Curs 2. Optimizare discretă

Optimal assignment

Familie finită de variabile

$A = \{x_1, x_2, \dots, x_n\}$

$$\min_{x \in A} J(x)$$

→ nr. finit de valori: $J(x_1), J(x_2), \dots, J(x_n)$

→ selecționăm cea mai mică valoare.

Evaluarea funcției pt fiecare variabilă este fezabilă dacă:

(# variabile) × (timpul de execuție pt evaluarea funcției)

~ Rezonabil.

$n = 10^9$ elemente
Calculul $J(x_i) \to 1s$ } $\underline{10^9 s}$ prea mult

Câteodată mulțimea discretă are o structură adițională care poate simplifica procesul de optimizare.

<u>Optimal assignment</u>

<u>Repartiția optimală</u>

	P_1	P_2	P_3
J_1	100	120	80
J_2	150	110	120
J_3	90	80	110

<u>Exemplu</u>:

<u>Ex1</u>

$J_1 \rightsquigarrow P_2$	$\to$ 120 €	$\rightsquigarrow P_3$	80 €
$J_2 \rightsquigarrow P_1$	$\to$ 150 €	$\rightsquigarrow P_2$	110 €
$J_3 \rightsquigarrow P_3$	$\to$ 110 €	$\rightsquigarrow P_1$	90 €

Total 380 € > 280 €

$1 \to 2$
$2 \to 1$
$3 \to 3$

$1 \to 3$
$2 \to 2$
$3 \to 1$

O aliger reprezintă o funcție

$\sigma : \{\underline{1,2,3}\} \to \{\underline{1,2,3}\}$

$\sigma \to$ bijectivă:

$\sigma : X \to Y$ injectivitate $x \neq y \Rightarrow \sigma(x) \neq \sigma(y)$

$\to$ surjectivitate $\forall y \in Y \ \exists x \in X \ \sigma(x) = y$

$\sigma \to$ permutare a mulțimii $\{1,2,3\}$

$\sigma : \{1,2,\dots,n\} \to \{1,2,\dots,n\}$

σ permutare $\Rightarrow$ avem $n! = 1 \cdot 2 \cdot \dots \cdot n$ posibilități

De ce?

$1 \longrightarrow n$ posibilități

$2 \longrightarrow n-1$ pos.

$3 \longrightarrow n-2$ pos

$\vdots$

$n \longrightarrow 1$ pos.

$n \cdot (n-1) \cdot \dots \cdot 1 = n!$

n persoane + n joburi → n! posibilități

Alg. brută forță: testează toate posibilitățile n! etape.

n	n!
1	1
2	2
3	6
4	24
5	120
6	720
10	3628800
20	$\sim 10^{18}$ <u>foarte mare</u>

Hungarian algorithm → $O(n^3)$

n	n^3
100	10^6
1000	10^9 ⇝ rezonabil.

	P_1	P_2	P_3
J_1	100	120	80
J_2	150	110	120
J_3	90	80	110

→ dacă toți persoanele reduc costul pt job-ul J_i cu x nu se schimbă configurația optimă

→ dacă o persoană P_j își micșorează toate prețurile cu y nu se schimbă repartiția optimă.

1 Scădem val min pe fiecare linie

Pas 3 Exemplu:

$$\begin{pmatrix} 1 & 2 & 0 \\ 3 & 4 & 0 \\ 0 & 0 & 5 \end{pmatrix}$$

2 linii

trebuie mers la pasul 5

$$\begin{array}{c|ccc} & P_1 & P_2 & P_3 \\ \hline J_1 & \multicolumn{3}{c}{\multirow{3}{*}{$\begin{pmatrix} 10 & 40 & \boxed{0} \\ 30 & 0 & 10 \\ 0 & 0 & 30 \end{pmatrix}$}} \\ J_2 & & & \\ J_3 & & & \end{array}$$

$$\begin{pmatrix} 0 & 17 & 2 \\ 15 & 0 & 6 \\ 0 & 26 & 88 \end{pmatrix} \xrightarrow[L_3-2]{L_1-2} \begin{pmatrix} -2 & 15 & 0 \\ 15 & 0 & 0 \\ -2 & 24 & 86 \end{pmatrix} \xrightarrow{C_1+2}$$

$$\rightarrow \begin{pmatrix} 0 & 15 & \boxed{0} \\ 17 & \boxed{0} & 0 \\ \boxed{0} & 24 & 86 \end{pmatrix}$$

Avem nevoie de cel puțin 3 linii să acoperim zero-urile.

Alegem repartiția cu costul 0

Operații cu linii și coloane.

Pasul 1 → n operații → $O(n^2)$

Pasul 2 ≤ n operații → $O(n^2)$

⋮

Global: $O(n^3)$

Jupyter Notebook: pagină web

Celule text

Celule cod → Python
poate fi executat "pe loc"
Ctrl + Enter
Shift + Enter.

Extensia: .ipynb
Putem deschide fișierele ↔ VS Code
↔ Google colab
↔ Browser

De ce Python: - limbaj de actualitate
- ușor de folosit.
- multe module aplicative
- Open source
- Multe librării ce facilitează dezvoltarea aplicațiilor AI

Documentații online: detaliate

a = Listă

$a = [(1,2,3), (1,3,2) \ldots]$

$a[0] = (1,2,3)$

$a[0][0] = 1$

$a[0][1] = 2$

$a[0][3] = 3$

Testați conținutul codului **Opt Assignment**.