Laborator 4

METODA GREEDY (metoda optimului local) – euristic

I. Repere teoretice

• Se aplică problemelor de optim – dintr-o mulțime de elemente A se cere o

submulțime B care verifică anumite condiții (de obicei soluția unei probleme de

optimizare)

• Are complexitate mică, deci timp de executare mic

• În funcție de particularitățile datelor de intrare, metoda

- Conduce la soluția optimă

- Conduce la o soluție apropiată de cea optimă

- Nu conduce la nicio soluție (deși există una)

II. Aplicații:

1. Problema restului

Fișierul date.in conține pe prima linie două numere naturale, S și N, iar pe al doilea

rând N numere naturale, reprezentând valorile a N tipuri de bancnote.

Să se determine o modalitate de a plăti suma S utilizând un număr minim de

bancnote dintre cele precizate. Există un număr nelimitat de bancnote din fiecare tip.

Datele se scriu în fișierul date.out. Notați timpul necesar executării sursei și

comparați soluția optimă și soluția propusă de metodele indicate.

(Indicație: se ordonează bancnotele după valoarea lor și se includ în sumă mai întâi

cele mai mari)

Exemplu: pentru suma S=267 si valorile 500, 100, 50, 10, 5, 1 se obține soluția

100x2+50x1+10x1+5x1+1x2 (6 bancnote)

Pentru suma S=15 si valorile 10, 7, 1 se obține soluția 10x1+1x5 (6 bancnote); soluția

optimă este 7x2+1x1 (3 bancnote).

2. Problema repartițiilor

Fișierul date.in conține pe prima linie două numere naturale n și m, reprezentând

numărul de sarcini, respectiv numărul de procesoare disponibile, iar pe următoarele n

linii câte un număr naturat ti, reprezentând timpul necesar unui procesor pentru a

executa sarcina i.

Să se determine o modalitate de a distribui sarcinile celor m procesoare, astfel încât

timpul în care cel puțin un procesor lucrează să fie minim. Datele se scriu în fișierul

date.out.

propusă de metodele indicate.

(Indicație: metoda 1 - se ordonează crescător sarcinile și se încearcă distribuirea

fiecăreia la toate procesoarele – se alege acela la care activitatea se va încheia cel

mai repede; metoda 2 - se ordonează descrescător sarcinile și se încearcă

distribuirea fiecăreia la toate procesoarele – se alege acela la care activitatea se va

încheia cel mai repede)

Exemplu: pentru n=9, m=3 si timpii de executare al sarcinilor 3,5,6,10,11,14,15,18,20

Pentru metoda 1 se obține distribuția

P1: 3, 10, 15 (timp de lucru 28)

P2: 5, 11, 18 (timp de lucru 34)

P3: 6, 14, 20 (timp de lucru 40) și timpul necesar 40

Pentru metoda 1 se obține distribuția

P1: 20, 10, 3 (timp de lucru 23)

P2: 18, 11, 6 (timp de lucru 35) și timpul necesar 35

P3: 15, 14, 5 (timp de lucru 34)

