SEMINAR 14: METODA GREEDY ŞI PROGRAMARE DINAMICĂ

METODA GREEDY

1. Se dau o serie de activităţi care trebuie realizate de o persoană într-o zi dată. Fiecare activitate este caracterizată de ora de început şi ora de sfârşit a ei. Se cere determinarea numărului maxim de activităţi care pot fi realizate, presupunând că persoana nu poate lucra la două activităţi în acelaşi timp.

Exemple:

→ criteriu: se alege activitatea care se termină cel mai devreme

Activitate	1	2	3
Ora de începere	2	5	7
Ora de finalizare	6	8	12

→ alte criterii

• durata cea mai scurtă

2 6 7 12

5 8

s-ar alege aceasta ⇒ nu conduce la soluţia optimă

Soluţie:

A_1, A_3
2 7
6 12

Activitate	1	2	3	4	5	6	7	8
Ora de începere	9	12	8	10	16	14	20	19
Ora de finalizare	11	13	10	12	18	16	22	21

Soluţie:

A_3	A_4	A_2	A_6	A_5	A_8
8	10	12	14	16	19
10	12	13	16	18	21

• activitatea cu cele mai puţine suprapuneri

nu conduce la sol. optimă

2. Se cere ocuparea optimă a unui camion care poate transporta o greutate maximă **G** cu *n* obiecte, fiecare obiect având asociată greutatea g_i şi un profit asociat, p_i. Din fiecare obiect se poate lua o fracţiune.

Exemple:

Criteriu: se alege obiectul cu valoarea cea mai mare pentru raportul profit/greutate

Obiect	1	2	3
Greutate G	10	20	30
Profit P	60	100	120
P/G:	6	5	4

G = 50

Soluţie

Item			Fracţiune	Profit
Nr	Greutate	Profit		
1	10	60	1	60
2	20	100	1	100
3	30	120	20/30=2/3	2/3·120=80
			Profit total	240

Obiect	1	2	3	4	5
Greutate	5	4	4	8	10
Profit	10	20	10	10	22

G = 20

Soluție

Item			Fracțiune	Profit
Nr	Greutate	Profit		
			Profit total	

1. PROGRAMARE DINAMICĂ

a = lista inițială
$i_1, i_2, i_3, \ldots, i_k$ = indexes/poziții din listă
a_{i_j} = elem. de la poz i_j din listă
⇒ sublista crescătoare este:

1. *Se dă o listă de numere. Să se determine lungimea sublistei crescătoare de lungime maximă și elementele acesteia.

$a_{i_1}, a_{i_2}, a_{i_3}, \ldots, a_{i_k}$ cu
$i_1 < i_2 < i_3 < \ldots < i_k$ și
$a_{i_1} \le a_{i_2} \le \ldots \le a_{i_k}$

Exemplu:

Index	0	1	2	3	4	5
a	1	-2	3	2	4	4
L	$\boxed{4}$ MAX	4	3	3	2	1
P	2	2	4	4	5	-1

L[i] = lungimea maximă a șirului care începe cu elementul a[i]
P[i] = indicele următorului element din sublista de lungime maximă care începe cu elementul a[i]

Soluție:

N = lungimea listei de numere

Relația de recurență folosită:

$$L[i] = \begin{cases} 1 & \text{dacă } i = N-1 \\ 1 + MAX(L[j]) & \text{dacă } a[i] \le a[j], \text{ pentru } i < j < N \end{cases}$$

sublista crescătoare de lungime max are lungimea 4 și începe la poziția 0

sublistă = [], P

0	1	2	3	4	5
2	2	4	4	5	-1

0 = poziția maximului din L

start here: sublista.append(a[0])
go to pos indicated by P[0]
sublista.append(a[2]

...

