# KAPITOLA 1

# FORMULACE ÚLOHY LINEÁRNÍHO PROGRAMOVÁNÍ


1 Základy lineárního programování . . . . . 8
2 Převody mezi tvary . . . . . 13
3 Úvod do celočíselného a smíšeného celočíselného programování . . . . . 16
4 Krátké srovnání s lineární algebrou . . . . . 18
5 Krátké historické okénko . . 19


Tato kapitola uvádí základní názvosloví lineárního programování a způsoby převodu slovních zadání do jednotného matematického tvaru. Jsou vysvětleny různé formy zápisu a tvary úloh lineárního programování, pojem přípustné množiny a optimálního řešení. Stručně také představíme celočíselné varianty lineárních programů, jejich vztah k LP-relaxaci a naznačíme souvislost s výpočetní složitostí. Kapitolu uzavírá krátký historický přehled vývoje oboru.

# 1 Základy lineárního programování

Na úvod připomeňme pro někoho možná překvapivý fakt: *lineární programování* historicky nesouvisí s programováním počítačů. Slovo „programování“ bylo převzato ve významu *rozvrh/plán* a tato disciplína vznikala už ve 40. letech 20. století, tedy dávno před nástupem běžně dostupných výpočetních strojů. Dnes se však při řešení praktických úloh bez počítače zpravidla neobejdeme – existuje bohatý ekosystém softwaru a teoreticky víme, že úlohu LP lze řešit efektivně (polynomiálně) speciálními metodami.

Pod *lineárním programem* (zkráceně LP) budeme rozumět úlohu, v níž hledáme *optimální* hodnotu lineární účelové funkce při splnění sady lineárních rovnic a/nebo nerovnic. V tomto smyslu se o LP často mluví také jako o *lineární optimalizaci*. Jednou z jeho velkých předností je formální jednoduchost: vše je lineární (jak účelová funkce, tak omezení), přesto jde o mimořádně mocný a v praxi hojně používaný aparát. Lineární modely totiž ve značném množství situací vystihují klíčové vazby dostatečně dobře: patří sem kapacitní omezení (stroje, zdroje), bilance materiálu či energie, rozpočtové limity, popř. parametricky zadané kvalitativní požadavky (např. minimální obsah suroviny). V dalším textu navážeme přesnými tvary zápisu a ukázkami typických převodů z verbální specifikace na maticový tvar LP.

## 1.1 Standardní a rovnicový tvar lineárního programu

Pro jednotu raději na začátku textu uvedeme implicitní předpoklady ve značení, které budeme uvažovat, pokud nebude řečeno jinak.

**Značení 1.1: Notace a konvence**

**Rozměry a prostory:** Nechť $m, n \in \mathbb{N}$ jsou počty omezení a proměnných. Vektory chápeme jako *sloupcové*. Pro matice uvažujeme $A \in \mathbb{R}^{m\times n}$. Vektory pak typicky značíme třemi znaky $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$, kde jde po řadě o vektor neznámé, vektor pravé strany a vektor koeficientů. Nerovnosti mezi vektory/maticemi jsou brány *po složkách*.

**Indexy:** Řádek $i$ matice $A$ označujeme $\mathbf{a}_i^\mathsf{T}$, $j$-tý prvek $i$-tého řádku značíme $a_{ij}$ Pro $j$-tou složku vektoru $\mathbf{x}$ píšeme $x_j$. Horním indexem $\cdot^\mathsf{T}$ značíme transpozici.

V této části přesně vymezíme, co budeme rozumět *standardním* tvarem lineárního programu a jak z něj jednoduše přejít do *rovnicového* tvaru, který je výhodný zejména pro algoritmické postupy (např. simplexovou metodu).

**Standardní tvar LP (maximalizační)**

Za *standardní maximalizační* tvar budeme považovat zápis

$$\begin{aligned} \textbf{maximalizovat} \quad & \mathbf{c}^\mathsf{T}\mathbf{x} \\ \textbf{při} \quad & A\,\mathbf{x} \leq \mathbf{b}, \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{1.1}$$

Zde explicitně uvedeme, že $A \in \mathbb{R}^{m\times n}$, $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$ a nerovnosti jsou chápány po složkách.

Dále definujeme pojem, který je nejen z hlediska lienárního programování, ale obecně z hlediska celé disciplíny optimalizace naprosto klíčový.

**Definice 1.2: Účelová funkce**

Pro lineární program budeme výraz

$$\mathbf{c}^\mathsf{T}\mathbf{x} \;=\; \sum_{j=1}^{n} c_j x_j$$

nazývat *účelovou funkcí*. Určuje hodnotu, kterou se snažíme maximalizovat nebo minimalizovat. Vektor $\mathbf{c} \in \mathbb{R}^n$ obsahuje koeficienty a proměnné $\mathbf{x} \in \mathbb{R}^n$ jsou rozhodovací proměnné úlohy.

V optimalizaci obecně nemusí účelová funkce nabývat konkrétního výše zmíněného tvaru. Například v nelineární optimalizaci si za účelovou funkcí můžeme představit všelijaké funkce různých tvarů – v tomto kurzu, i vzhledem k jeho jménu, si vystačíme pouze s účelovými funkcemi v lineárním tvaru.

¨

**Poznámka 1.3: Volné proměnné**

V některých zdrojích se jako „standardní“ používá i varianta bez podmínky $\mathbf{x} \geq 0$. V takovém případě lze každou volnou proměnnou (tj. bez vynucené nerovnosti, $\mathbf{x} \in \mathbb{R}$) rozložit (viz níže).

