PROBLEMA CU MOȘ CRĂCIUN

07.04.2023

PII

AA

Seminar 4

Problema rucsacului

$O = \{o_1, \dots o_m\}$ → mulț. de obiecte

- fiecare o_i are o greutate g_i și o valoare val_i
- avem un rucsac de capacit C, vrem ca val. să fie cât mai mare a.î. să nu dep. capacit. rucsacului.

<=> Vrem să sel. o submulț. $S \subseteq O$ a.î. suma greut $\leq C$, și suma val. să fie maximă

formalizare matematică: $\sum_{o_i \in S} g_i \leq C \quad (1)$

$\sum_{o_i \in S} val_i \to max \quad (2)$

O soluție care respectă 1 s.n. soluție candidat.

— — respectă 2 s.n. soluție optimă.

1. VARIANTA FRACȚIONARĂ
    - putem împărți obiectele

```
o[n], g[n], v[n]
sort (o, lambda x[i] -> v[i]/g[i])
capacitateCurentă = 0 , valoareTotală = 0;
for (i = m-1; i ≥ 0; i--)
    if (g[i] ≤ C - capacitateTotală)
        { capacitateTotală += g[i];
          valoareTotală += v[i];
        }
    else { j = i;
        break;
      }
```

```
if (capacitateTotală != c)
        valoareTotală += C - capacitateTotală * g[j]/
```

2. VARIANTA DISCRETĂ

- obiectele fie sunt luate, fie nu

d[i] = { val. max. care poate fi luată în lucru
cu obiecte care au greutate i
total?

```
d[0] = 0;
for(i=0 ; i<n ; i++)
  ~~d[g[i]] = val[i]g[i]~~
    for(j=C-g ; j>=0 ; j--)
      if(d[j]!=0)
      if(d[j] + val[i] >= d[j+g[i]])
          d[j + g[i]] = d[j] + val[i];
```

exemplu:

greut, val

$o_1 = (3,5)$

$o_2 = (1,3)$

$o_3 = (4,2)$

$o_4 = (5,6)$

C = 11

PAS 1:

d =	0	0	0	--	0
	0	1	2		11

după ce îl adaug pe o_1

d =	0	0	0	5	0	--	0
	0	1	2	3	4		11

PAS 2: îl ad. pe o_2

		3			5+3			
d =	0	3	0	5	8	0	--	0
	0	1	2	3	4	5		11

pe poz 3+1

actualizat la final în acest pas

PAS 3: se ad. o_3

val pt acea greut.

d =	0	3	0	5	8	5	0	7	10	0	0	0
greut. tot. →	0	1	2	3	4	5	6	7	8	9	10	11

PAS 4: se ad. o_4

d =	0	3	0	5	8	6	9	7	11	14	13	0
	0	1	2	3	4	5	6	7	8	9	10	11

⇒ răsp = 14.

! NU are cplx. polinomială. Are complexitate PSEUDOPOLINOMIALĂ.

cplx: O(n*c)

Alte probleme pseudopolinomiale

1. Se dă o mulț. de ~~obiecte~~ n numere naturale $\overset{n}{M} \leq 10.000$.
Să se det. dacă ∃ o submulț. cu suma egală cu M.

ex: n = 6
mulțimea = {12, 1, 3, 4, 5, 7}
M = 14
7+3+4 → soluție.

d[i] = {1, dacă ∃ suma i
 0, altfel

procedeul la fel ca la pb. rucsacului

- d[i] = 0|0|0| -- |0|
 0 1 2 -- 14
- PAS 1: 0|1|1 ... |1|0|0|
 0 1 2 ... 12 13 14
- PAS 2: 0|1|0 --- 1|1|0
 0 1 2 ... 12 13 14
- PAS 3: 0,1,0,1,1 -- ,1,1,0
 0 1 2 3 4 12 13 14
- PAS 4: 0,1,0,1,1,1,0,1,1 -- ,1,1,0
- PAS 5: 0,1,0,1,1,1,1,1,1,1,1,1,0,1,1,0
 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14
- PAS 6: 0,1,0,1,1,1,1,1,1,1,1,1,1,1,$\boxed{1}$
 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

Alte probleme de progr. dinamică

1. probl. spectacolelor, generalizată

n activități, fiecare cu timp început, sfârșit → profit

vrem să det. o submulț. de activități de profit maxim.