	0	1	2	3	4	5	6	7	8	9	10	11
a	9	5	1	-9	3	4	8	2	7	12	18	3
L	3	4	$\boxed{6}$	$\boxed{6}$	5	4	3	4	3	2	1	1
P	9	6	4	4	5	6	9	8	9	10	-1	-1

MAX LUNGIME

MAX SUBLISTĂ: a[2], a[4], a[5], a[6], a[9], a[10]

(v1) 1 3 4 8 12 18

2. Într-un rucsac se poate transporta o greutate maximă **G** și există **n** obiecte, fiecare având greutatea g_i și un profit asociat, p_i. Obiectele nu pot fi fracționate. Să se determine profitul maxim care poate fi obținut prin obiectele transportate în rucsac.

Exemplu:

→ exemplu pt care Greedy cu alegere candidat ca și la fractional knapsack nu funcționează:

O	1	2	3
G	2	2	3
P	10	10	18
	5	5	6

N=3
G=4

O3 → P_TOTAL=18

O1+O2 → P_TOTAL=20

N = 4
G = 10

Obiect	1	2	3	4
Greutate G	2	3	6	4
Profit P	20	40	50	45
P/G	10	13,3	8,33	11,25

w →

i ↓

M		Greutate disponibilă										
		0	1	2	3	4	5	6	7	8	9	10
Obiect	1											
	2											
	3											
	4											SOL

*insert a row filled with 0s

Relația de recurență folosită:

$$M[i][w] = \begin{cases} 0 \text{ dacă } i=0 \text{ sau } w=0 \\ M[i-1][w] \text{ dacă } G_i > w \text{ pentru } i \geq 1 \\ MAX(M[i-1][w], P_i + M[i-1][w-G_i]) \text{ dacă } G_i \leq w,\ i \geq 1 \end{cases}$$

Completați tabelul pentru:

M[i][j] = profitul maxim obținut în subproblema i pentru greutatea disponibilă j

N = 3
G = 6

M[i][0] = profitul maxim al unui rucsac cu greutatea disponibilă 0 (care nu permite adăugare de obiecte)

Obiect	1	2	3
Greutate	1	2	3
Profit	2	15	40

M[0][j] = profitul maxim al unui ~~obiect~~ rucsac gol cu greutate disponibilă j

		Greutate disponibilă						
		0	1	2	3	4	5	6
Obiect	1							
	2							
	3							

→ pe fiecare linie i se determină profitul maxim pe baza valorilor maxime ale profiturilor care s-au determinat pentru subproblemele anterioare

→ profiturile sunt maxime pe fiecare linie.

3. Se consideră un triunghi de numere naturale a_{ij} cu n linii. Pornind de la numărul din linia 1, mergând în jos până la linia n, să se determine **o selecție de elemente** astfel încât suma elementelor să fie maximă. Trecerea la linia următoare se poate face doar mergând în jos, direct sau pe diagonală (la dreapta).

Triunghiul inițial:

	1	2	3	4
1	5			
2	4	2		
3	5	4	3	
4	4	3	2	5

Matricea sumelor maxime

MAX SUM

	1	2	3	4
1	18			
2	13	10		
3	9	7	8	
4	4	3	2	5

Matricea direcției de deplasare

	1	2	3	4
1	1			
2	1	2		
3	1	1	2	
4	0	0	0	0

MAX SEQUENCE POSITIONS

Relația de recurență folosită:

→ rezolvare inclusă în cod

4. Se dă o țeavă de lungime N și o listă de prețuri, unde prețuri[i] este prețul aferent țevii de lungime i. Să se determine cea mai mare valoare care se poate obține prin tăierea țevii și vânzării pieselor aferente.

Exemplu:

N = 8

Lungime	1	2	3	4	5	6	7	8
Preț	1	5	8	9	10	17	17	20

Valoarea maximă: 22
Explicație: Valoarea 22 se obține prin tăierea țevii în piese de lungime 2 și 6, i.e., 5 + 17 = 22.

N = 8

Lungime	1	2	3	4	5	6	7	8
Preț	3	5	8	9	10	17	17	20

Valoarea maximă: 24
Explicație: Valoarea 24 se obține prin tăierea țevii în 8 piese de lungime 1, 8*3=24.

Relația de recurență folosită:

→ rezolvare inclusă în cod