

Facultatea de Matematică și Informatică
Secția Informatică

Didactica informaticii
-proiect de grup-

Metode de programare: GREEDY

Nume student, grupa, adresa email...

Nume student, grupa, adresa email...

Anul universitar

Compararea surselor bibliografice

Sursele selectate:

1. LICA Dana, PAȘOI Mircea - „*Fundamentele programării; Culegere de probleme – Pascal și C++ pentru clasa a XI-a*” – Editura L&S Soft, București, 2006;
2. MILOȘESCU Mariana – „*Manual pentru clasa a XI-a Informatică – varianta C++*” – Editura Didactică și Pedagogică, București, 2006
3. OPRESCU Daniela, BEJAN IENULESCU Liana - „*Manual pentru clasa a XI-a Informatică - varianta Pascal*” – Editura Niculescu, București, 2006

Prezentarea generală a surselor citate mai sus:

1. Culegerea de probleme „*Fundamentele programării*”, scrisă de doamna Dana Lica este deosebit de utilă pentru toți cei care vor să aprofundeze noțiunile teoretice, prin rezolvarea unei game variate de exerciții (teste cu alegere multiplă și duală, probleme rezolvate, probleme propuse), prezentate gradual din punctul de vedere al dificultății.

Cartea se adresează atât elevilor de la specializarea matematică – informatică, cât și acelor de la matematică - informatică, intensiv informatică, cuprinzând probleme dintre cele mai ușoare, menite să fixeze chestiunile teoretice elementare, alături de exerciții dificile, propuse pentru concursurile județene de informatică. De asemenea, materialul poate fi parcurs de elevi indiferent de limbajul de programare în care lucrează la clasă (*Pascal* sau *C/C++*), întrucât toate problemele rezolvate sunt implementate în ambele limbaje. Rezolvarea problemelor din această culegere garantează aprofundarea și însușirea tuturor noțiunilor, dar și deschiderea spre strategii originale de abordare a problemelor, care să pună în valoare elevii cu potențial.

Prezentăm, în continuare, argumente pro și contra referitoare la expunerea temei tehnicilor de programare.

<i>Argumente pro</i>	<i>Argumente contra</i>
<ul style="list-style-type: none"> • Culegerea este un instrument util în evaluarea elevilor, venind în completarea manualelor cu cel puțin 15 probleme propuse spre rezolvare pentru fiecare metodă, de la probleme simple, la unele dificile, de concurs. 	<ul style="list-style-type: none"> • Nu sunt abordate separat noțiuni teoretice, acest lucru lăsându-se în seama manualelor.
<ul style="list-style-type: none"> • Pentru fiecare metodă de programare, există o serie de probleme rezolvate didactic, în detaliu, cu explicarea noțiunilor teoretice folosite, a modalității de codificare a datelor, a ideii generale de abordare și a procedurilor folosite. 	<ul style="list-style-type: none"> • Pentru unele probleme nu există deloc indicații de rezolvare, iar cei care nu reușesc să vină cu o idee salvatoare nu au altă soluție decât consultarea profesorului de la clasă.
<ul style="list-style-type: none"> • Se oferă strategii de lucru, dar și soluții interesante pentru diverse probleme, astfel încât elevii își pot dezvolta și latura creativă în găsirea de soluții, nu doar capacitatea de a aplica mecanic un algoritm standard. 	
<ul style="list-style-type: none"> • Fiecare exercițiu este urmat de exemple concrete de date de intrare și datele de ieșire asociate, pentru a permite elevilor să se verifice. 	
<ul style="list-style-type: none"> • Multe dintre enunțurile problemelor sunt descrise printr-o situație reală din mediul înconjurător, astfel punând elevii în situația de a traduce în limbaj matematic o formulare care nu îi duce cu gândul la un algoritm anume. 	

