Otázka číslo 33 (A4B33OPT)

Martin Stránský

5. června 2012

"Snažili jsme se udělat to co nejlépe, ale dopadlo to jako vždycky."

1 Lineární programování

Linieární programování znamená řešení úlohy minimalizace lineární funkce $f = c^T x$ $(+d)$ za podmínek affinních funkcí g_i, h_i (affinita viz níže).

Slouží například k řešení úloh (viz skripta):

- optimální výrobní program (z různých druhů surovin vyrábíme různé druhy zboží s různou cenou)
- směšovací problém (kuchařka má uvařit oběd, aby v něm bylo b_1 vitamínů, b_2 bílkovin a b_3 tuků)
- distribuční problém
- dopravní problém
- hledání rovnovážných stavů u lineárních systémů a pod.

Obecná formulace úlohy LP:

$$\begin{array}{lcl}
\min\,(\max) & c_1x_1 + c_2x_2 + \dots c_1x_n & \\
z.p. & a_{i1}x_1 + \dots + a_{in}x_n \geq b_i & i \in I_+ \\
 & a_{i1}x_1 + \dots + a_{in}x_n \leq b_i & i \in I_- \\
 & a_{i1}x_1 + \dots + a_{in}x_n = b_i & i \in I_0 \\
 & x_j \geq 0 & j \in J_+ \\
 & x_j \leq 0 & j \in J_- \\
 & x_j \in \mathbb{R} & j \in J_0
\end{array}$$

Obecnou formulaci převádíme na standardní tvar

$$\begin{array}{lcl}
\min & c_1x_1 + c_2x_2 + \dots c_1x_n & \\
z.p. & a_{i1}x_1 + \dots + a_{in}x_n = b_i & i = 1, \dots, m \\
 & x_j \geq 0 & j = 1, \dots, n
\end{array}$$

zkráceně

$$\min\{c^T x \mid Ax = b,\ x > 0\}$$

a zpět následovně:

- Maximalizaci nahradíme minimalizací podle $\min_{x \in X} f(x) = -\max_{x \in X}(-f(x))$.
- ($a_{i1}x_1 + \ldots + a_{in}x_n = b_i$ nahradíme $a_{i1}x_1 + \ldots + a_{in}x_n \leq b_i$ a $a_{i1}x_1 + \ldots + a_{in}x_n \geq b_i$.)
- $a_{i1}x_1 + \ldots + a_{in}x_n \leq b_i$ doplníme slackovou proměnnou $u_i \geq 0$, $a_{i1}x_1 + \ldots + a_{in}x_n + u_i = b_i$.
- $a_{i1}x_1 + \ldots + a_{in}x_n \geq b_i$ vynásobíme -1 a doplníme slackovou proměnnou $u_i \geq 0$, $-a_{i1}x_1 - \ldots - a_{in}x_n + u_i = -b_i$.
- $x_i \in \mathbb{R}$ převedeme na $x_i^+ \geq 0$, $x_i^- \geq 0$, $x_i = x_i^+ - x_i^-$.

Příklad (vlastní): převedení z obecného tvaru na standardní.

$$\begin{array}{rc} \max & c^T x \\ z.p. & a_{11}x_1 + \ldots + a_{2n}x_n \geq b_i \\ & a_{21}x_1 + \ldots + a_{2n}x_n \leq b_i \\ & a_{i1}x_1 + \ldots + a_{in}x_n = b_i \\ & x_1, \ldots, x_n \geq 0 \end{array}$$

se převede na

$$\begin{array}{rcc} -\min & -c_1x_1 \ldots - c_nx_n & \\ z.p. & -a_{11}x_1 - \ldots - a_{2n}x_n + u_1 = -b_i & \\ & a_{21}x_1 + \ldots + a_{2n}x_n + u_2 = b_i & \\ & a_{i1}x_1 + \ldots + a_{in}x_n = b_i & i \in [3, m] \\ & x_1, \ldots, x_n, u_1, u_2 \geq 0 & \end{array}$$

Ještě poznámka k řešení přeurčených rovnic $Ax = b$, $A \in \mathbb{R}^{m \times n}$ $(m > n)$. Řeší se taková úloha:

$$\min\{||Ax - b||_p \mid x \in \mathbb{R}\}.$$

Dualita

Každá úloha LP má svojí duální úlohu, kterou lze dostat následující konstrukcí. Z duální úlohy lze poté tím samým postupem dostat primární. Popíšu jenom konstukci ze speciálního tvaru, více ve skriptech (s. 95):

$$\begin{array}{cccc} \min & \sum_j c_j x_j & \max & \sum_i y_i b_i \\ z.p. & \sum_j a_{ij} x_j \geq b_i & z.p. & y_i \geq \mathbb{R} \\ & x_j \geq 0 & & \sum_i y_i a_{ij} \leq c_j \end{array}$$

přehledněji:

$$\begin{array}{cccc} \min & c^T x & \max & y^T b \\ z.p. & Ax \geq b & z.p. & y_i \geq \mathbb{R} \\ & x \geq 0 & & y^T A \leq c^T \end{array}$$

U vět budeme postupovat přesně opačně, než jak je tomu ve skriptech:

Důležitá je zejména silná věta o dualitě která říká, že když se $c^T x = y^T b$ (kritéria) pro přípustná x, y ($Ax \geq b$ atd.), pak jsou x i y optimálními řešeními obou úloh.

