Lineární programování a kombinatorická optimalizace – příklady na 1. cvičení*

17. února 2025

1 Lineární nerovnice

Spousta praktických úloh i čistě kombinatorických lze naformulovat jako úloha lineárního programování (LP). Na úlohu LP můžeme použít známé metody a efektivně ji vyřešit. Každá úloha lineárního programování se dá převést do jednoho z těchto dvou tvarů.

Úloha LP v kanonickém tvaru je následující optimalizační úloha daná maticí $A \in \mathbb{R}^{m \times n}$ a vektory $\mathbf{b} \in \mathbb{R}^m$ a $\mathbf{c} \in \mathbb{R}^n$:

$$\begin{aligned} \max \mathbf{c}^\top \mathbf{x} \\ \text{pro } \mathbf{x} \in \mathbb{R}^n \\ \text{za podmínek } A\mathbf{x} \leq \mathbf{b}. \end{aligned}$$

Úloha LP v rovnicovém tvaru je následující optimalizační úloha daná maticí $A \in \mathbb{R}^{m \times n}$ a vektory $\mathbf{b} \in \mathbb{R}^m$ a $\mathbf{c} \in \mathbb{R}^n$:

$$\begin{aligned} \max \mathbf{c}^\top \mathbf{x} \\ \text{pro } \mathbf{x} \geq \mathbf{0} \\ \text{za podmínek } A\mathbf{x} = \mathbf{b}. \end{aligned}$$

Příklad 1. *Pekárna peče chleby, housky, bagety a koblihy.*

- *K upečení jednoho chleba potřebuje půl kila mouky, 10 vajec a 50 g soli.*
- *Na jednu housku je zapotřebí 150 g mouky, 2 vejce a 10 g soli.*
- *Na bagetu potřebuje 230 g mouky, 7 vajec a 15 g soli.*
- *Na jednu koblihu je třeba 100 g mouky a 1 vejce.*

Pekárna má k dispozici 5 kilo mouky, 125 vajec a půl kila soli. Za jeden chleba získá pekárna 20 korun, za housku 2 koruny, za bagetu 10 korun a za koblihu 7 korun.

Pekárna se snaží vydělat co nejvíce. Jak ale zjistí kolik chlebů, housek, baget a koblih má upéci? Zformulujte příslušnou úlohu LP.

Grafická metoda na řešení úloh LP: dá se použít máme-li jen dvě proměnné, tedy $\mathbf{x} \in \mathbb{R}^2$. Potom množina přípustných řešení je konvexní polygon P, který si můžeme nakreslit a hledat na něm optimum účelové funkce $\mathbf{c}^\top \mathbf{x}$. Optimum najdeme tak, že uvážíme přímku $\{\mathbf{x} \in \mathbb{R}^2 \colon \langle \mathbf{c}, \mathbf{x} \rangle = 0\}$ kolmou na vektor $\mathbf{c}$ a procházející počátkem a posouváme ji ve směru vektoru $\mathbf{c}$, čímž dostáváme přímky $\{\mathbf{x} \in \mathbb{R}^2 \colon \langle \mathbf{c}, \mathbf{x} \rangle = h\}$ s $h = \langle \mathbf{c}, \mathbf{x} \rangle = \mathbf{c}^\top \mathbf{x}$ odpovídající hodnotě účelové funkce. Poslední body z P, na které narazíme, jsou potom hledanými optimy.

Příklad 2. *Vyřešte grafickou metodou následující systém nerovnic (neboli následující lineární program):*

$$\begin{aligned} -2x + 3y &\leq 3 \\ x + y &\leq 6 \\ -x + y &\geq -4 \\ x + 3y &\leq 12 \\ x &\geq 0 \\ y &\geq 0 \end{aligned}$$

Pro účelové funkce:

*Informace o cvičení naleznete na http://kam.mff.cuni.cz/~balko/

(a) max $x + y$

(b) max $-3x + y$

Co se stane, když odebereme poslední dvě podmínky, tedy $x \geq 0, y \geq 0$?

Příklad 3. *Ukažte, jak lze:*

1. *Převést maximalizační úlohu LP na minimalizační a naopak.*
2. *Převést úlohu LP, která má všechny proměnné $\mathbf{x} \geq \mathbf{0}$, na úlohu LP s proměnnými $\mathbf{x}' \in \mathbb{R}^n$ a naopak.*
3. *Převést úlohu LP s podmínkami ve tvaru nerovností na úlohu LP, jejíž podmínky jsou pouze rovnosti a naopak.*

Síť je uspořádaná čtveřice (G, z, s, c), kde $G = (V, E)$ je orientovaný graf, neboli $E \subseteq V \times V$, z a s jsou dva různé vrcholy grafu G (zvané *zdroj* a *stok*) a *kapacita* $c\colon E \to \mathbb{R}_0^+$ je funkce ohodnocující hrany. *Tok v síti* je každá funkce $f\colon E \to \mathbb{R}$ splňující $0 \leq f(e) \leq c(e)$ pro každou hranu $e \in E$ a

$$\sum_{v:(u,v)\in E} f(u,v) = \sum_{v:(v,u)\in E} f(v,u)$$

pro každý vrchol $u \in V$ mimo stok a zdroj. *Velikost toku* je

$$w(f) = \sum_{v:(z,v)\in E} f(z,v) - \sum_{v:(v,z)\in E} f(v,z).$$

Příklad 4. *Vytvořte lineární program, který najde maximální tok v síti $(G = (V, E), z, s, c)$.*

Příklad 5. *Formulujte* Prokládání přímkou *jako úlohu LP. Neboli máme-li n bodů $(x_1, y_1), \ldots, (x_n, y_n)$ v rovině, tak najděte přímku $\{x \in \mathbb{R}\colon y = ax + b\}$, která minimalizuje součet vertikálních vzdáleností bodů od výsledné přímky.* Vertikální vzdálenost *je vzdálenost měřena pouze na ose y.*

Pro jednoduchost předpokládejte, že výsledná přímka není kolmá na osu x.