2. Manualul conceput sub îndrumarea doamnei Mariana Miloşescu este unul de filieră *teoretică*, profil *real*, specializare *matematică- informatică, intensiv informatică, aprobat* prin Ordinul Ministrului Educației și Cercetării. Acesta prezintă toate tehnicile de programare: atât „*Backtracking*” și „*Divide et impera*”, cât și metoda „*Greedy*” și *programarea dinamică*. Maniera de abordare este una mai amănunțită, detaliată, cu foarte multe noțiuni teoretice, însă cu destul de puține probleme propuse spre rezolvare.

Manualul cuprinde foarte multe *studii de caz*, în care se prezintă *scopul general* al părții de capitol sau al capitolului, urmat de probleme care să îl illustreze cât mai bine cu putință. De asemenea, apar și probleme date la concursurile și olimpiadele de informatică, dovadă că materialul didactic nu este potrivit decât pentru elevii pasionați de această disciplină. Prezентăm, în continuare, argumente pro și contra referitoare la expunerea temei tehnicilor de programare.

<i>Argumente pro</i>	<i>Argumente contra</i>
<ul style="list-style-type: none"> Fiecare metodă este descrisă atât informal, cât și formal, încercându-se încadrarea perfectă a fiecărei tehnici într-un cadru tipic, într-o clasă de probleme. 	<ul style="list-style-type: none"> Se prezintă unele noțiuni care depășesc cu mult programa pentru clasa a XI-a (de exemplu paragraful „<i>Generearea modelelor fractale</i>”), care presupun cunoașterea în prealabil a unor concepte nepredat, cum ar fi noțiuni de grafică în C++.
<ul style="list-style-type: none"> Apar numeroase exemple de probleme clasice, rezolvate, iar fiecare algoritm este explicat în amănunt, specificându-se fiecare caz posibil și detaliindu-se fiecare pas. Se prezintă strategii de lucru pentru o înțelegere profundă a metodelor. 	<ul style="list-style-type: none"> Manualul este destul de sărăcăcios în probleme propuse spre rezolvare individual sau ca temă de laborator, iar nivelul unora propuse este destul de ridicat (subiecte de concursuri, olimpiade).
<ul style="list-style-type: none"> Autorii folosesc scheme, tabele, reprezentări grafice ale problemelor, pentru a facilita înțelegerea enunțului și a ideii de rezolvare. 	<ul style="list-style-type: none"> Nu se realizează un feedback al cunoștințelor elevilor, prin diferite tipuri de exerciții (nu doar probleme) după fiecare tehnică.
<ul style="list-style-type: none"> Se fac demonstrații formale, matematice ale complexităților problemelor, ceea ce duce la aprofundarea însușirii noțiunilor teoretice, prin analiza timpului de execuție pentru fiecare pas. 	<ul style="list-style-type: none"> Manualul este mult prea dificil pentru copiii care poate nu sunt atât de motivați sau pasionați de informatică. O ierarhizare mai bună a problemelor, de la simplu la complex, ar fi fost benefică.
<ul style="list-style-type: none"> Pentru unele probleme se dau mai multe versiuni de rezolvare (de exemplu „<i>Quicksort</i>”), împreună cu explicații suplimentare pentru cei care vor să învețe mai mult. 	

3. Cel de-al treilea manual, apărut la Editura Niculescu în 2006 se adresează elevilor care urmează filiera *teoretică*, profilul *real*, specializarea *matematică – informatică* și filiera *vocațională*, profil *militar* *MapN*, specializarea *matematică – informatică*, fiind un manual *aprobat* prin Ordinul Ministrului Educației și Cercetării.

Acesta prevede atât noțiuni pentru formarea competențelor necesare elaborării algoritmilor (metode de rezolvare a unor clase de programe), cât și noțiuni pentru formarea competențelor de programator (tehnici de programare). S-a ales o prezentare ascendentă, progresivă din punctul de vedere al dificultății, majoritatea capitolelor deschizându-se cu o scurtă recapitulare a noțiunilor anterioare, cu scopul realizării feedback-ului. De asemenea, un lucru esențial este faptul că fiecare capitol începe printr-o scurtă prezentare a obiectivelor și a conținutului principal, introdusă prin: „În acest capitol veți învăța despre:”. În ceea ce privește exercițiile propuse, manualul este bogat în exemple rezolvate și probleme pentru laborator și studiu individual.