Přitom platí (věta o slabé dualitě), že $c^T x \geq y^T b$ pro jakékoli přípustné x, y; protože $y^T A \leq c^T$ vynásobeno zprava $x \geq 0$ rovná se $y^T Ax \leq c^T x$ (a stejně tak pro x), takže napsáno v řadě $c^T x \geq y^T Ax \geq y^T b$.

Věta o komplementaritě udává podmínky, kdy ($\Leftrightarrow$) se $c^T x = y^T b$, více ve skriptech na s. 96.

Dualita je dobrá k:

- lepšímu pochopení problému
- ověření optimality
- někdy lze jednodušeji spočítat duální úlohu a z ní řešení primární, než rovnou primární.

Takhle shrnuto to snad stačí.

2 Simplexový algoritmus

Řeší úlohu lineárnho programování ve standatdním tvaru $\min\{c^T x - d \mid Ax = b,\ x > 0\}$.

2.1 Části algoritmu

2.1.1 Přechod k sousední bázi

Co nejjednodušeji: máme soustavu $[A \mid b]$ tak, že v m *bázových* sloupcích jedna jednička a samé nuly a na každém řádku je alepoň jedna jednička.

$$\begin{array}{ccccccc} 5 & 0 & 0 & -1 & 1 & | & 3 \\ 3 & 0 & 1 & 2 & 0 & | & 5 \\ -1 & 1 & 0 & 2 & 0 & | & -2 \end{array}$$

Chceme přejít k sousední bázi, která bude mít jeden jiný bázový sloupec, to znamená, že jestliže jsou teď báze sloupce 2, 3, 5, pak příště to bude například 1, 3, 5 nebo 2, 3, 4.

Zvolme pivot a_{ij} (níže) a vydělme řádek i a_{ij} (dostaneme na místo pivotu jedničku). Dále pomocí násobků ostatních řádků vynulujeme prvky ve sloupcích s bázemi (jako v GEM). Zbylé báze mají ve sloupci jedničku, takže to jde dobře.

2.1.2 Přípustné bázové řešení

Protože nenulové složky bázového řešení jsou rovny složkám vektoru b, je bázové řešení přípustné tehdy, když $b \geq 0$. Pakliže teď máme přípustné řešení, k tomuto nedojde když $a_{ij} \geq 0$ kde a_{ij} je pivot a pro každé $i' \neq i$ platí $a_{i'j} \leq 0$ nebo $b_i / a_{ij} \leq b_{i'} / a_{i'j}$

2.1.3 Nekladný sloupec

Pakliže jeden z nebázových sloupců obsahuje jen nekladné prvky, leží optimum v $-\infty$ - úloha je neomezená.

2.1.4 Úpravy účelového řádku

Simplexová tabulka:

$$\begin{bmatrix} c & d \\ A & b \end{bmatrix}$$

Přičtení jakékoli lineární kombinace z $[A\, d]$ k $[c\, d]$ zachová hodnotu účelové funkce.

2.2 Algoritmus

1. Vyber pivot (kde výběrem $j \in \text{argmin}\, c_j$ se zajistí, že účelová funkce nestoupne). i se vybírá například argmin b_i/a_{ij}.
2. Udělej ekvivalentní úpravu.
3. Vynuluj c_j pro bázová j.
4. Končíme, když všechny koeficienty c_j jsou nezáporné (optimum) nebo když máme nekladný sloupec (neomezená úloha).

3 Něco o tom zbytku

Konvexní množinu tvoří konvexní kombinace vektorů (je definována tak, že je uzavřená na konvexní kombinace). Konvexní kombinace je současně affinní a nezáporná kombinace: $\sum_k \alpha \geq 1 \wedge \forall k[\alpha_k \geq 0]$.

Konvexní polyedr je definován jako průnik konečně mnoha poloprostorů (H-reprezentace (half-space)). Extrémní body jsou vrcholy polyedru a jejich specifikem je to, že je lze dostat právě jednou konvexní kombinací. Reprezentovat jde ještě jako konvexní obal konečně mnoha vektorů (V-reprezentace (vertex)).

Funkce je konvexní právě tehdy, když $f(\alpha x + (1-\alpha y)) = \alpha f(x) - (1-\alpha) f(y)$.

Konvexní optimalizační úlohu řešíme tehdy, když je účelová funkce konvexní. Nejdůležitější je, že nalezené (lokální) minimum je vždy zároveň globální. Například u simplexové metody postupujeme po vrcholech konvexního polyedru k tomu nejnižšímu. K konvexním optimalizačním úlohám patří LP a QP (qudratic, také vytváří konvexní množiny).