Standardní tvar (1.1) lze psát i „po složkách“:

$$\max \sum_{j=1}^{n} c_j x_j \quad \text{při} \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i \;\; (i = 1, \ldots, m).$$

Minimalizační varianta standardního tvaru se zapíše analogicky jako

$$\begin{aligned} \textbf{minimalizovat} \quad & \mathbf{c}^\mathsf{T}\mathbf{x} \\ \textbf{při} \quad & A\,\mathbf{x} \geq \mathbf{b}, \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

případně lze každou minimalizační úlohu převést na maximalizační (a naopak) změnou znaménka účelové funkce. Systematické převody mezi tvary budou shrnuty dále.

**Rovnicový tvar LP**

Rovnicový tvar získáme z (1.1) *doplněním pomocných proměnných* tak, aby se všechny nerovnosti změnily na rovnosti (po případném rozkladu volných proměnných na rozdíl nezáporných složek).
Po případném přeuspořádání řádků rozdělíme omezení na tři bloky:

$$A = \begin{pmatrix} A_{\leq} \\ A_{=} \\ A_{\geq} \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} \mathbf{b}_{\leq} \\ \mathbf{b}_{=} \\ \mathbf{b}_{\geq} \end{pmatrix},$$

kde bloky odpovídají řádkům s „$\leq$“, „$=$“ a „$\geq$“. Zaveďme

$$m_{\leq} := \text{počet nerovností „}\leq\text{“}, \quad m_{\geq} := \text{počet nerovností „}\geq\text{“}, \quad m' := m_{\leq} + m_{\geq},$$

a vektor pomocných proměnných $\mathbf{u} \in \mathbb{R}^{m'}$ s $\mathbf{u} \geq \mathbf{0}$, rozdělený jako $\mathbf{u} = \binom{\mathbf{u}_{\leq}}{\mathbf{u}_{\geq}}$.
Potom platí ekvivalentní rovnicový zápis

$$\begin{aligned} A_{\leq}\,\mathbf{x} + I_{m_{\leq}}\,\mathbf{u}_{\leq} &= \mathbf{b}_{\leq}, \\ A_{=}\,\mathbf{x} \qquad\qquad\quad &= \mathbf{b}_{=}, \qquad \mathbf{x} \geq \mathbf{0},\ \mathbf{u} \geq \mathbf{0}, \\ A_{\geq}\,\mathbf{x} - I_{m_{\geq}}\,\mathbf{u}_{\geq} &= \mathbf{b}_{\geq}, \end{aligned}$$

kde $I_{m_{\leq}}$ je jednotková matice $m_{\leq} \times m_{\leq}$ a $-I_{m_{\geq}}$ je rozměru $m_{\geq} \times m_{\geq}$.
Po sloučení do jedné matice dostaneme

$$A\,\mathbf{x} + E\,\mathbf{u} = \mathbf{b}, \qquad \mathbf{x} \geq \mathbf{0},\ \mathbf{u} \geq \mathbf{0},$$

kde

$$E \;=\; \begin{pmatrix} I_{m_{\leq}} & 0 \\ 0 & 0 \\ 0 & -I_{m_{\geq}} \end{pmatrix} \quad \text{(po přeuspořádání řádků do bloků } \leq, =, \geq\text{)}.$$

Zde $E \in \mathbb{R}^{m \times m'}$ a bloky 0 jsou nulové matice příslušných rozměrů.

Tento tvar je detailní a technický: slouží hlavně k tomu, abychom přesně viděli, jak se nerovnosti převádějí na rovnosti pomocí jednotkových bloků. Pro další práci se ale vyplatí zavést také zhuštěnou notaci, která sloučí všechny původní i pomocné proměnné do jedné vektorizované formy.

**Rovnicový tvar LP (zhuštěný)**

Kombinací původních a pomocných proměnných dostáváme úlohu

$$\mathbf{max}\ \tilde{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{x}} \quad \text{při} \quad \tilde{A}\,\tilde{\mathbf{x}} = \mathbf{b}, \qquad \tilde{\mathbf{x}} \geq \mathbf{0},$$

kde

$$\tilde{\mathbf{x}} = \begin{pmatrix} \mathbf{x} \\ \mathbf{u} \end{pmatrix} \in \mathbb{R}^{n+m'}, \qquad \tilde{A} = \begin{bmatrix} A & E \end{bmatrix}, \qquad \tilde{\mathbf{c}} = \begin{pmatrix} \mathbf{c} \\ \mathbf{0} \end{pmatrix}.$$

Platí $\tilde{A} \in \mathbb{R}^{m\times(n+m')}$ a $\tilde{\mathbf{c}} \in \mathbb{R}^{n+m'}$.

Na první pohled jsou oba zápisy poněkud technické, protože se snažíme zachovat původní značení $A, \mathbf{x}, \mathbf{b}, \mathbf{c}$ ze standardního tvaru a jen k nim přidáváme bloky. Tato volba má výhodu v tom, že můžeme přímo porovnávat standardní a rovnicový tvar. V dalším textu pak pro jednoduchost zápisu zpravidla vlnky u objektů vynecháme, vždy však s ohledem na jasné rozlišení kontextu.

**Poznámka 1.4: Proč rovnicový tvar?**

Rovnicový tvar je přirozený pro algoritmy, které pracují s *bázemi* (viz další sekce) a *bázickými řešeními*: rovnosti $\tilde{A}\tilde{\mathbf{x}} = \mathbf{b}$ spolu s nezápornostní proměnných umožní vybrat sloupce tvořící bázi, vyřešit odpovídající soustavu a zbylé proměnné položit na nulu.

Tato algebraická konstrukce přímo odpovídá „rohům“ přípustné množiny a je klíčová pro simplexovou metodu.

