8 Podrobnosti a variace metody

Dostáváme se k poslední lekci našeho textu týkající se lineárního programování, která pojednává o třech doposud opomíjených (a problematických) detailech simplexové metody:

- Jak efektivně získat výchozí přípustné bázické řešení úlohy (pomocí umělých proměnných).
- Jak účinně zamezit zacyklení simplexové metody v degenerovaných bázických řešeních (lexikografické pravidlo).
- Jak odhadnout celkový počet iterací simplexové metody.

Zatímco pro první dva body si ukážeme dostačující řešení, ten třetí je už po dlouhá léta těžkým teoretickým problémem v optimalizaci a uspokojující odpověď na něj není známa.

Stručný přehled lekce

- Dvoufázová simplexová metoda pro zbavení se umělých proměnných.
- Ukázkový výpočet dvou fází.
- Lexikografická varianta simplexové metody pro zabránění zacyklení.
- Další poznámky k variantám a složitosti metody.

8.1 Umělé proměnné; dvě fáze

Jak bylo popsáno v Algoritmu 7.2, umělé proměnné přidáváme do výchozí tabulky, abychom snadno získali výchozí přípustné bázické řešení. Pochopitelně však požadujeme, aby ve výsledném řešení měly tyto umělé proměnné nulovou hodnotu.

K odstranění umělých proměnných se přirozeně nabízejí dva postupy:

- *"Nekonečná" cena:*

  Umělým proměnným $\vec{u}$ přiřadíme cenu $-\nu$, kde ν je velmi velké číslo (nepřesně bychom mohli zapsat, že $\nu = \infty$). Pokud některá umělá proměnná vyjde ve výsledném řešení větší než nula, pak původní úloha nemá přípustné řešení.

  Pozor, nezapomeňme, že ceny $-\nu$ v nultém řádku se musíme zbavit už ve výchozí tabulce, a to přičtením ν-násobku i-tého řádku k nultému řádku.

- *Dvoufázová simplexová metoda:*

  Ve dvoufázové metodě všem umělým proměnným $\vec{u}$ nejprve přiřadíme cenu -1 a původním proměnným cenu 0. (Součet všech umělých proměnných tak minimalizujeme.) Jestliže optimum této *první fáze* vyjde nenulové, původní úloha nemá přípustné řešení. Naopak pro nulové optimum umělé proměnné následně vypustíme a pokračujeme *druhou fází* v řešení původní úlohy.

Algoritmus 8.1. Implementace dvoufázové simplexové metody

Dvoufázová simplexová metoda (zapsaná simplexovou tabulkou) je založena na postupu Algoritmu 6.4 jednofázové metody s následujícími vylepšeními.

0. Vyjdeme z (už známého) kanonického tvaru úlohy v redukovaném zápise:

$$\begin{aligned} \min\ & x_0 \\ x_0 + \vec{c}\cdot\vec{x} &= 0 \\ \boldsymbol{A}\cdot\vec{x} &= \vec{b} \\ \vec{x} &\geq 0 \end{aligned}$$

1. Jako v Algoritmu 7.2, bod 1, po případném přidání umělých proměnných zapíšeme výchozí simplexovou tabulku v jednotkovém tvaru

$$\left[\begin{array}{cc|c} \vec{c} & 0\ldots0 & 0 \\ \hline \underbrace{\boldsymbol{A}' \quad \boldsymbol{I}}_{\boldsymbol{A}^u} & & \vec{b}' \end{array}\right],$$

která vyjadřuje vztahy

$$\begin{aligned} x_0 + \vec{c}\cdot\vec{x} &= 0 \\ \boldsymbol{A}^u\cdot(\vec{x},\vec{u})^T &= \vec{b}' \\ \vec{x},\vec{u} &\geq 0\,. \end{aligned}$$

Nyní však pro *první fázi* k ocenění umělých proměnných zavedeme novou účelovou funkci a pro $\vec{c}^{\,u} = (1, \vec{c}, \vec{0})$ novou úlohu zapíšeme jako

$$\begin{aligned} \max \quad & -u_1 - \ldots - u_k \\ & \vec{c}^{\,u} \cdot (x_0, \vec{x}, \vec{u})^{\,T} &&= 0 \\ & \boldsymbol{A}^u \cdot (\vec{x}, \vec{u})^{\,T} &&= \vec{b}' \\ & \vec{x}, \vec{u} &&\geq 0\,. \end{aligned}$$

$$\left[\begin{array}{c|c} 0 \ldots 0 \quad -1 \ldots -1 & 0 \\ \hline \vec{c} \quad 0 \ \ldots \ 0 & 0 \\ \hline \boldsymbol{A}^u & \vec{b}' \end{array}\right]. \tag{1}$$

Nakonec ještě musíme tabulku upravit tak, aby účelový řádek obsahoval 0 ve sloupcích umělých proměnných.