Sortăm vectorul de activități după timpul de sfârșit

d[i] = profitul maxim obținut organizând primele i activit.

```
for i in activități
    j = căutăm binar ultima activitate care se termină înaintea ei
    d[i] = max(d[j] + profit[i], d[i-1])
→ O(n log n)
```

2. generalizarea problemei planificării de activități

- se dă lista cu activități:
    - → deadline t_i
    - → durată l_i
    - → profit p_i
- trebuie să găsim o submulț. ord. de activ. pt. a obține profitul maxim.

exemplu:

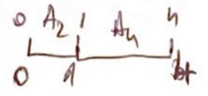
a_1 = 4 5 d_i, 3 l_i, 3 (p_i) p = 3, t = 5, l = 3

a_2 = 2 2 p = 2, t = 2, l = 1

a_3 = p = 3, t = 2, l = 2

a_4 = p = 5, t = 4, l = 3

sol. optimă : A_2 și A_4



Se sort. ~~după~~ crescător deadline.

d[i][t] = {profitul maxim, planificând primele i activități înainte de timpul t}

```
T = max{ t_i | i = 1,n }
for (i=1, i<=n; i++)

    for (t=1; t<=T; t++) {
        if (t <= t[i])
            d[i][t] = max(d[i-1][t-l[i]]+p[i],
                          d[i-1][t]),
        else d[i][t] = d[i-1][t};
```

28.04.2023

P2

Seminar 5 AA / part. 2

T.SEM → deade

1. Vouchere de vacanță → asem. cu pt. rucsacului

- sumă de bani S
- n destin. cu costul $c_1 -- c_n$
- Vrem: un mod cât mai ef. de a folosi voucherele ?
  (să maximizeze costul destinațiilor)

Se cere un alg. 1/2 aproximativ.

1/0:

```
sort(c);
cost=0;

for(i=1;i<=n;i++)
    if (cost + c[i] < S)
        cost += c[i];
    else break;
```

1 2 5 7

8

9

Fie $OPT_{1/0}$ = sol. optimă pe caz 1/0 a prob.

OPT_G = —"— var. fracționară

ALG = sol. dată de alg. nostru (pseudocod)

Fie C_s = dest. de cost maxim

Vrem să ar. că $\underbrace{\frac{1}{2} OPT_{1/0} \le ALG}$

Obs. că $OPT_{1/0} \le OPT_G$

≤ decât dacă am fi luat complet și ultimul obiect fracționat

$OPT_{1/0} \overset{\le OPT_G}{\le} \sum_{1 \le i \le k} C_i$, unde k e indicele ultimei dest. selectate

k = primul care nu mai încape la 1/0

$OPT_{1/0} \le \sum_{1 \le i \le k} C_i = \sum_{1 \le i < k} C_i + C_k$ ①

$ALG \geq C_k$ Și $ALG \geq \sum_{1 \le i < k} C_k \Rightarrow OPT_{1/0} \le ALG + ALG$

$OPT_{1/0} \le 2ALG \Rightarrow \frac{1}{2} OPT_{1/0} \le ALG$

2. Camioane și transporturi

- n colete cu greut. $w_1, w_2 \dots w_n$
- ~~w_1~~ Avem camioane de capacitate G.
- știm că $w_i \le G$, $\forall i = \overline{1,n}$ nr. min. camioane
- Avem urm. algoritm:

  „Iterăm prin mult. obiectelor, până dăm de unul care nu încape, iar când nu mai încape, luăm urm. camion."
- <u>Cerință</u>: 5p 1) dați un ex. care să arate că sol. nu e opt.