**Poznámka 1.5: Různé tvary v literatuře**

Je třeba upozornit, že terminologie pro označování různých „základních“ tvarů lineárního programu není v literatuře zcela jednotná. Někteří autoři používají pojem *kanonický tvar* pro zápis s nerovnostmi a nezápornými proměnnými, jiní naopak takto označují právě *rovnicový* tvar. Podobně se lze setkat i s pojmem *standardní tvar*, který může znamenat buď nerovnostní formulaci (jakou jsme zvolili zde), nebo naopak rovnicovou formulaci s doplněnými proměnnými.

Zopakujme, že pro účely tohoto textu budeme důsledně rozlišovat mezi *standardním* tvarem (nerovnosti) a *rovnicovým* tvarem (s pomocnými proměnnými), vždy tak, aby bylo jasné, v jakém kontextu se nacházíme.

## 1.2 Optimální řešení lineárního programu

Při jakémkoliv druhu optimalizace je důležité předem vymezit, v jaké množině optimální řešení chceme hledat tak, abychom zachovali námi stanovené podmínky. U lineárního programu zapsaného standardním tvarem nám vymezení této množiny vznikne poměrně přirozeně ze samotného zápisu. Shrneme ho v následující definici:

**Definice 1.6: Přípustná množina a optimální řešení.**

Množinu

$$P \;=\; \left\{\mathbf{x} \in \mathbb{R}^n \;\middle|\; A\mathbf{x} \leq \mathbf{b},\; \mathbf{x} \geq \mathbf{0}\right\}$$

nazveme *přípustnou množinou*. Prvek $\mathbf{x} \in P$ nazveme *přípustné řešení*. *Optimální řešení* $\mathbf{x}^\star \in P$ je takové, pro něž účelová funkce nabývá nejvyšší možné hodnoty $z^\star := \mathbf{c}^\mathsf{T}\mathbf{x}^\star$ (v minimalizační úloze by šlo o hodnotu nejnižší). Hodnotu $z^\star$ nazýváme *optimální*.

Přípustná množina pro rovnicový tvar LP bude přirozeně muset být zapsána jinak. Pro rovnicový tvar ji definujeme jednoduše jako

$$\tilde{P} \;:=\; \left\{\,\tilde{\mathbf{x}} \in \mathbb{R}^{n+m'} \;\middle|\; \tilde{A}\,\tilde{\mathbf{x}} = \mathbf{b},\; \tilde{\mathbf{x}} \geq \mathbf{0}\,\right\},$$

Zde $\tilde{\mathbf{x}}$ je vektor *všech* proměnných po standardních úpravách (původní proměnné po případném rozkladu na nezáporné složky a doplněné o $m'$ pomocných proměnných). V samotné definici není nutné tyto složky vyjmenovávat – důležité je, že pracujeme s rovnicemi a nezápornými proměnnými.

Zdůrazněme, že *projekce* přípustné množiny rovnicového tvaru do prostoru původních proměnných $x$ je stejná jako přípustná množina standardního tvaru. Je nutné mít vždy na paměti, že dané úloze „je jedno“, zda ji zapíšeme ve standardním nebo v rovnicovém tvaru a že přechod k rovnicovému tvaru LP je hlavně trik usnadňující fungování používaných algoritmů, a tedy by neměl nijak ovlivnit tvar řešení.

Jakmile máme jasně vymezeno, co rozumíme pod pojmem přípustná množina, můžeme snadno rozlišit i základní osudy celé úlohy lineárního programování. Toto rozdělení se objevuje v literatuře v různých variantách, pro naše účely si je shrneme do následující poznámky.

**Poznámka 1.7: Triáda osudů lineárního programu**

Každá úloha lineárního programování spadá právě do jedné z následujících kategorií:

a) **Nepřípustná** – přípustná množina $P$ je prázdná ($P = \varnothing$). Úloha tedy nemá žádné řešení splňující dané podmínky. Nepřípustnost lze dokládat vhodným certifikátem, jehož typickým příkladem je Farkašovo lemma.

b) **Neomezená** – přípustná množina $P$ je neprázdná ($P \neq \varnothing$), ale účelová funkce může růst (v maximalizačním tvaru), resp. klesat (v minimalizačním tvaru) libovolně bez dosažení konečného optima. Symbolicky zapisujeme

$$\sup_{\mathbf{x}\in P} \mathbf{c}^\mathsf{T}\mathbf{x} \;=\; +\infty, \quad \text{resp.} \inf_{\mathbf{x}\in P} \mathbf{c}^\mathsf{T}\mathbf{x} \;=\; -\infty.$$

c) **Má optimální řešení** – existuje bod $\mathbf{x}^\star \in P$, pro který je hodnota účelové funkce konečná a optimální:

$$z^\star \;=\; \mathbf{c}^\mathsf{T}\mathbf{x}^\star,$$

přičemž žádné jiné přípustné řešení nedosahuje vyšší (resp. nižší) hodnoty.

Toto trojí rozdělení představuje základní slovník, s nímž budeme pracovat po celý kurz, při analýze konkrétní úlohy se vždy nejprve ptáme, do které z těchto tří kategorií spadá.

# 2 Převody mezi tvary

V této části shrneme praktické recepty, jak převést obecně zadanou úlohu do jednotného tvaru, se kterým se dobře pracuje jak v teorii (geometrie, bázická řešení), tak v algoritmech (simplex, metody vnitřního bodu). Každý z převodů slouží k tomu, abychom dokázali řešit co nejširší škálu úloh v jedné sjednocené podobě.

