pe metoda Greedy poate fi redată cu ajutorul unui ciclu: [3]



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.

Pașii algoritmici metodei greedy [2]

Pasul 1- se inițializează mulțimes S cu mulțimea vidă.
Pasul 2 –se alege din mulțimea A elementului a care este candidatul optim al soluției;
Pasul 3- se elimină elementul a din mulțimea A
Pasul 4 – dacă el poate fii elemnent al soluției,atunci elementul a se adaugă la mulțimea S;
Pasul 5- dacă mulțimea S este soluția problemei,atuni se afisează soluția.astfel se afisează mesajul "Nu s-a găsit soluția".

Dezavantaje și avantaje [7]

Tehnica Greedy este una de optimizare, rulând mai rapid, dar nefiind întotdeauna cea mai bună. Algoritmii Greedy nu conduc în mod necesar la o soluție optimă. Şi nici nu este posibilă formularea unui criteriu general conform căruia să putem stabili exact dacă metoda Greedy rezolvă sau nu o anumită problemă de optimizare. Din acest motiv, orice algoritm Greedy trebuie însoțit de o demonstrație a corectitudinii sale. Demonstrația faptului că o anumită problemă are proprietatea alegerii Greedy se face de obicei prin inducție matematică. Avantajul timpului polinomial, conduce la necesitatea utilizării tehnicii Greedy. Pentru problemele pentru care nu se cunosc algoritmi care necesită timp polinomial, se caută soluţii, chiar dacă nu optime, dar apropiate de acestea și care au fost obţinute în timp util. Multe din aceste soluţii sunt obţinute cu Greedy. De asemenea, Tehnica Greedy poate fi aplicată multor probleme, iar soluția se construiește progresiv, pas cu pas.

Probleme rezolvate

Exemplu 1 [6]

Scrieţi un program, care afişează modalitatea de plată, folosind un număr minim de bancnote, a unei sume întregi S de lei (S<20000). Plata se efectuează folosind bancnote cu valoarea 1, 5, 10, 50, 100, 200 şi 500 de lei. Numărul de bancnote de fiecare valoare se citeşte din fişierul text BANI.IN, care conţine 7 rânduri, în fiecare din care sunt indicate numărul de bancnote respectiv de 1, 5, 10, 50, 100, 200 şi 500 de lei.

```
Program bani;
type tablou=array[1..3,1..7] of integer;
var s,ss,i : integer;
a:tablou;
f:text;
Procedure Afisare(sa:integer);
begin writeln('suma ',s);
if sa<>0 then writeln('nu poate fi transformata cu
bancnotele date ') else
begin
writeln('se plateste cu urmatoarele bancnote');
for i:=1 to 7 do if a[3,i]<>0 then writeln('bancnote de
',a[1,i]:6,' sau folosit ',a[3,i]);
end
end;
```

```
Procedure calcul(var sa:integer);
var nb:integer;
begin i:=7;
while (i>=1) and (sa>0) do
begin nb:=sa div a[1,i];
if nb<>0 then if nb>= a[2,i] then a[3,i]:=a[2,i] else
a[3,i]:=nb;
sa:=sa-a[3,i]*a[1,i];
i:=i-1;
end; end;
begin a[1,1]:=1;
a[1,2]:=5;
a[1,3]:=10;
a[1,4]:=50;
a[1,5]:=100;
a[1,6]:=200;
a[1,7]:=500;
assign (f,'bani.in');
reset(f);
for i:=1 to 7 do readln(f,a[2,i]);
write ('introduceti suma de lei S ');
readln(s);
ss:=s;
calcul(ss);
Afisare(ss);
end.
```

Exemplu 2 [1]

Se consideră mulţimea A={a1, a2, ..., ai, ..., an} elementele căreia sînt numere reale, iar cel puţin unul din ele satisface condiţia ai>0. Elaboraţi un program care determină o submulţime B, B⊂A, astfel încît suma elementelor din B să fi e maximă.

```
Program P153;
{ Tehnica Greedy }
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;
```

```
i:=1;
while A[i]<=0 do i:=i+1;
x:=A[i];
A[i]:=0;
end; { AlegeUnElement }
procedure IncludeElementul(x : real);
begin
m:=m+1;
B[m]:=x;
end; { IncludeElementul }
begin
write('Daţi n='); readln(n);
writeln('Daţi elementele mulţimii A:');
for i:=1 to n do read(A[i]);
writeln;
m:=0;
```

```
begin
ExistaElemente:=false;
for i:=1 to n do
if A[i]>0 then ExistaElemente:=true;
end; { ExistaElemente }
procedure AlegeUnElement(var x : real);
var i : integer;
begin
```

```
while ExistaElemente do
begin
AlegeUnElement(x);
IncludeElementul(x);
end;
writeln('Elementele mulţimii B:');
for i:=1 to m do writeln(B[i]);
readln;
end.
```

Exemplu 3 [5]