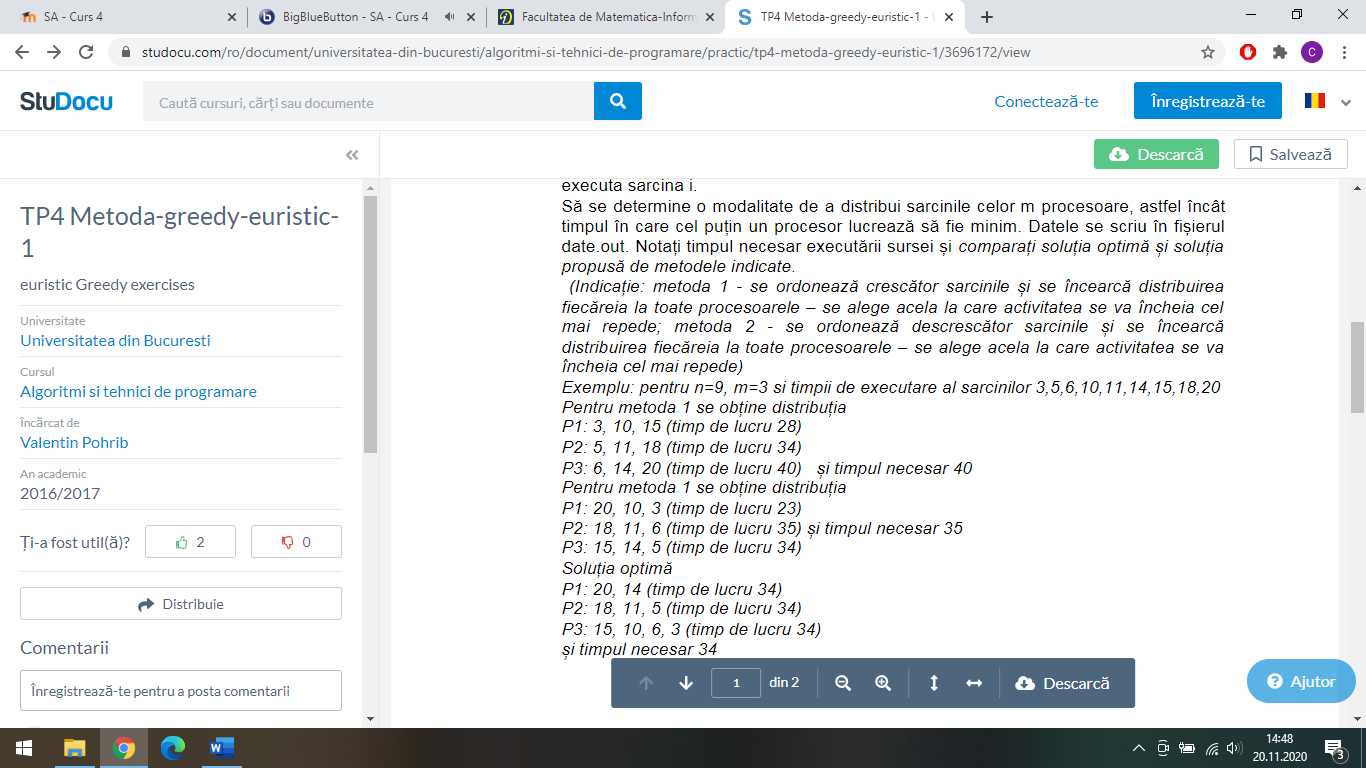
Soluția optimă

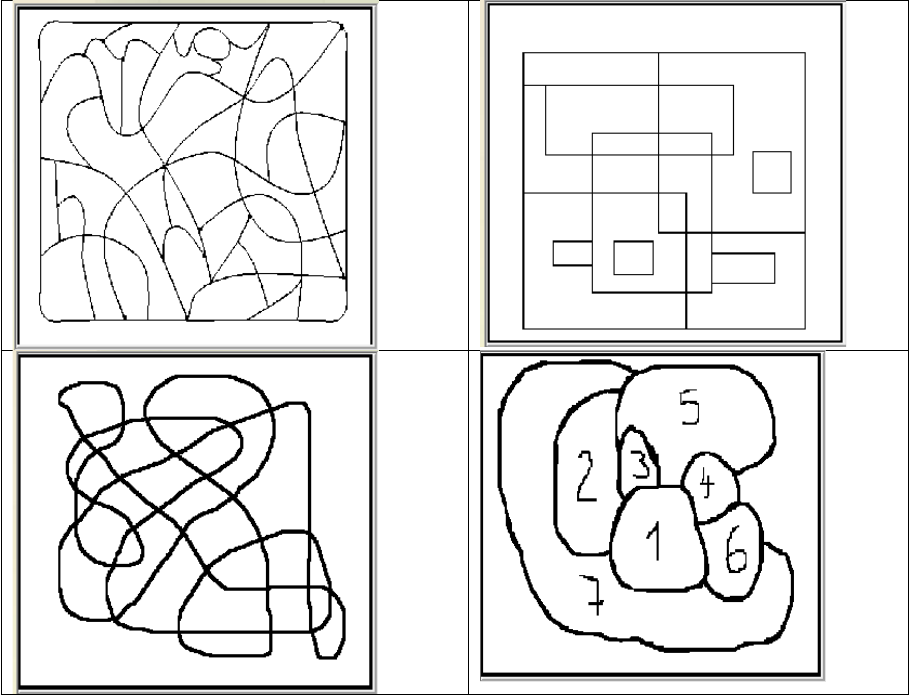
P1: 20, 14 (timp de lucru 34)

P2: 18, 11, 5 (timp de lucru 34)

P3: 15, 10, 6, 3 (timp de lucru 34)

și timpul necesar 34





3. Problema colorării unei hărți

Fișierul date.in conține pe prima linie numărul natural n, reprezentând numărul de țări

figurate pe o hartă, iar pe următoarele linii perechi de forma x y, unde x și y sunt țări

care se învecinează pe această hartă. S-a demonstrat că orice hartă poate fi colorată

cu 4 culori.