  10p 2) Ar. că sol. e 1/2 aprox.

1) cap. 6

5 6 1 ~~5~~, 6 → OPTIM ar fi: 5 1
6
6 → 3 camioane

ALG dă 5
6 → 4 camioane
1
6

2) Fie OPT ~~1/0~~ = sol. optimă a probl.

~~OPT$_G$ = sol. problemei dacă am putea fracționa fiecare obiect~~

ALG = sol. dată de program.

G[i] = încărcătura din camionul i

Avem 2 cazuri
I numărul de camioane folosit e par (ALG = 2p)
II ——— " ——— impar (ALG = 2p+1)

~~I 2p camioane~~

②

< IA II part. 2

$\underline{I}$ 2 camioane

Ne uităm la perechi de câte 2 camioane:

$(G[2i+1], G[2i+2])$, $i=\overline{0,p-1}$

Obs. că $\underbrace{G[2i+1] + G[2i+2] > G}$. deci în sol. optimă, pentru fiecare pereche se fol. cel puțin un camion $\Rightarrow$ $OPT \geq p$ $/ \cdot 2$ $ALG \leq 2OPT$.

$\Rightarrow$ ~~$ALG \geq \frac{1}{2} OPT$~~

$\Rightarrow$ E oricum nevoie de minim jumătate din nr. de camioane prezis de ALG, chiar pe sol. optimă

Analog și cazul II

2) Moș Crăciun ..

- n copii
- n cadouri cu valori $val_1, val_2, \dots val_n$
- Vrem să împărțim cadourile a.î. copilul cu ~~cadoul~~ cel mai "mic" cadou să aibă val. cea mai mare. (pentru val)

Notații : OPT = sol. optimă ; ALG = sol dată de alg.

W[k] = val. totală a cadourilor pe care le primește copilul k

C[k] = lista ...

VAL ~~val~~ = suma totală a cadourilor

RESTR: $val[i] \leq$ $val_i \leq \frac{VAL}{2m}$

Cerințe : 1) propuneți un alg 1/2 aprox de cplx. $O(n \log m)$ $\xleftarrow{\quad}$ .. .. $\xrightarrow{\quad}$ 7 5 $\overset{1}{4}$ $\overset{2}{3}$

2) care e rel. dintre :

Fie k copilul cel mai "vitregit". Fie $q \neq k$ și $\neq i$ ultimul cadou primit. Care e relația dintre $W(k)$ și $W(q)-val_i$?

3) Arătați că $ALG \geq \frac{VAL}{2m}$

4) Ar. că alg de la 1 e 1/2 aprox.

5) Dați un exemplu din care să se vadă că alg. nu e optim.

③

$O(n \log n)$
priority queue

```
for (i=1; i<= ; i++)
    Q ← copilul cu W(Q) minim
    ad. val(i) la W(Q)
    ad. i la C[Q]
```

2) k - cel mai vitregit

Q ~~nu~~ nu prim. ultimul cadou $W(Q) - val_i$

pp. că $W(k) \leq W(Q) - val_i$

Așadar, k ar fi extras înaintea lui Q din structura de date ⇒ i ar fi fost adăugat lui $w(k)$, nu lui $w(Q)$ ⨳

$\not\exists W(k) > W(Q) - val_i$

$W(Q) - val_i \leq w(k)$,
$W(Q) \leq W(k) + val_i$

3) pp. prin absurd că $ALG < \frac{VAL}{2m}$

notăm cu k copilul ~~care a dat soluția din~~ cel mai vitregit

$\Rightarrow ALG = W(k)$

dar $val_i \leq \frac{VAL}{2m}$; $VAL = \sum_{1 \leq i \leq n} val_i = \sum_{Q=1} W(Q)$ fiecare copil tot să fie atrib. unui copil

$$\sum_{\substack{Q=\overline{1,m} \\ Q \neq k}} W(Q) + W(k) \leq W(k) + \sum_{\substack{Q=1 \\ Q \neq k}}^{m} (W(k) + val_i) \leq$$

$$\leq W(k) + (m-1)\left(W(k) + \frac{VAL}{2m}\right) =$$

$$= m \cdot W(k) + (m-1) \frac{VAL}{2m} < m \cdot \frac{VAL}{2m} + (m-1) \cdot \frac{VAL}{2m} < \frac{VAL}{2} + m \cdot \frac{VAL}{2m} = VAL$$