**1. Minimalizace vs. maximalizace.** Každou minimalizační úlohu lze převést na maximalizační (a naopak) prostou změnou znaménka v účelové funkci:

$$\min \mathbf{c}^\mathsf{T}\mathbf{x} \quad \Longleftrightarrow \quad \max (-\mathbf{c})^\mathsf{T}\mathbf{x}.$$

Optimální řešení $\mathbf{x}^\star$ se nemění, pouze se změní znaménko optimální hodnoty.

**2. Směr nerovností a rovnosti.** Označme $\mathbf{a}_i^\mathsf{T}$ $i$-tý řádek $A \in \mathbb{R}^{m\times n}$ a $b_i$ $i$-tou složku $\mathbf{b} \in \mathbb{R}^m$. Nerovnost $\mathbf{a}_i^\mathsf{T}\mathbf{x} \geq b_i$ nahradíme vynásobením $-1$ tvarem

$$(-\mathbf{a}_i)^\mathsf{T}\mathbf{x} \leq -b_i.$$

Rovnost $\mathbf{a}_i^\mathsf{T}\mathbf{x} = b_i$ lze vždy chápat jako dvojici nerovností

$$\mathbf{a}_i^\mathsf{T}\mathbf{x} \leq b_i \qquad \text{a} \qquad (-\mathbf{a}_i)^\mathsf{T}\mathbf{x} \leq -b_i.$$

V praxi je však vhodné „skutečné“ rovnosti (např. bilance v sítích) ponechat jako rovnosti, aby se zbytečně nerozmnožovala omezení a nezvyšovalo riziko degenerace.

**3. Přechod do rovnicového tvaru: slack/surplus.** Chceme-li sjednotit tvar na rovnosti, doplníme vektor pomocných proměnných $\mathbf{u} \geq \mathbf{0}$, rozdělený podle směru nerovností jako $\mathbf{u} = \binom{\mathbf{u}_\leq}{\mathbf{u}_\geq}$, kde $\mathbf{u}_\leq \in \mathbb{R}^{m_\leq}$ a $\mathbf{u}_\geq \in \mathbb{R}^{m_\geq}$, $m' = m_\leq + m_\geq$. Po případném přeuspořádání řádků na bloky $\leq, =, \geq$ píšeme pro každý řádek

$$\sum_{j=1}^{n} a_{ij}x_j \leq b_i \implies \sum_{j=1}^{n} a_{ij}x_j + u_{\leq,i} = b_i, \quad u_{\leq,i} \geq 0,$$

$$\sum_{j=1}^{n} a_{ij}x_j \geq b_i \implies \sum_{j=1}^{n} a_{ij}x_j - u_{\geq,i} = b_i, \quad u_{\geq,i} \geq 0.$$

Po sloučení všech pomocných proměnných dostáváme kompaktní rovnicový tvar

$$A\,\mathbf{x} + E\,\mathbf{u} = \mathbf{b}, \qquad \mathbf{x} \geq \mathbf{0},\ \mathbf{u} \geq \mathbf{0},$$

kde

$$A = \begin{pmatrix} A_\leq \\ A_= \\ A_\geq \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} \mathbf{b}_\leq \\ \mathbf{b}_= \\ \mathbf{b}_\geq \end{pmatrix}, \qquad E = \begin{pmatrix} I_{m_\leq} & 0 \\ 0 & 0 \\ 0 & -I_{m_\geq} \end{pmatrix}.$$

**Poznámka 1.8: Slack a surplus proměnné**

Pomocné proměnné se tradičně označují jako *slack* (pro „≤“ nerovnosti, představují volnou rezervu) a *surplus* (pro „≥“ nerovnosti, představují přebytek). Tyto názvy přebíráme z cizojazyčné literatury i klasického názvosloví lineárního programování.

**4. Volné proměnné a intervalové meze.** Volnou proměnnou $x_j \in \mathbb{R}$ nahradíme rozdílem dvou nezáporných složek

$$x_j = x_j^+ - x_j^-, \qquad x_j^+, x_j^- \geq 0.$$

Má-li proměnná interval $L_j \leq x_j \leq U_j$, zavedeme posun $x_j = L_j + y_j$ a pracujeme s $0 \leq y_j \leq U_j - L_j$. Samotnou horní mez lze ponechat jako lineární omezení nebo převést na rovnost doplněním slacku.

**5. Absolutní hodnota a maximum z lineárních funkcí.** V minimalizačních úlohách se často objevují následující konstrukce:

- *Absolutní hodnota v omezení:* $|\ell(\mathbf{x})| \leq t$ s lineární $\ell : \mathbb{R}^n \to \mathbb{R}$ nahradíme dvojicí

$$-t \;\leq\; \ell(\mathbf{x}) \;\leq\; t, \qquad t \geq 0.$$

- *Absolutní hodnota v účelové funkci:* pro minimalizaci

$$\min_{\mathbf{x}} \sum_i w_i \left|\ell_i(\mathbf{x})\right| \quad \text{(s váhami } w_i \geq 0)$$

  zavedeme proměnné $y_i, z_i \geq 0$ tak, že

$$\ell_i(\mathbf{x}) \;=\; y_i - z_i.$$

  Potom řešte LP

$$\min_{\mathbf{x},\mathbf{y},\mathbf{z}} \sum_i w_i \,(y_i + z_i) \quad \text{při} \quad y_i, z_i \geq 0, \;\; \ell_i(\mathbf{x}) = y_i - z_i.$$

  V optimu platí $|\ell_i(\mathbf{x})| = y_i + z_i$ (alespoň jedna z $y_i, z_i$ je nulová).