2.–5. Postupujeme v těchto bodech podle Algoritmu 6.4, ale máme na paměti, že i původní účelový řádek je vyloučen z působnosti řádkového pravidla, třebaže jinak je pokládán za běžný řádek tabulky.

6. Pokud optimální řešení první fáze má kladnou hodnotu (tedy $\vec{u} \neq \vec{0}$), pak původní úloha nemá žádné přípustné řešení.

7. Jinak z výsledné tabulky první fáze vypustíme umělý účelový řádek i všechny sloupce umělých proměnných. Pokračujeme ve výpočtu *druhé fáze* metody už známým způsobem od bodu **2** Algoritmu 7.2.

Věta 8.2. *Pokud první fáze Algoritmu 8.1 skončí s optimálním řešením s hodnotou* 0*, pak po vypuštění umělých proměnných v bodě 7 získáme správnou výchozí simlexovou tabulku v jednotkovém tvaru pro původní úlohu LP. Pokud naopak první fáze skončí s řešením s kladnou hodnotou, pak původní úloha LP nemá žádné přípustné řešení.*

Důkaz:

. . . . . □

Důsledek 8.3. *Algoritmus 8.1 správně řeší úlohy LP.*

8.2 Ukázkový výpočet

Pro lepší pochopení dvoufázové simplexové metody je nejlepší se podívat na malý názorný příklad jejího použití.

Příklad 8.4. *Vyřešme Příklad 7.5 za pomocí dvoufázové simplexové metody.*

Připomeneme, že v kanonickém tvaru úloha zní

$$\begin{array}{lrrrrrl} \max & 80x_1 + 50x_2 & & & & & \\ & 20x_1 + 15x_2 & +x_3 & & & = & 1000 \\ & 4x_1 \quad + 2x_2 & & +x_4 & & = & 160 \\ & x_1 & & & -x_5 & = & 30 \\ & & x_1, x_2, & & x_3, x_4, x_5 & \geq & 0\,, \end{array}$$

což bohužel nedává tabulku v jednotkovém tvaru. Opět tak budeme muset nejprve zavést umělou proměnnou do poslední rovnice.

Podstatou dvoufázové simplexové metody je, že optimalizace probíhá ve dvou stupních, nejprve vyloučením umělých proměnných a pak teprve původní účelovou funkcí. Proto nejprve musíme původní účelovou funkci převést na rovnost s x_0 v redukovaném tvaru.

$$\begin{array}{rcl} -x_0 + 80x_1 + 50x_2 & = & 0 \\ 20x_1 + 15x_2 \quad +x_3 & = & 1000 \\ 4x_1 \quad + 2x_2 \qquad +x_4 & = & 160 \\ x_1 \qquad\qquad -x_5 & = & 30 \\ x_1, x_2, \qquad x_3, x_4, x_5 & \geq & 0\,, \end{array}$$

Nyní je čas zavést umělou proměnnou x_6, jejíž hodnotu budeme v první fázi minimalizovat.

$$\max\ -x_6$$

$$\begin{array}{rcl} -x_0 + 80x_1 + 50x_2 & = & 0 \\ 20x_1 + 15x_2 \quad +x_3 & = & 1000 \\ 4x_1 \quad + 2x_2 \qquad +x_4 & = & 160 \\ x_1 \qquad\qquad -x_5 + x_6 & = & 30 \\ x_1, x_2, \qquad x_3, x_4, x_5, x_6 & \geq & 0\,, \end{array}$$

Z poslední formulace teď sestavíme výchozí simplexovou tabulku, která již po úpravě nultého řádku bude v jednotkovém přípustném tvaru.

0	0	0	0	0	-1	0
80	50	0	0	0	0	0
20	15	1	0	0	0	1000
4	2	0	1	0	0	160
1	0	0	0	-1	1	30

$\longrightarrow$

1	0	0	0	-1	0	30
80	50	0	0	0	0	0
20	15	**1**	**0**	0	**0**	1000
4	2	**0**	**1**	0	**0**	160
1	0	**0**	**0**	-1	**1**	30

Všimněme si dobře, že naše tabulka obsahuje dva účelové řádky. To je přirozené, neboť si potřebujeme pamatovat původní účelovou funkci, tj. vztah proměnné x_0, i novou účelovou funkci $\max -x_6$.