O persoană are un rucsac cu ajutorul căruia poate transporta o greutate maximă G. Persoana are la dispoziţie n obiecte şi cunoaşte pentru fiecare obiect greutatea şi câştigul care se obţtine în urma transportului său la destinaţie. Se cere să se precizeze ce obiecte trebuie să transporte persoana în aşa fel încat câştigul să fie maxim.

```
Program rucsac;
Type obiect=record
C,Go,E:real;
end ;
 var ok:boolean;
v:array[1..100] of obiect;
aux:obiect
i,n,t:integer;
G, ct:real;
Begin
write('n='); readln(n);
write('G='); readln(G);
For i:=1 to n do Begin
write('castigul pentru obiectul', i);
readln(v[i].c);
write('greutatea obiectului ',i);
readln(v[i].Go);
v[i].e:=v[i].c/v[i].Go;
End;
t:=1;
Repeat
ok:=true;
for i:=1 to n-t do
if v[i].e<v[i+1].e then Begin
ok:=false;
aux:=v[i];
v[i]:=v[i+1];
v[i+1]:=aux;
end;
 t:=t+1;
until ok;
i:=1; ct:=0;
while (G<>0) and (i<=n) do Begin
if v[i].Go<G then Begin
ct:=ct+v[i].c;
G:=G-v[i].Go;
writeln('obiectul',I,'1');
end
else begin
P:=G*100/v[i].c*p)/100;
writeln('obiectul,I,se introduce in procent de ',P);
end;
i:=i+1;
end;
writeln('castigul total este ',ct);
end;
readln;
End.
```

Exemplu 4 [4]

```
Program teatru;
type teatru=record
ins, sfs:integer; (ora de inceput si de sfarsit a unui spectacol calculata in minute scurse fata de miezul noptii)
ord:integer; (numarul de ordin al spectacolului)
end;
```

```
v[i].ins:=hh*60+mm;
v[i].ord:=i;
end; end;
Procedure afis_piese;
Var i:integer;
Begin
Write ('Inceputurile si sfarsiturile pieselor in minute
```

```
Var v:array [1..30] of teatru;
n, ultim, nr:integer; (n=numarul de spectacole, in variabila ultim avem in permanenta ultimul spectacol selectat, nr=numarul maxim de spectacole)
Procedure sortare_piese;
Var i,j:integer;
temp:teatru;
Begin
For i:=1 to n-1 do
for j:=i+1 to n do
if v[j].sfs<v[i].sfs then
begin
temp:=v[i];
v[i]:=v[j];
v[j]:=temp;
end;
Procedure citire_piese;
Var hh,mm,i:integer;
begin
Write ('Numarul de piese de teatru n= '); Readln (n);
for i:=1 to n do begin
Write ('Piesa cu nr ',i, cand incepe? (ora si minutul)');
Readln (hh,mm);
v[i].ins:=hh*60+mm;
Write ('Piesa cu nr ',i, cand se termina? (ora si minutul)');
Readln (hh,mm);
```

```
scurse de la miezul noptii: ');
for i:=1 to n do
write ('(',v[i].ins,',',v[i].sfs,',',v[i].ord,')');
writeln;
end;
Procedure algo_greedy;
Var i:integer;
Begin
Write ('Piesele posibile, in ordine: ');
ultim:=1; nr:=1;
write (v[i], ' ');
for i:=2 to n do
If (v[i].ins>v[ultim].sfs) then
Begin
Write (v[i].ord, ' ');
ultim:=i;
nr:=nr+1; end;
Writeln ('In total se pot alege maxim',nr,' piese');
end;
Begin
citire_piese;
afis_piese;
sortare_piese;
afis_piese;
algo_greedy;
end.
```

Exemplu 5 [4]

```
Program Maxim;
Var n, a1, a2, c:Integer;
Begin
a1:=-MAXINT;  (initializam primele 2 numere si n cu o constanta predefinita)
a2:=-MAXINT;
n:=-MAXINT;
While n<>0 Do Begin
```

```
If (n>a1) Then a1:=n;  (daca numarul n este mai mare decat primul cel mai mare numar atunci maximul este n)
If (a2<a1) Then Begin
c:=a1;
a1:=a2;
a2:=c;  end; (interschimbare)
Readln (n); end;
Writeln ('a1, ' ',a2');
End.
```

Concluzii

Algoritmii Greedy sunt caracterizați de metoda lor de funcționare: la fiecare pas se alege cel mai bun candidat posibil, după evaluarea tuturor acestora. Metoda determină întotdeauna o singură solutie.

În funcție de specificul problemei, un algoritm greedy poate conduce la soluția optimă sau la o soluție destul de bună, deși suboptimală. Rezultatul unui algoritm greedy pentru o problemă dată depinde și de datele concrete ale problemei, sau chiar de ordinea introducerii lor.

Bibliografie

- http://ctice.md/public/download.php?id=75 [1]
- https://www.slideshare.net/LuminiaMihailov/metoda-greedy-47870787 [2]
- https://www.slideshare.net/BalanVeronica/tehnica-greedy-34061538?next_slideshow=1 [3]
- https://tpascalblog.wordpress.com/ [4]
- https://www.slideshare.net/yoanna_ioana/problema-rucsacului-presentation-948687 [5]
- https://www.slideshare.net/BalanVeronica/metoda-greedy1 [6]
- https://prezi.com/ys_iuqvyoxsi/tehnica-greedy/ [7]