- *Maximum z lineárních funkcí:* výraz

$$\min_{\mathbf{x}} \; \max_k \{\, a_k^{\mathsf{T}}\mathbf{x} + b_k \,\}$$

  převedeme pomocí pomocné proměnné $t$ na

$$\min_{\mathbf{x},t} \;\; t \quad \text{při} \quad a_k^{\mathsf{T}}\mathbf{x} + b_k \;\leq\; t \;\; \text{pro všechna } k.$$

**Poznámka 1.9: Shrnující recept: převod do jednotného tvaru**

1. **Zvolte minimalizaci.** Je-li úloha původně maximalizační, změňte znaménko v účelové funkci.
2. **Sjednoťte směr nerovností na „≤“.** Nerovnosti „≥“ převeďte vynásobením $-1$.
3. **Přejděte do rovnicového tvaru.** Ke každé nerovnosti „≤“ doplňte *slack*, ke každé „≥“ *surplus*, viz výše.
4. **Ošetřete proměnné.** Volné proměnné rozložte na rozdíl dvou nezáporných; intervalové meze převeďte posunem.
5. **Linearizujte jednoduché nelinearity.** Absolutní hodnoty a maximum z lineárních funkcí přepište standardními kroky.
6. **Získejte finální tvar.** Výsledkem je rovnicový tvar s nezápornými proměnnými:
$$\min \mathbf{c}^{\mathsf{T}}\mathbf{x} \quad \text{při} \quad A\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}.$$

Na závěr sekce ilustrujeme převody na dvou stručných příkladech. Nejprve ukážeme postup, jak převést úlohu s minimalizací, nerovností typu „≥“ a volnou proměnnou, a poté si ukážeme práci s intervalovými mezemi.

**Příklad 1.10: Krátký převod I**

**Minimalizovat účelovou funkci** $2x_1 - x_2$ **při**

$$-x_1 + 2x_2 \geq 1, \qquad x_1 + x_2 = 3, \qquad x_2 \text{ je volná.}$$

**Kroky převodu:**

- min→max: $\max(-2x_1 + x_2)$,
- „≥”→„≤”: $x_1 - 2x_2 \leq -1$,
- rozklad $x_2 = x_2^+ - x_2^-$, $x_2^+, x_2^- \geq 0$,
- slack $s$: $x_1 - 2(x_2^+ - x_2^-) + s = -1$, $s \geq 0$.

Rovnost zůstává $x_1 + (x_2^+ - x_2^-) = 3$. Pokud je i $x_1$ volná, rozložte ji analogicky.

Následující příklad ukazuje, jak pracovat s intervalovými mezemi proměnné. Tento postup se hodí v praxi například tehdy, když proměnná představuje kapacitu, která má dolní i horní hranici.

**Příklad 1.11: Krátký převod II**

Má-li proměnná mezní interval $1 \leq x \leq 5$, zaveďte posun

$$x = 1 + y, \qquad 0 \leq y \leq 4,$$

a ve všech omezeních i v účelové funkci nahraďte $x$ výrazem $1 + y$.
**Volitelně pro rovnicový tvar:**

- ponechte horní mez jako nerovnost $y \leq 4$, *nebo*
- použijte slack: $y + s = 4$, $s \geq 0$.

# 3 Úvod do celočíselného a smíšeného celočíselného programování

V mnoha praktických úlohách nestačí požadovat, aby byly proměnné pouze reálné a nezáporné. Potřebujeme rozhodovat *diskrétně*: otevřít/neotevřít provoz, zvolit jednu z variant, počítat kusy, směny nebo vozidla po celých jednotkách apod. To vede k *celočíselnému* (ILP) a *smíšenému celočíselnému* (MILP) lineárnímu programování.

ILP a MILP tedy rozšiřují LP o celočíselné podmínky na proměnné, které umožní modelovat mnoho reálných siutací. V této fázi kurzu budeme ILP/MILP pouze formulovat a hlouběji se tomuto tématu budeme věnovat v samotném závěru, kde se seznámíme s konkrétními metodami řešení takovýchto úloh.

## 3.1 Tvary úloh ILP a MILP

Maximalizační úloha celočíselného programování není nic jiného než nám dobře známá základní úloha LP s tím, že proměnné smějí nabývat pouze celočíselných hodnot.

**Celočíselný lineární program (ILP)**

Za maximalizační *celočíselnou* uvažujeme takovou úlohu LP, ve které *všechny* rozhodovací proměnné musí nabývat celočíselných hodnot, tj. úlohu

$$\begin{aligned} \textbf{maximalizovat} \quad & \mathbf{c}^\mathsf{T}\mathbf{x} \\ \textbf{při} \quad & A\,\mathbf{x} \leq \mathbf{b}, \\ & \mathbf{x} \in \mathbb{Z}^n,\ \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

kde $A \in \mathbb{R}^{m \times n}$ je matice koeficientů omezení, $\mathbf{b} \in \mathbb{R}^m$ je vektor pravých stran a $\mathbf{c} \in \mathbb{R}^n$ je vektor koeficientů účelové funkce. Zvláštní, ale velmi častý případ tvoří *binární* ILP, kde $x_j \in \{0, 1\}$.

Vynutíme-li kombinaci výše zmíněného, tj. část z proměnných bude nabývat reálných a část celočíselných hodnot, dostaneme smíšenou úlohu LP.