⨳

⇒ pp făcută e falsă ⇒ $ALG \geq \frac{VAL}{2m}$

~~An-1 major~~

1. val tot = suma val

toti cop. - k + cop. k

$W(k)\ val_i + \frac{val}{2m}$

4) $OPT \leq \frac{VAL}{m}$

$ALG \geqslant \frac{VAL}{2m} \quad \cdot 2 \qquad \Rightarrow 2ALG \geqslant \frac{VAL}{m} \geqslant OPT$

$\Rightarrow \frac{1}{2}$ aproximativ.

5) exemplu pt mai.

10.05.2023

PII

AA - Algoritmi probabiliști

seminar săpt 11
(SEMINAR 6)

2 categorii → Las Vegas (nu știm când se termină, dar știm rez.)
Monte Carlo (știm când se termină)

	Timp de rulare	Corectitudine rezultat
Las Vegas	probabilist	probabilist
Monte Carlo	determinist	determinist

Ex.1 Se dă o fcț. random() care gener. un nr ∈ [0,1] ~~≠~~ (uniform) și vrem să generăm un nr. ~~unif~~ distrib. unif. în [a, b]

```
generează (a, b)
    nr_1 = random ()
    return((b-a)* ~~~~ nr_1 + a)
```

Ex2 Se dă o fcț. random_biased, care gener. 0 sau 1, (gen. 0 cu o probabilitate $p \neq \frac{1}{2}$). Vrem să def. o fcț. random ~~sa~~.. care gen 0 sau 1 cu $p = \frac{1}{2}$

hint: ne uităm la per. de 2 numere

00	01	10	11
p^2	$p(1-p)$	$p(1-p)$	$(1-p)^2$

```
generează2()
    nr_1 = random_biased ()
    nr_2 = random_biased ()
    if(nr_1 == nr_2)
        return generează2()
    if (nr_1 == 0 && nr_2 == 1)
        return 1;
    else return 0;
```

Ex 8 ~~Gen~~ Generați nr π

1) Care e aria cercului unitate?

2) Aria pătr. cu vf de coord (−1,−1), (−1,1), (1,−1), (1,1)

3) Dc. aleg un pct din □ →, care e prob. să fie în cercul unitate?

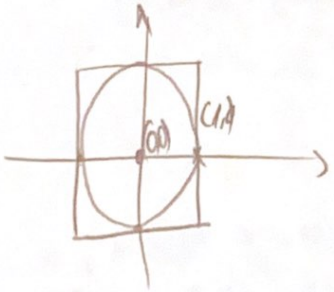
4) dat $(x,y) \in \mathbb{R}^2$, → cum det. dc. e în cercul unitate

5) se dă alg:

→ gen. un pct. $\in [-1,1] \times [-1,1]$

→ dacă pct ∈ cercului, îl contoriz.

→ se calc. nr pct contoriz / nr total.



1. $\pi \cdot r^2 = \pi \cdot 1^2 = \pi$

2. $l^2 = 2^2 = 4$

3. $P = \frac{A_\circ}{A_\square} = \frac{\pi}{4}$

4. $d[(x,y), (0,0)] < 1$

5. $\lim\limits_{n\to\infty} \frac{\text{nr puncte-cerc}}{\text{nr puncte-pătrat}} = \frac{A_\circ}{A_\square} = \frac{\pi}{4}$

6) Ce tip de alg? Monte Carlo, ptc. fixăm nr. de rulări, însa rezultatul poate fi ușor diferit

---

Alg. genetici

Progr. liniară

Alg. genetici

Se dau n obiecte → (val, prob)

$\hspace{4cm}\downarrow$ prob. să aj. intact

Se cere val. maximă a unor obiecte care să fie transportat, a.î. probabilitatea să ajungă toate intacte $\geq p$

ex: $n=3$, $p=\frac{1}{2}$

obiectele: $(4, \frac{4}{5})$, $(6, \frac{3}{5})$, $(3, \frac{4}{5})$.