Prezentăm, în continuare, argumente pro și contra referitoare la expunerea temei tehnicilor de programare. Fiind vorba despre un manual de informatică neintensiv, acesta cuprinde doar două metode de programare, și anume „*Backtracking*” și „*Divide et Impera*”, fără a fi discutate și tehnica „*Greedy*” sau *programarea dinamică*.

<i>Argumente pro</i>	<i>Argumente contra</i>
<ul style="list-style-type: none"> Se analizează unele exemple intuitive și ușor de înțeles de către copii, detaliindu-se rezolvările și explicându-se în cuvinte ideea de pornire și tratare al problemei; se folosesc tabele care ilustrează situația variabilelor la fiecare pas al algoritmilor. 	<ul style="list-style-type: none"> Se începe prin prezentarea metodei „Divide et Impera”, înaintea metodei „Backtracking”, care este cea mai inefficientă tehnică de programare din punctul de vedere al timpului de execuție. Logic ar fi fost să se discute inițial metoda cea mai rudimentară, care practic generează toate posibilitățile de formare a soluțiilor și apoi celelalte, care presupun o îmbunătățire a complexității.
<ul style="list-style-type: none"> Autorii folosesc scheme, desene menite să sporească înțelegerea noțiunilor, a cerințelor problemelor și ideii de rezolvare a acestora 	<ul style="list-style-type: none"> Ambele metode încep mai întâi cu exemple și apoi cu prezentarea și definirea conceptelor și a mecanismelor specifice. Totuși, ar fi fost mai bine din punct de vedere didactic dacă s-ar fi inversat aceste două subcapitole, pentru a implica activ elevii la discutarea exemplelor și a-i ajuta să fixeze mai bine teoria.
<ul style="list-style-type: none"> Gama de probleme propusă este diversificată și bogată, propunându-se un număr considerabil de probleme spre rezolvare. Apar și exerciții de tip grilă care vin să ajute la aprofundarea înțelegerii noțiunilor teoretice. 	<ul style="list-style-type: none"> Nu se face o recapitulare finală din toate tehnicile învățate astfel încât profesorul să poată să își dea seama dacă elevii au înțeles noțiunile și știu ce metodă să aleagă în funcție de problemă sau doar au aplicat mecanic niște algoritmi oarecum standard. Manualul se termină brusc cu prezentarea unor „norme de elaborarea a unui proiect”.