**Smíšený celočíselný lineární program (MILP)**

Za maximalizační *smíšenou celočíselnou* uvažujeme takovou úlohu LP, kde část proměnných je celočíselná a zbytek reálný, tj. po přeuspořádání proměnných chceme

$$\begin{aligned} \textbf{maximalizovat} \quad & \mathbf{c}_y^\mathsf{T}\mathbf{y} + \mathbf{c}_z^\mathsf{T}\mathbf{z} \\ \textbf{při} \quad & A_y\,\mathbf{y} + A_z\,\mathbf{z} \leq \mathbf{b}, \\ & \mathbf{y} \in \mathbb{Z}^p, \quad \mathbf{z} \in \mathbb{R}^{n-p},\ \mathbf{y} \geq \mathbf{0},\ \mathbf{z} \geq \mathbf{0}, \end{aligned}$$

kde $\mathbf{y}$ jsou celočíselné proměnné, $\mathbf{z}$ jsou reálné a $p \in \{0, 1, \ldots, n\}$ je počet celočíselných proměnných, $A_y \in \mathbb{R}^{m\times p}$, $A_z \in \mathbb{R}^{m\times(n-p)}$, $\mathbf{c}_y \in \mathbb{R}^p$, $\mathbf{c}_z \in \mathbb{R}^{n-p}$.

Pohledem na množinu přípustných řešení je zásadní rozdíl v tom, že u LP je přípustná množina *konvexní mnohostěn* (což rozebereme v dalších částech kurzu), kdežto u ILP/-MILP jde o *diskrétní* (typicky nekonvexní) množinu bodů či sjednocení polyedrických částí. To má důsledky jak teoretické (nelze spoléhat obecné výsledky z teorie LP), tak algoritmické (nelze použít jen metody pro LP).

## 3.2 Výpočetní složitost ILP/MILP

Z pohledu složitosti jsou úlohy ILP a MILP obecně rovněž odlišené od klasické LP úlohy. Pro účely krátkého shrnutí této výpočetní složitosti, rozlišíme nyní dvě verze úloh:

- Rozhodovací verze: „Existuje přípustné řešení s hodnotou účelové funkce alespoň $K$?“ (ano/ne).
- Optimalizační verze: „Najdi nejlepší hodnotu účelové funkce.“

Zatímco lineární programy lze obecně řešit v polynomiálním čase díky existujícím odvozeným metodám, celočíselný lineární program je *NP-úplný* v rozhodovací verzi a *NP-těžký* v optimalizační verzi. Totéž platí pro MILP, protože zahrnuje ILP jako speciální případ.

**Poznámka 1.12: Co znamenají pojmy NP-těžký a NP-úplný?**

V teorii komplexity říkáme, že problém $H$ je **NP-těžký**, jestliže na něj lze v polynomiálním čase převést každý problém z třídy NP. Pokud bychom našli rychlý algoritmus pro $H$, vyřešili bychom tím i slavný problém „$P = NP$?“ – většina odborníků však tipuje, že takový algoritmus neexistuje.

**NP-úplný** problém je ještě specifičtější: patří do třídy NP (tedy řešení lze *ověřit* v polynomiálním čase) a zároveň je NP-těžký. NP-úplné problémy jsou tedy „nejtěžší“ uvnitř třídy NP – jsou v NP a jsou stejně těžké jako všechny ostatní problémy z NP.

## 3.3 LP-relaxace

V praxi často nastává situace, kdy při řešení úloh ILP/MILP pomůže dočasně uvolnit, čili relaxovat, požadavky na celočíselnost proměnných. Tomuto procesu se říka *LP-relaxace* – jde tedy o prostou záměnu celočíselných podmínek za reálné:

$$\mathbf{y} \in \mathbb{Z}^p \;\rightsquigarrow\; \mathbf{y} \in \mathbb{R}^p,$$

případně se přitom zachovají nezápornosti či intervalové meze. Tato úprava odstraní diskrétnost a úlohu převede zpět na klasický lineární program, který lze řešit standardními metodami.

**Poznámka 1.13: LP-relaxace jako mez**

K čemu řešení relaxované úlohy může být dobré? Zcela jistě poskytuje užitečný *horní/dolní odhad* na nejlepší možné řešení původní celočíselné úlohy. To proto, že každé přípustné řešení původní úlohy je také přípustným řešením LP relaxace, čili v LP relaxaci optimalizujeme přes větší množinu vektorů.

- U maximalizace platí $z^\star_{\text{LP}} \;\geq\; z^\star_{\text{ILP}}$.
- U minimalizace platí $z^\star_{\text{LP}} \;\leq\; z^\star_{\text{ILP}}$.

Pro konkrétní instanci úlohy lze rozdíl mezi celočíselným a relaxovaným řešením vyjádřit jako jejich *poměr*, který se označuje jako *integrality gap*:

$$\frac{z^\star_{\text{ILP}}}{z^\star_{\text{LP}}} \quad \text{(pro minimalizaci)}, \qquad \frac{z^\star_{\text{LP}}}{z^\star_{\text{ILP}}} \quad \text{(pro maximalizaci)}.$$

Tyto poměry jsou vždy $\geq 1$ a vyjadřují, o kolik může být řešení relaxované úlohy „optimističtější“ než nejlepší celočíselné řešení, a tedy jak moc je daná relaxace pro daný případ těsná.

# 4 Krátké srovnání s lineární algebrou

První semestr jste pravděpodobně strávili u matic a Gaussovy eliminace; v LP naproti tomu místo čistých rovnic $Ax = b$ běžně povolíme i nerovnosti a místo *libovolného* řešení hledáme to *nejlepší*. Nejde však o skok do neznáma – většina pojmů z lineární algebry se v LP znovu objeví, jen v ostřejším světle. Níže uvádím pět klíčových paralel, vždy s „algebraickou“ a „programovací“ stránkou.

**Ústřední úloha.** *Lineární algebra* řeší soustavu rovnic $A\,\mathbf{x} = \mathbf{b}$ a výsledkem je (pokud existuje) afinní podprostor všech řešení. *Lineární programování* maximalizuje nebo minimalizuje lineární funkci $\mathbf{c}^\mathsf{T}\mathbf{x}$ při systému rovnic a/nebo nerovnic $A\,\mathbf{x} \leq \mathbf{b}$; přípustná oblast je mnohostěn a zajímá nás nejlepší bod uvnitř.