Cerințe: a) Să se codifice ca pb. de alg. genetici (5p)
b) Funcția de fit (10p)

a) Codif. ca un șir de biți $x_0 x_1 \ldots x_{n-1}$, $x_i = \begin{cases} 0, & \text{ob. i nu e luat} \\ 1, & \text{altfel} \end{cases}$

b)
$$f(c) = \begin{cases} \sum_{i=0}^{n-1} p_i \cdot \cancel{x_i}\, val(i), & \overset{\text{dacă}}{\left[\prod_{i=0}^{n-1} p(i) \geq p \right.}, \text{ dacă } x_i = 1 \\ 0, & \text{altfel} \end{cases}$$

Programare liniară

Se dă o fcț. de cost cu d nedeterminate
$c = c_1 \cdot x_1 + c_2 \cdot x_2 + \ldots + c_d \cdot x_d$. Se dau m constrângeri liniare:

$$a_{11} \cdot x_1 + a_{12} \cdot x_2 + \ldots + a_{1d} x_d \leq b_1$$
$$\vdots$$
$$a_{m1} x_1 + \ldots + a_{md} x_d \leq b_m$$

ex: Se dă un proiect cu n task-uri, fiecare task poate fi procesat de una din două mașini.

$(t_i, \underbrace{x_i, y_i})$

t_i ↓ timpul

x_i, y_i → cele 2 mașini care pot rezolva task-ul

VREM: timp min. pt. a rez. toate task-urile

a) 1 → pb de progr. liniară cu nr. întregi.

2 → ——— progr. lin.

b) fol. a ca să se găsească un alg. 2-aprox

a.1 Notații: A^i_q = task-ul i este rez. de mașina q

Restricții: $A^i_{x_i} + A^i_{y_i} = 1$

$A^i_q = 0, \forall q \neq x_i, q \neq y_i$

vrem să minimizăm $\boxed{\max_{q} \sum_{i=1}^{n} A^i_q \cdot t_i}$ → NU O PUTEM CONSID. FCȚ. DE COST PTC. NU E LINIARĂ

nu e liniar

max (y_1, y_2, y_3) se adaugă z

$z \geq y_1, z \geq y_2$

$z \geq y_3$ / VREM SĂ ~~M~~ MINIMIZ. z.

adăugăm restricțiile:

$\sum_{i=1}^{n} A^i_q t_i \leq$ ~~ILP~~ iLP nedef. adăugată

și vrem să minimizăm ~~ILP~~ iLP

e liniară

→ 3 restr.

a.2. restr: $A^i_{x_i} + A^i_{y_i} = 1$, $A^i_q = 0, \forall q \neq x_i, y_i$, $0 \leq A^i_q \leq 1$, $\sum A^i_q t_i \leq$ LP

~~+ Σ~~ vrem să minimizăm LP

→ în timp polinomial

b) Alg. 2 aprox. pt. pct. a.

justificare

$T = 0{,}5$ → $\frac{1}{2}$ (vrem 2 aprox)

→ TRASHHOLD

$A^i_q = \begin{cases} 1, & A^i_q \geq T \\ 0, & \text{altfel} \end{cases}$

LP → alg. progr. lin; OPT = sol. optimă

iLP = alg. pr. lin cu 0/1; ALG = Alg. nostru

LP ≤ iLP = OPT

ALG ≤ 2 LP ≤ 2 ILP = 2 OPT ⇒ ALG ≤ 2 OPT

0,3 → nu se face // 0,7 → se face // punem lim. la luat/neluat