	<i>Backtracking</i>	<i>Divide et Impera</i>	<i>Metoda Greedy</i>	<i>Programarea dinamică</i>
LICA Dana, PAȘOI Mircea - „Fundamentele programării; <i>Culegere de probleme</i> – Pascal și C++ pentru clasa a XI-a” – Editura L&S Soft, București, 2006;	Cele mai comune probleme (elemente de combinatorică) explicate didactic, împreună cu alte exemple semnificative. 23 de probleme propuse spre rezolvare, de diferite dificultăți, care implică exersarea și aprofundarea noțiunilor predate în manuale.	De asemenea probleme tratate didactic, cu evidențierea particularităților metodei de programare și multe probleme propuse spre rezolvare individuală sau în cadrul orelor de laborator. Problemele sunt menite atât să fixeze noțiunile de teorie elementare, cât și să dezvolte ideile elevilor.	Poate materialul cel mai complet din punctul de vedere al diversității problemelor, atât rezolvate, cât și propuse. Deși algoritmul este unul standard, în unele cazuri este esențială intuirea soluției și a structurilor de date necesare. Această culegere relevă cel mai bine acest aspect.	Se prezintă diverse modalități de tratare a problemelor, multe dintre ideile de lucru presupunând găsirea unor recurențe care nu se observă ușor. Cartea cuprinde multe exerciții rezolvate, prin a căror parcurgere elevul nu dobândește numai niște cunoștințe de programare dinamică, ci și curajul de a trata problemele din mai multe perspective, axându-se pe idei originale și nu pe algoritmi memorati mecanic.
MILOȘESCU Mariana – „Manual pentru clasa a XI-a Informatică – varianta C++” – Editura Didactică și Pedagogică, București, 2006	Explicat deosebit de amănunțit, destulă teorie, dar puține probleme propuse, față de celălalt manual sau de culegere, oferă mult mai multe exerciții.	Se tratează foarte riguros, spre deosebire de primul manual care se așază mai mult pe aplicații, acesta cuprinde foarte multă teorie și demonstrații ale complexităților problemelor de divide et impera.	Și această metodă apare explicată atât informal, cât mai ales formal, cu probleme sugestive și explicate foarte amănunțit, de la strategia Greedy utilizată, la descrierea structurilor de date folosite și a procedurilor și funcțiilor.	Este cel mai bine explicată tehnică față de toate celelalte surse. Totul este descris de asemenea în detaliu, cu tratarea problemelor clasice de programare dinamică și destul de puține exerciții.
OPRESCU Daniela- „Manual pentru clasa a XI-a Informatică - varianta Pascal” – Editura Niculescu, București, 2006	Explicat în detaliu, atât forma nerecursivă, cât și cea recursivă; numeroase probleme, foarte diversificate, de la cele clasice, la unele foarte interesante.	Prima metodă prezentată, cu probleme sugestive și intuitive, și cu destule probleme propuse spre rezolvare individuală.	Nu se tratează în acest manual.	Nu se tratează în acest manual.

Colegiul : ...

Profesor :

Disciplina : Informatică

Clasa : A XI-a

Profil : Matematică – Informatică

Anul școlar:

Proiect de lecție:

Metoda de programare Greedy

Unitatea de învățare: Tehnici de programare

Titlul lecției: Predare: Introducere în metoda de programare Greedy

Timpul alocat lecției: 50 minute

Mediul de instruire: Laboratorul de informatică

Obiectivele didactice operaționale:

- Ob. 1 – elevul să definească strategia Greedy;
- Ob. 2 – elevul să identifice avantajele și dezavantajele utilizării metodei Greedy;
- Ob. 3 – elevul să enunțe structura generală a unui algoritim Greedy;
- Ob. 4 - elevul să implementeze algoritmul Greedy într-un limbaj de programare la alegere, pentru probleme concrete, propuse de profesor.

Competențele generale:

- C1 – Identificarea datelor care intervin într-o problemă și aplicarea algoritmilor fundamentali de prelucrare a acestora;
- C2 – Elaborarea algoritmilor de rezolvare a problemelor;
- C3 – Implementarea algoritmilor într-un limbaj de programare;

Competențele specifice:

- C2.1 – Analiza problemei în scopul identificării metodei de programare adecvate pentru rezolvarea problemei;
- C2.2 – Aplicarea creativă a metodelor de programare pentru rezolvarea unor probleme intradisciplinare sau interdisciplinare;
- C2.3 – Analiza comparativă a eficienței diferitelor metode de rezolvare a aceleiași probleme și alegerea unui algoritm eficient de rezolvare a unei probleme;

Metode și procedee:

- Conversația euristică;
- Conversația examinatorie;
- Explicația;
- Demonstrația;
- Problematizarea;
- Exercițiul.

Mijloace de învățământ:

- Aplicația Power Point;
- Laboratorul dotat cu calculatoare;
- Tabla;
- Soft-uri : Turbo Pascal, Free Pascal, Microsoft Visual Studio 2008
- manual MILOȘESCU Mariana – „*Manual pentru clasa a XI-a Informatică – varianta C++*” –Editura Didactică și Pedagogică, București, 2006.