Dále budeme postupovat pivotováním obdobně jako v Algoritmu 7.2.

1	0	0	0	-1	0	30
80	50	0	0	0	0	0
20	15	**1**	**0**	0	**0**	1000
4	2	**0**	**1**	0	**0**	160
1	0	**0**	**0**	-1	**1**	30

0	0	0	0	0	-1	0
0	50	0	0	80	-80	-2400
0	15	1	0	20	-20	400
0	2	0	1	4	-4	40
1	0	0	0	-1	1	30

Jak vidíme z poslední tabulky, dosáhli jsme optimálního řešení v první fázi metody, tj. máme minimální přípustnou hodnotu 0 umělé proměnné x_6.

Nyní vymazáním nultého řádku a sloupce umělé proměnné x_6 z poslední tabulky získáme správnou výchozí tabulku původní úlohy v přípustném jednotkovém tvaru.

0	50	0	0	80	-2400
0	15	**1**	**0**	20	400
0	2	**0**	**1**	4	40
1	0	**0**	**0**	-1	30

0	10	0	-20	0	-3200
0	5	1	-5	0	200
0	0.5	0	0.25	1	10
1	0.5	0	0.25	0	40

0	0	0	-25	-20	-3400
0	0	1	-7.5	-10	100
0	1	0	0.5	2	20
1	0	0	0	-1	30

Výsledek výpočtu je $x_1 = 30$, $x_2 = 20$, $x_3 = 100$, $x_4 = x_5 = 0$. Účelová funkce má hodnotu 3400. Porovnejte si tento výsledek s výsledkem v Příkladu 7.5. □

8.3 Degenerace a prevence zacyklení

Jak plyne z definice, degenerovaná řešení se v průběhu simplexové metody objevují, pokud pravá strana tabulky obsahuje 0. Pak nová báze získaná iterací simplexové metody může odpovídat totožnému řešení (vrcholu). Jak pak ale u degenerovaných řešení zabránit zacyklení bází ve stejném vrcholu?

Příklad 8.5. *Ukázka zacyklení Algoritmu 7.2 simplexové metody na úloze LP.*

Použijme k řešení následující úlohy LP Algoritmus 7.2 s tím, že při nejednoznačné volbě sloupcového a řádkového pravidla vybereme podle nejnižšího indexu.

$$\begin{aligned}
\max\ & -20x_1 + 53x_2 + 41x_3 - 204x_4 & & & & \\
& 2x_1 - 11x_2 - 5x_3 + 18x_4 & +x_5 & & = & \ 0 \\
& -x_1 + 4x_2 + 2x_3 - 8x_4 & & +x_6 & = & \ 0 \\
& -2x_1 + 11x_2 + 5x_3 - 18x_4 & & +x_7 & = & \ 1 \\
& x_1, x_2, x_3, x_4 & & x_5, x_6, x_7 & \geq & \ 0
\end{aligned}$$

Čtenář nechť si sám zkusí spočítat několik iterací simplexové metody podle popsané implementace. (Po šesté iteraci uvidí, že získal zpět jednu z předchozích tabulek.) □

Definice 8.6. Lexikografické porovnání vektorů ("zprava"):
Vektor $\vec{x}$ je lexikograficky menší než $\vec{y}$, píšeme $\vec{x} \prec \vec{y}$, pokud

$$\exists i : x_i < y_i \ \wedge \ \forall j > i : x_j = y_j \, .$$

Vektor $\vec{x}$ je *lexikograficky kladný*, pokud $\vec{x} \succ \vec{0}$.

Lexikografické porovnávání již asi čtenář zná jako způsob řazení slov ve slovníku (lexikon) – jednotlivá písmena slov odpovídají složkám vektorů. Je třeba si však uvědomit, že naše definice porovnává zprava, tedy od konců vektorů.

Význam lexikografického porovnání vektorů pro vylepšení Algoritmu 7.2 simplexové metody tkví v následujících poznatcích.