**Algoritmický „dělník“.** V algebře dominuje Gaussova eliminace – posloupnost řádkových operací vedoucí k trojúhelníkovému tvaru. V LP hrají obdobnou roli *simplexová metoda* (skoky po hranách od vrcholu k vrcholu) a *metody vnitřního bodu* (sledování křivky uvnitř mnohostěnu).

**Certifikát neřešitelnosti.** Neřešitelnost soustavy $A\,\mathbf{x} = \mathbf{b}$ lze snadno ověřit pomocí hodnostního kritéria. U nerovnic hraje roli Farkašovo lemma, s kterým se setkáme později.

**Geometrie.** Afinní podprostor je „plochý“ – nemá vyhraněné body. Mnohostěn má naopak *vrcholy*, *hrany* a *stěny* – právě vrcholy se ukážou klíčové, protože optimum lineárního programu leží (při existenci) na některém z nich, jak uvidíme později.

# 5 Krátké historické okénko

První zárodky „lineárního“ optimalizačního přístupu se dají vystopovat už v memorandu Gasparda Mongeho z roku 1781. Monge v něm řešil, jak co nejúsporněji přesunout zeminu z výkopů na násypy – klasický transportní problém, který se o téměř 160 let později stane jedním z pilířů lineárního programování [1]. Ještě před tím, než se vůbec objevilo slovo „programování“ ve smyslu matematického plánování, vznikaly v 19. a na počátku 20. století teoretické stavební kameny lineárního programování. **Lagrangeovy multiplikátory** na konci 18. století převáděly hledání vázaných extrémů na řešení soustav rovnic; Joseph Fourier pak počátkem 19. století popsal eliminační metodu pro soustavy lineárních *nerovnic*, kterou o více než sto let později znovuobjevil Theodore Motzkin – dnes tuto techniku známe jako Fourier-Motzkinovu eliminaci [2]. K přelomu 19. a 20. století se váže také **Farkasova lemma** o řešitelnosti lineárních nerovnic, jež tvoří základ teorie duality a bývá označováno za fundamentální větu lineární algebry. Toto lemma bylo poprvé dokázáno Farkasem a nezávisle Hermannem Minkowskim; další zpřesnění přidali matematici jako Constantin Carathéodory či Hermann Weyl [3]. Právě jejich práce o konvexních množinách (Minkowskiho a Carathéodoryho věty) společně s Farkasovým lemmatem předznamenaly pozdější rozvoj teorie lineárního programování.

Třicátá léta dvacátého století patřila především „čisté“ geometrii nerovnic. **Theodor S. Motzkin** ve své disertační práci z roku 1936 podal ucelený přehled desítek do té doby známých výsledků o soustavách lineárních nerovnic, aniž se přitom mluvilo o nějaké optimalizaci – ta ještě nebyla hlavním tématem. Motzkin ve své práci systematicky sesbíral a okomentoval veškerou dostupnou literaturu o lineárních nerovnicích a souvisejících tématech [4]. Teprve rok 1939 znamenal první skutečný průlom směrem k modernímu pojetí optimalizace: **Leonid Kantorovič** v Leningradu formuloval ekonomické modely, v nichž je třeba maximalizovat (či minimalizovat) lineární funkci nákladů nebo zisku při splnění daných lineárních omezení. Ve svých studiích – od problémů průmyslového plánování až po logistiku zásobování obleženého Leningradu – dal Kantorovič rodícímu se lineárnímu programování jasný obsah i společenskou naléhavost [5]. Za tyto myšlenky obdržel Kantorovič společně s Tjallingem Koopmansem Nobelovu cenu za ekonomii v roce 1975 [6].

Skutečnou explozi oboru odstartoval **George B. Dantzig**. Jako matematik působící v Pentagonu hledal roku 1947 formální nástroj, jak koordinovat výcvik, logistiku a nasazení letectva, a přijal výzvu „mechanizovat“ tehdejší pracné ruční plánování vojenských operací [7]. Inspirován mj. Leontiefovým vstupně-výstupním modelem ekonomiky [8] formuloval Dantzig obecný model lineárního programování a během několika týdnů v létě 1947 pro něj navrhl simplexovou metodu [7] – algoritmus, který hledá optimální řešení postupným přechodem z vrcholu na vrchol příslušného mnohostěnu při monotónním zlepšování hodnoty účelové funkce. Sám Dantzig později připomínal, že tehdejší termín „programování“ neznamenal psaní kódu, nýbrž plán či harmonogram [9]. Svůj objev zároveň intenzivně konzultoval s předními ekonomy a teoretiky her – kromě Koopmanse a Leontiefa také s

Johnem von Neumannem, který okamžitě rozpoznal ekvivalenci lineárního programování a dvoučlenných her a jako první formuloval princip duality [10]. Již v červnu 1951 uspořádalo americké letectvo ve Washingtonu první velké sympozium o lineárním programování [11] – signál, že na konci 40. let se zrodil obecný nástroj, o jehož významu už nebylo pochyb.