Obiectiv	Etapele instruirii	Activitatea profesorului	Activitatea elevilor	Metode și procedee	Mijloace de învățământ	Timp
	1. Moment organizatoric	- cere elevilor pornirea calculatoarelor - pregătirea aplicației Power Point	- pornirea calculatoarelor - așezarea în bănci - pregătirea caietelor	Explicația		1'
	2. 3. Captarea atenției; verificarea cunoștințelor	- anunțarea temei lecției; aceasta va rezulta ca acronim dintr-o listă de cuvinte care constituie răspunsurile date de elevi la întrebările profesorului prin care se face reactualizarea cunoștințelor din lecțiile anterioare - adresarea unor alte întrebări legate de Backtracking (definiție, utilizare, complexitate; vezi Anexa 1)	- obțin cuvântul GREEDY - notează titlul lecției în caiete - participă la discuție, dând răspunsuri clare la întrebările profesorului	Conversația euristică Conversația examinatorie	Caietul cu lecțiile precedente	5'
O1	4. Prezentarea noilor noțiuni	- definirea metodei Greedy - cere elevilor să scrie în caiet caracteristicile principale ale metodei Backtracking enunțate anterior pe o primă coloană - realizarea unei comparații între cele două metode discutate până atunci : Backtracking și Greedy - cere elevilor notarea caracteristicilor pe a doua coloană - formularea, în cuvinte, a algoritmului general Greedy - cere elevilor scrierea pe tablă a algoritmului în pseudocod - prezentarea problemelor demonstrative (găsirea submulțimii de sumă maximă și problema folosirii unei resurse), demonstrând corectitudinea (vezi Anexa 2)	- notează pe caiete definiția - notează în caiet conform cerințelor profesorului	Explicația Conversația euristică	Aplicația Power Point	2' 5'
O2			- notează caracteristicile noii metode conform cerințelor	Explicația		2'
						2'
O3			- elevul ales rezolvă cerința la tablă - notează datele considerate importante în caiete - pun întrebări legate de greutățile întâmpinate în înțelegerea problemelor	Conversația euristică Demonstrația		5' 12'

Obiectiv	Etapele instruirii	Activitatea profesorului	Activitatea elevilor	Metode și procedee	Mijloace de învățământ	Timp
O4	5. Feedback	<ul style="list-style-type: none"> - cere elevilor implementarea soluției unei probleme, la alegere, în limbajul de programare dorit - verifică dacă toți elevii au reușit rezolvarea problemei alese 	- pornesc soft-ul preferat și încearcă implementarea problemei dorite	Exercițiul	Calculatorul	8'
						2'
	6. Fixarea notiunilor	<ul style="list-style-type: none"> - cere elevilor să enunțe metoda Greedy, să formuleze algoritmul, și să enunțe cel puțin o asemănare și o diferență față de metoda Backtracking 	- răspund la întrebările profesorului	Conversația euristică Conversația examinatoare		5'
	7. Încheierea activității	Tema pentru acasă: <ul style="list-style-type: none"> - să citească din manualul [2] de la pag 59 la pag 65 - să găsească o problemă care nu poate fi rezolvată optim prin metoda Greedy - să implementeze ultima problemă prezentată în Anexa 2 - opțional, să rezolve problema 4 din Anexa 2. - să repete notiunile legate de subprograme 	- notează tema pentru acasă	Problematizarea Explicația		1'

Anexa 1

Y												

Linia 1: Tehnica Backtracking se bazează pe tuturor soluțiilor. Ce cuvânt completează corect enunțul? (R: generarea)

Linia 2: În afară de varianta iterativă, ce metodă mai este folosită pentru implementarea unui algoritm Backtracking? (R: recursivă)

Linia 3: Ce complexitate generală are algoritmul Backtracking? (R: exponențială)

Linia 4: În fiecare a algoritmului Backtracking se contruiește o soluție parțială. Ce cuvânt completează corect enunțul? (R: etapă)