Tvrzení 8.7. *Mějme simplexovou tabulku, jejíž každý řádek mimo účelového je lexikograficky kladný. Zvolme její libovolný sloupec s s kladnou redukovanou cenou.*

a) Je-li v tabulce $a_{i,s} > 0$, pak aplikace pivotu na $a_{i,s}$ lexikograficky zmenší vektor účelového řádku.

b) Je-li $i > 0$ index řádku tabulky s lexikograficky nejmenším vektorem $\frac{1}{a_{i,s}}(\vec{a}_i, b_i)$, pak po aplikaci pivotu na $a_{i,s}$ zůstane každý řádek mimo účelového lexikograficky kladný.

Důkaz: Oba závěry přímočaře plynou z definice lexikografického uspořádání a z významu pivotu. Detaily ponecháme čtenáři. □

Algoritmus 8.8. Lexikografické simplexové metody
Algoritmus 7.2 simplexové metody zpřesníme v následujících dvou bodech.

- V kroku 1 u výchozí tabulky přesuneme sloupce vybrané jednotkové podmatice na konec. (Tím zajistíme lexikografickou kladnost výchozí tabulky.)
- V kroku 4.b řádkového pravidla vybíráme řádek $i > 0$ tabulky s lexikograficky nejmenším vektorem $\frac{1}{a_{i,s}}(\vec{a}_i, b_i)$.

Věta 8.9. *Algoritmus 8.8 lexikografické simplexové metody vždy skončí se správným výsledkem.*

Důkaz: Jelikož podle Tvrzení 8.7 se každou iterací Algoritmu 8.8 lexikograficky zmenší účelový řádek tabulky, nikdy se v průběhu algoritmu stejná tabulka nezopakuje. Jak víme (Oddíl 6.4), přípustná bázická řešení jednoznačně odpovídají přípustným zápisům tabulky úlohy v jednotkovém tvaru. Proto se nezopakuje v průběhu algoritmu ani stejné bázické řešení úlohy. Takže algoritmus musí skončit v konečném čase.

Správnost výsledného řešení úlohy pak plyne z Lematu 6.5. □

8.4 Poznámky k simplexové metodě

Z předchozích Vět 8.2,8.9 plyne hlavní závěr našeho výkladu:

Důsledek 8.10. *Dvoufázová lexikografická simplexová metoda vždy skončí a správně nalezne optimální řešení úlohy, nebo případně potvrdí jeho neexistenci.*

Jak jsme již uvedli, exponenciální horní odhad počtu kroků simplexové metody prostou enumerací všech možných bázických řešení zatím neumíme nijak podstatně vylepšit. Co se týče složitosti nejhoršího případu výpočtu, je známo následující:

Fakt: Pro každá běžná pravidla výběru pivota v simplexové metodě jsou známy protipříklady vyžadující exponenciální počet kroků výpočtu.

Ukázka [H.J. Greenberg,
`http://carbon.cudenver.edu/ hgreenbe/glossary/notes/Klee-Minty.pdf`].

Pro řešení úloh LP jsou známy algoritmy s nejhorší polynomální časovou složitostí (viz Khachiyan, Karmarkar), avšak pro svou jednoduchost a rychlost v běžných praktických výpočtech se stejně nejčastěji používá simplexová metoda.

Na závěr si uděláme malý přehled (nástin) některých dalších užitečných variant implementace simplexové metody. Pro jejich podrobné nastudování čtenáře odkazujeme na doplňkovou literaturu.

- *Redukovaná simplexová metoda*

  Jedná se o paměťově úspornější variantu simplexové metody, ve které samotná jednotková podmatice $\boldsymbol{I}$ v matici $\boldsymbol{A}$ není uložena v tabulce a místo toho bázické proměnné přiřazujeme řádkům.

- *Revidovaná simplexová metoda*

  Její použití je výhodné, když má úloha mnohem více proměnných než nerovností. Pak vůbec nemodifikujeme výchozí tabulku – matici $\boldsymbol{A}$, ale místo toho řádkové úpravy (pivoty) provádíme na $(n+1) \times (n+1)$ matici $\boldsymbol{D}$, která zleva násobí $\boldsymbol{A}$. Tabulka simplexové metody je tedy v každém kroku dána maticí $\boldsymbol{D} \cdot \boldsymbol{A}$.

  Výhodou je, že upravujeme pouze "malou" matici $\boldsymbol{D}$ a celá tabulka se ani nemusí držet v pracovní paměti.

- *Duální simplexová metoda*

  Tato metoda zjednodušeně řečeno prochází přípustná řešení duální úlohy, až najde přípustné primární řešení.