Problém lineární komplementarity a kvadratické programování (stručný učební text)[1]

J. Rohn

Univerzita Karlova

Matematicko-fyzikální fakulta

Verze: 17. 6. 2002

[1]Sepsání tohoto textu bylo podpořeno Grantovou agenturou České republiky z grantu 201/01/0343

Obsah

Kapitola 1

Problém lineární komplementarity

1.1 Formulace

Problémem lineární komplementarity (LCP)[1] nazýváme problém

$$y = Mz + q, \tag{1.1}$$

$$y \geq 0,\ z \geq 0, \tag{1.2}$$

$$y^T z = 0, \tag{1.3}$$

kde M je čtvercová matice $n \times n$. Za předpokladu (1.2) lze psát (1.3) ekvivalentně ve tvaru

$$y_j z_j = 0 \qquad (j = 1, \ldots, n),$$

takže pro každé j má aspoň jedna z proměnných y_j, z_j nulovou hodnotu. Pro každé j nazýváme proměnné y_j, z_j doplňkové (komplementární). Odtud je odvozen i název úlohy. LCP není optimalizační problém, protože neobsahuje žádnou účelovou funkci: jde nám pouze o nalezení řešení soustavy (1.1)-(1.3), která však vzhledem k podmínce (1.3) je nelineární.

1.2 Lemkeho algoritmus

Pro řešení LCP (1.1)-(1.3) existuje řada metod. My zde uvedeme tzv. Lemkeho algoritmus, který používá tabulku simplexového typu. Místo základního problému (1.1)-(1.3) řeší modifikovaný problém

$$y = Mz + z_0 e + q, \tag{1.4}$$

$$y \geq 0,\ z \geq 0,\ z_0 \geq 0, \tag{1.5}$$

$$y^T z = 0, \tag{1.6}$$

[1] z angl. „linear complementarity problem“

kde z_0 je přidaná dodatečná proměnná a $e = (1, \ldots, 1)^T$. Zavedení proměnné z_0 má za účel dosažení nezápornosti sloupce pravých stran v tabulce hned v prvním kroku algoritmu. Algoritmus pak v každém kroku udržuje řešení modifikovaného problému a směřuje k dosažení $z_0 = 0$, potom řešení y, z problému (1.4)-(1.6) se stává řešením původního problému (1.1)-(1.3).

Algoritmus pracuje s tabulkou simplexového typu

$$\begin{array}{|c||c|c|}\hline B & \overline{b} & \qquad \overline{A} \qquad \\ \hline \end{array} \tag{T}$$

(bez kriteriálního řádku), kde B je opět sloupec indexů bázických proměnných. Z důvodů, které vyplynou z dalšího, je však sloupec pravých stran $\overline{b}$ zapisován nalevo od bloku $\overline{A}$. Převedeme-li v (1.4) všechny proměnné na levou stranu, dostáváme rovnici

$$-z_0 e + y - Mz = q,$$

takže tabulka je inicializována hodnotami

$$\overline{A} = (-e, I, -M), \tag{1.7}$$
$$\overline{b} = q, \tag{1.8}$$

které odpovídají počátečnímu řešení $z_0 = 0$, $y = q$, $z = 0$. Jelikož proměnné

$$z_0, y_1, \ldots, y_n, z_1, \ldots, z_n$$

číslujeme, tak jak je to obvyklé u simplexového algoritmu, v pořadí

$$x_1, \ldots, x_{2n+1}, \tag{1.9}$$

pokládáme na začátku, kdy v bázi jsou proměnné $y_1, \ldots, y_n$,

$$B = (2, \ldots, n+1)^T. \tag{1.10}$$

Algoritmus postupuje po tzv. téměř komplementárních bázích: bázické řešení z_0, y, z se nazývá téměř komplementární, jestliže existuje $k \in \{1, \ldots, n\}$ tak, že

(i) pro každé $j \in \{1, \ldots, n\}$, $j \neq k$ je právě jedna z proměnných y_j, z_j v bázi[2],
(ii) y_k ani z_k nejsou v bázi,
(iii) z_0 je v bázi.

Tím je zaručeno splnění podmínky (1.6).