Să se determine o modalitate de a colora harta cu un număr minim de culori dintre

cele 4 posibile, astfel încât două țări vecine să aibă culori diferite. Se consideră că

două țări care au doar un punct comun pot avea aceeași culoare. Datele se scriu în

fișierul date.out. Notați timpul necesar executării sursei și comparați soluția optimă și

soluția propusă de metodele indicate.

(Indicație: metoda 1 - se parcurg țările și, pentru fiecare, se caută prima culoare

permisă (nicio vecină să nu aibă culoarea respectivă); metoda 2 – se parcurg culorile

și, pentru fiecare, se colorează cu aceasta toate țările pentru care culoarea este

permisă)

Obs. pentru fig. 4 niciuna dintre metode nu obține soluție, deși există una!

4. Problema săriturii calului

Fișierul date.in conține pe prima linie trei numere naturale n și l0 și c0, reprezentând

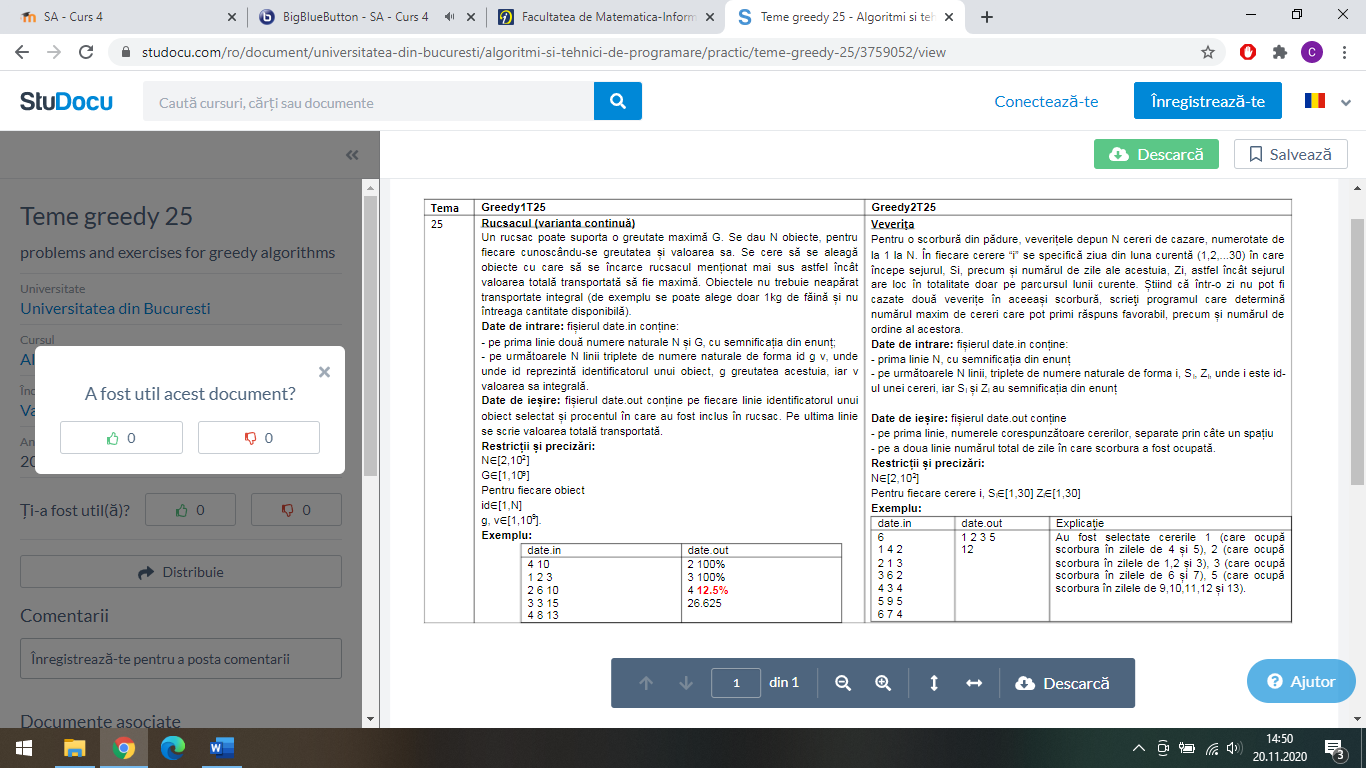
dimensiunea unei table de șah, respectiv linia și coloana acesteia pe care se află