Vzápětí následoval rychlý průnik do praxe. Už koncem 40. let se na stolních kalkulačkách řešil tzv. **dietní problém**: hledal se nejlevnější jídelníček splňující dané nutriční požadavky. Ekonom George Stigler publikoval roku 1945 aproximativní řešení modelu o 9 výživových podmínkách a 77 potravinách, neboť na přesné spočtení tehdy chyběla metoda. Když byl tentýž problém roku 1947 vůbec poprvé vyřešen simplexovou metodou „na papíře“, šlo o výpočtářsky mimořádně náročný počin – trval řádově stovky člověko-hodin – který však potvrdil, že Stiglerův odhad se od optima lišil jen nepatrně [12]. S nástupem počítačů se však situace dramaticky zlepšila: již v roce 1951 Alex Orden naprogramoval simplexovou metodu na experimentálním počítači SEAC a vyřešil tak pro armádu model o 48 rovnicích a 71 neznámých za 18 hodin [13]. Krátce poté následovaly pokročilejší implementace – například na stroji IBM 701 v roce 1954 už bylo možné řešit úlohy se stovkou podmínek [13]. Simplexová metoda se tak s příchodem elektronických počítačů stala standardním průmyslovým nástrojem pro optimalizaci.

Vedle již zmíněného transportního problému – poprvé formulovaného Mongem a znovu analyzovaného ve 30. letech A. N. Tolstým a ve 40. letech F. L. Hitchcockem [14] – se záhy prosadily i další klasické úlohy operačního výzkumu řešené pomocí LP. **Přiřazovací problém**, spočívající v optimálním přiřazení pracovníků k úkolům, vedl k zavedení tzv. „maďarské metody“ (Kuhn 1955), postavené na výsledcích maďarských matematiků Dénese Kőniga a Jenő Egerváryho [15]. V 50. letech **Abraham Charnes a William W. Cooper** ukázali, jak LP využít v řízení výroby a podniku – například při optimálním míchání paliv v rafinérii [16] – a během 50. a 60. let se zájem o lineární programování rozlil do státní správy i široké akademické obce [17]. Během necelých dvou dekád se z něj stal ústřední nástroj celého výzkumu operací.

Sedmdesátá a osmdesátá léta přinesla zásadní teoretické objevy. V roce 1972 ukázali Victor Klee a George J. Minty na „protažené krychli“ (Klee–Minty polytope), že simplexová metoda může v nejhorším případě vyžadovat exponenciálně mnoho pivotových kroků [18]. Koncem sedmdesátých let pak Leonid Chačijan – nezávisle na práci Davida Judina a Arkadije Němirovského v SSSR – dokázal, že lineární programy jsou řešitelné v polynomiálním čase pomocí ellipsoidní metody [19]. Definitivní průlom směrem k praktickému využití polynomiálních algoritmů ale přinesl až rok 1984, kdy **Narendra Karmarkar** publikoval projektivně škálující metodu vnitřního bodu. Šlo o první široce použitelný algoritmus s teoretickou polynomiální časovou složitostí, který odstartoval zlatou éru metod vnitřního bodu a stal se základem moderních solverů [19]. Od té doby se v praxi osvědčila kombinace obou přístupů: robustní simplexová metoda pro start a dotahování řešení do vrcholu, a metody vnitřního bodu pro rychlé získání hrubého optimálního řešení u velkých a dobře strukturovaných úloh [20].

Na přelomu tisíciletí zařadil časopis *Computing in Science & Engineering* Dantzigovu simplexovou metodu – po bok například rychlé Fourierovy transformace, Quicksortu či Metropolisova algoritmu – mezi deset nejvlivnějších algoritmů 20. století [21]. Šlo o symbolické uznání celému oboru, který v sobě jedinečně spojuje eleganci teoretické matematiky s obrovským dopadem na reálný svět.

# BIBLIOGRAFIE


[1] G. Monge, "Mémoire sur la théorie des déblais et des remblais," Histoire de l'Académie Royale des Sciences, 1781.

[2] J. Fourier, "Solution d'une question particulière du calcul des inégalités," Journal de l'École Royale Polytechnique, 1827.

[3] G. Farkas, "Über die Theorie der einfachen Ungleichungen," Journal für die reine und angewandte Mathematik, 1902.

[4] T. S. Motzkin, "Beiträge zur Theorie der linearen Ungleichungen," Dissertation, Universität Basel, 1936.

[5] L. Kantorovich, "Mathematical Methods in the Organization and Planning of Production," Publication of the Leningrad University, 1939.

[6] Nobel Prize Committee, "The Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel 1975," Nobel Foundation, 1975.

[7] G. B. Dantzig, "Maximization of a Linear Function of Variables Subject to Linear Inequalities," Pentagon Report, 1947.

[8] W. Leontief, "The Structure of American Economy, 1919–1929," Harvard University Press, 1941.

[9] G. B. Dantzig, "Linear Programming and Extensions," Princeton University Press, 1963.

[10] J. von Neumann, "A Model of General Economic Equilibrium," Review of Economic Studies, 1947.

[11] Proceedings of the Symposium on Linear Programming, Washington D.C., 1951.

[12] G. J. Stigler, "The Cost of Subsistence," Journal of Farm Economics, 1945.

[13] A. Orden, "The Solution of Linear Programming Problems on the SEAC," Technical Report, NBS, 1952.

[14] F. L. Hitchcock, "The Distribution of a Product from Several Sources to Numerous Localities," Journal of Mathematics and Physics, 1941.

[15] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," Naval Research Logistics Quarterly, 1955.

[16] A. Charnes and W. W. Cooper, “Programming with Linear Fractional Functionals,” Naval Research Logistics Quarterly, 1953.

[17] M. J. Todd, “The Many Facets of Linear Programming,” Mathematical Programming, 1990.

[18] V. Klee and G. J. Minty, “How Good is the Simplex Algorithm?,” Inequalities III, 1972.

[19] L. Khachiyan, “A Polynomial Algorithm in Linear Programming,” Doklady Akademii Nauk SSSR, 1979.

[20] S. Wright, “Primal-Dual Interior-Point Methods,” SIAM, 1997.

[21] J. Dongarra and F. Sullivan, “Top Ten Algorithms of the 20th Century,” Computing in Science & Engineering, 2000.