Linia 5: Care este prenumele matematicianului american care a inventat termenul de „backtrack”? (R: Derrick [Lehmer])

Anexa 2

Tehnica Greedy – suport teoretic

Definire:

Tehnica Greedy este reprezentată de o strategie care duce constructiv la soluția finală, adăugând treptat soluții locale. O definiție generală a acestei metode ar putea fi formulată astfel: dându-se o mulțime finită A , trebuie determinată o submulțime $S \subset A$ care să îndeplinească anumite condiții. Această metodă furnizează o soluție unică, reprezentată de mulțimea S (un vector $= \{x_1, x_2, \dots, x_n\}$), în general fiind și soluția optimă.

Analiza algoritmilor (Backtracking – Greedy) :

Comparativ cu metoda Backtracking, metoda Greedy poate fi privită ca o particularizare a acesteia, în care se renunță la mecanismul de întoarcere. În plus, tehnica Greedy oferă o singură soluție obținută într-un timp mult mai bun: timpul de calcul exponențial (de la Backtracking unde putem genera toate submulțimile lui A în timp $2^{|A|}$ și alege apoi soluția care convine) este redus la timp de calcul polinomial. De remarcat este faptul că obținerea unei soluții optime nu este suficientă, fiind necesară și o demonstrație a acestui fapt.

Algoritm general :

Construcția soluției:

- Pas 1: se inițializează mulțimea soluțiilor S la mulțimea vidă ($S = \emptyset$);
- Pas 2: se alege, după mecanismul specific problemei, un element x din A ;
- Pas 3: se verifică dacă elementul x poate fi adăugat la mulțimea soluțiilor S ; dacă da, atunci va fi adăugat ($S = S \cup \{x\}$);
- Pas 4: se revine la pasul 2 dacă mulțimea soluțiilor este încă incompletată.

Observații: NU întotdeauna există un algoritm de tip Greedy care găsește soluția optimă!

Probleme pentru care Greedy obține soluția optimă :

1. Dându-se o mulțime de n numere întregi A să se determine submulțimea de sumă maximă S .

Exemplu: pentru $A = \{-4, 5, 6, -10, -9, 12, 5, -1\}$ soluția va fi $S = \{5, 6, 12, 5\}$

Soluție: Evident, un număr din lista A va fi adăugat la soluția S dacă este pozitiv, deoarece numerele negative nu vor face altceva decât să scadă suma curentă. Așadar, mulțimea soluțiilor este formată din elementele pozitive ale mulțimii A .

Implementare:

$k \leftarrow 0$

for $i = 1, n$

if $A[i] > 0$

$k \leftarrow k+1$

$S[k] \leftarrow A[i]$

write(S)

Optimalitate: În cazul acestei probleme, demonstrarea optimalității este banală, întrucât este evident faptul că dacă se elimină elementele negative dintr-o mulțime, suma celor rămase este maximul posibil.

2. Să se repartizeze optim o resursă (de exemplu o sală de spectacole, o sală de conferințe, o sală de sport) mai multor activități (spectacole, prezentări, respectiv meciuri) care concurează pentru a obține resursa respectivă.

Explicații suplimentare : Mulțimea activităților A are cardinalul n . Fiecare activitate are un timp de începere t_i și un timp de terminare f_i , unde $t_i < f_i$ și ocupă resursa în intervalul de timp $[t_i, f_i]$. Considerăm că două activități i și j sunt compatibile dacă intervalele lor de ocupare $[t_i, f_i]$ și $[t_j, f_j]$ sunt disjuncte ($f_i \leq t_j$ sau $f_j \leq t_i$). Problema cere găsirea unei mulțimi maximale de activități compatibile. Așadar, condiția care trebuie verificată de S este ca toate activitățile să fie compatibile între ele și, în plus, S să conțină maximul de elemente care îndeplinesc această condiție.