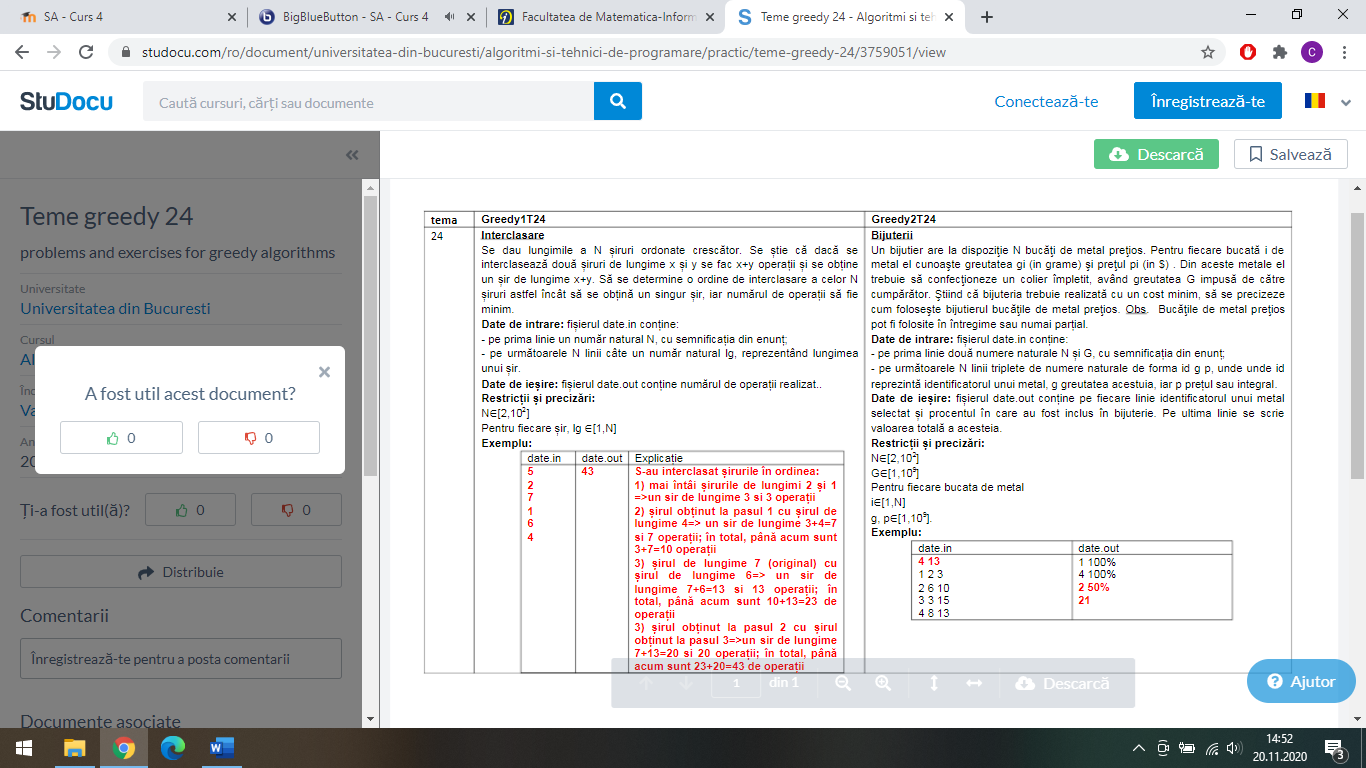
inițial un cal.

Să se determine o succesiune de mutări ale calului, astfel încât el să parcurgă toate

căsuțele tablei, trecând o singură dată prin fiecare dintre acestea. Datele se scriu în

fișierul date.out. Notați timpul necesar executării sursei și comparați soluția optimă și





Ce reprezintă Tehnica Greedy?  

Tehnica Greedy presupune că problemele pe care trebuie să le rezolvăm au următoarea structură:

* se dă o mulțime A={a1, a2, ..., an} formată din n elemente;
* se cere să determinăm o submulțime B, care îndeplinește anumite condiții pentru a fi acceptată ca soluție.

Necesitatea studierii metodei

       Tehnica Greedy este o metodă care trebuie studiată în cursul de liceu la informatică, deoarece permite rezolvarea mai multor probleme care ar fi mai greu de rezolvat prin alte metode. 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.

Exemplu de problemă

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

**Rezolvare:**

var a:matrice; c:tablou; f:text;  
loc,n,g,i,j:integer; max,castig,dg:real;  
begin  
assign (f,'rucsac.txt'); reset (f);  
readln(f,n,g);  
for i:=1 to n do  
begin readln(f,a[i,1],a[i,2]);  
a[i,3]:=a[i,1]/a[i,2]; a[i,4]:=0;  
end;  
{sortam tabloul dupa eficienta}  
for i:=1 to n-1 do  
begin max:=a[i,3];loc:=i;  
for j:=i+1 to n do  
if a[j,3]>max then begin max:=a[j,3]; loc:=j; end;  
c:=a[i]; a[i]:=a[loc]; a[loc]:=c;  
end;  
{Aflam cat din fiecare obiect se pune in rucsac si calculam castigul}  
castig:=0;  
i:=1; dg:=g;  
writeln ('greutatea ','costul ','eficienta ','rucsac');  
while (i<=n) and (dg>0) do  
begin;  
if dg>=a[i,2]  
then begin castig:=castig+a[i,1];  
dg:=dg-a[i,2]; a[i,4]:=1;  
end  
else begin castig:=castig+dg\*a[i,3];  
a[i,4]:=dg/a[i,2];dg:=0;  
end;  
writeln (a[i,1]:6:2,a[i,2]:8:2,a[i,3]:12:2,a[i,4]:10:2);  
i:=i+1;  
end;  
writeln ('greutatea rucsacului este ',g-dg:0:2);  
writeln ('costul este ',castig:0:2);  
end.

Probleme din cotidian care pot fi rezolvate utilizând această metodă:

1. Vizionarea unui număr cît mai mare de spectacole într-o anumită perioadă de timp.
2. Plata unei sume în monede de mai multe tipuri.
3. Alegerea cît mai multor feluri de mîncare întîlnite în lista unui meniu din restaurant.
4. Alegerea unui număr maxim de obiecte care pot încăpea într-un rucsac.
5. Problema selectării activităților este caracteristică acestui tip de probleme, în cazul în care obiectivul este de a alege numărul maxim de activități care nu intră în conflict unele cu altele.
6. Într-un calculator Macintosh jocul *Crystal Quest*are obiectivul de a colecta cristale. Jocul are un mod demo, în acest caz jocul foloseste un algoritm greedy pentru a merge la fiecare cristal. Inteligența artificială însă nu ține cont de obstacole, deci modul demonstrativ de multe ori se termină repede.

Utilizarea metodei în alte țări ale lumii:

       În alte țări, tehnica Greedy este utilizată cu rolul de a construi soluția optimă pas cu pas, la fiecare pas fiind selectat în soluție elementul care pare optim la momentul respectiv. Problemele la care este aplicată metoda respectivă sunt probleme de optimizare, majoritatea constînd în determinarea unei submulţimi *B* a unei mulţimi *A* cu *n* elemente care să îndeplinească anumite condiţii pentru a fi acceptată.

Concluzie:

       Metoda Greedy este foarte eficientă atunci când dorim să aflăm rezultatul optim în cît mai scurt timp posibil, deoarece algoritmii sunt polinomiali. Cu regret, aceasta poate fi aplicată numai atunci cînd din enunțul problemei poate fi dedusă regula care asigură selecția directă a elementelor necesare din mulțimea dată.