Exemplu : pentru $A = \{(2,3) ; (1,4) ; (3.5,10) ; (3.5,11) ; (7,9)\}$ și $n = 5$ avem $S = \{(2,3) ; (3.5,6) ; (7,9) \}$

Soluție : Inițial vom ordona toate activitățile din A crescător după timpul de terminare, urmând să parcurgem mulțimea de activități anterior ordonată selectând, inițial, prima activitate și, apoi, la fiecare pas, activitatea compatibilă cu ultima activitate aleasă.

Implementare :

$A \leftarrow \text{sort}(A)$

$S \leftarrow A[1]$

$uas \leftarrow 1$

for $ac = 2, n$

 if $A[t_{ac}] \geq S[f_{uas}]$

$S \leftarrow S \cup \{ac\}$

$uas \leftarrow ac$

write(S)

Observații :

- Funcția sort întoarce elementele vectorului A sortate crescător după timpul de terminare
- In variabila uas vom reține, la fiecare pas, ultima activitate selectată

Optimalitate(corectitudine) :

- Considerând, pentru simplitate, elementele activităților din A ordonate crescător după ora de terminare, este evident că prima activitate din A va face parte din mulțimea soluțiilor(dacă ea nu aparține soluțiilor, activitatea din S cu cel mai mic timp de terminare va putea fi înlocuită cu succes de activitatea 1 din A, deoarece aceasta are timpul de terminare minim și nu poate afecta desfășurarea celorlalte activități)
 - Considerând A' mulțimea activităților care încep după ce se termină 1, putem ajunge la concluzia că S soluție optimă pentru $A \leftrightarrow S' = S - \{1\}$ este soluție optimă pentru A'(prin reducere la absurd).
3. Să se ocupe optim un mijloc de transport (de exemplu, un rucsac, un autocamion) care are o capacitate maximă de ocupare (poate transporta o greutate maximă G) cu n obiecte, fiecare obiect având greutatea g_i și un profit obținut în urma transportului c_i , iar din fiecare obiect se poate lua orice fracțiune a sa.

Explicații suplimentare : Mulțimea A este formată din cele n obiecte. Considerăm că fiecare obiect i are o eficiență a transportului e_i reprezentând profitul pentru o unitate de greutate (c_i / g_i). Problema cere determinarea unei mulțimi S astfel încât eficiența transportului să fie

maximă. Mulțimea S este formată din obiectele care vor ocupa mijlocul de transport, iar condiția pe care trebuie să o îndeplinească elementele mulțimii S este ca, prin contribuția adusă de fiecare obiect la eficiența transportului, să se obțină o eficiență maximă, iar greutatea obiectelor/fracțiunilor de obiect selectate să totalizeze o greutate egală cu G .

Exemplu : pentru $A = \{(20,3) ; (12,4) ; (5,10) ; (10,9)\}$, unde (x,y) reprezintă greutatea, respectiv costul obiectului și $G = 30$ avem $S = \{(20,3) ; (10,4)\}$.

Soluție : Dacă suma greutăților lui A este mai mică decât G atunci încărcăm toate obiectele. Din motive clare putem presupune că $g_1 + \dots + g_n > G$.

Conform strategiei Greedy, ordonăm obiectele descrescător după eficiență: $e_1 = c_1 / g_1 \geq \dots \geq e_n = c_n / g_n$.

Algoritmul constă în încărcarea obiectelor în mijlocul de transport în această ordine, fără a depăși greutatea G (ultimul obiect poate fi eventual încărcat parțial).

Implementare:

$A \leftarrow \text{sort}(A)$

$G_d \leftarrow G$

for $i = 1, n$

 if $g_i \leq G_d$

$S[i] \leftarrow A[i]$

$G_d \leftarrow G_d - g_i$

 else

$S[i] \leftarrow (G - G_d, c_i)$

 stop

write(S)

Optimalitate : Temă / Idei generale (fără demonstrație completă)