[2] striktně vzato jsou v bázi *indexy* bázických proměnných ve smyslu očíslování (1.9); pro větší srozumitelnost však říkáme, že proměnná je v bázi (resp. vstupuje do báze, vystupuje z báze), jestliže to platí pro její index

Algoritmus (Lemke 1965)

0. Sestav tabulku (T) s počátečními hodnotami (1.7), (1.8), (1.10). Je-li $\bar{b} \geq 0$, ukonči: $y = q$, $z = 0$ je řešením problému (1.1)-(1.3).
1. Jinak nalezni $t = \max\{k;\ \bar{b}_k = \min_j \bar{b}_j\}$, zaveď z_0 do báze s pivotem v řádku t a polož $B_t := 1$.
2. Nechť $\overline{A}_s$ je sloupec proměnné **doplňkové** k té, která právě vystoupila z báze[3]. Je-li $\overline{A}_s \leq 0$, ukonči: algoritmus selhává.
3. Jinak urči r ze vzorce
$$\frac{\bar{b}_r}{\bar{a}_{rs}} = \min\left\{\frac{\bar{b}_j}{\bar{a}_{js}};\ \bar{a}_{js} > 0\right\}, \quad \bar{a}_{rs} > 0.$$
4. Proveď eliminaci s pivotem $\bar{a}_{rs}$ a polož $B_r := s$.
5. Je-li $z_0 = 0$, ukonči: $y = (x_2, \ldots, x_{n+1})^T$, $z = (x_{n+2}, \ldots, x_{2n+1})^T$ je řešením[4] problému (1.1)-(1.3). Jinak jdi na krok 2.

Neskončí-li algoritmus v kroku 0, potom v kroku 1 je $\bar{b}_t = \min_j \bar{b}_j < 0$ a tedy na konci kroku 1 je $\bar{b}_t := -\bar{b}_t > 0$ a $\bar{b}_j := \bar{b}_j - \bar{b}_t \geq 0$ pro $j \neq t$, takže $\bar{b} \geq 0$. Algoritmus udržuje téměř komplementární řešení. Po provedení kroku 1 jsou y_j, $j \neq t$ v bázi a z_0 je v bázi, přičemž y_t vystoupila z báze, takže dvojice y_t, z_t není v bázi. Dále indukcí: je-li to v jistém kroku splněno a y_k, z_k je nebázická dvojice, potom v dalším kroku jedna z těchto proměnných vstupuje do báze a některá jiná (y_j nebo z_j, $j \neq k$) vystupuje z báze, čímž vzniká nová nebázická dvojice y_j, z_j. Ze známých vlastností simplexové tabulky plyne tato věta:

Věta 1 *V běžném kroku algoritmu (tj. po provedení kroku 4) má tabulka (T) tvar*
$$\begin{aligned}\overline{A} &= (-A_B^{-1}e, A_B^{-1}, -A_B^{-1}M),\\ \bar{b} &= A_B^{-1}q,\end{aligned}$$
kde A_B je matice jejíž j-tý sloupec je roven B_j-tému sloupci matice $(-e, I, -M)$ $(j = 1, \ldots, n)$, a dává průběžné řešení $z_0 = x_1$, $y = (x_2, \ldots, x_{n+1})^T$, $z = (x_{n+2}, \ldots, x_{2n+1})^T$, kde $x_{B_j} = \bar{b}_j$ pro $j = 1, \ldots, n$ a $x_j = 0$ pro $j \notin B$.

Nevýhodou Lemkeho algoritmu je možnost jeho selhání v kroku 2 z důvodu nenalezení kladného prvku v s-tém sloupci, potřebného dále v kroku 3. Na rozdíl od simplexového algoritmu, který na tomto místě indikuje neomezenost účelové funkce, nedává Lemkeho algoritmus žádnou odpověď a zastaví se (LCP může mít řešení, ale algoritmus ho nenalezne). Vzorec pro výpočet r v kroku 3 je typické „podílové pravidlo“ simplexového algoritmu, které slouží k udržení nezápornosti vektoru $\bar{b}$. Konečnost Lemkeho algoritmu v této podobě není zaručena (může dojít k zacyklení) a k jejímu dosažení je třeba zesílit formulaci kroku 3, jak bude uvedeno dále.

[3] tj. jestliže y_j vystoupila z báze, je to sloupec proměnné z_j; jestliže z_j vystoupila z báze, je to sloupec proměnné y_j $(j = 1, \ldots, n)$

[4] připomeňme, že tak jako u simplexové metody je bázické řešení x v tabulce s bází B dáno předpisem $x_{B_j} = \bar{b}_j$ $(j = 1, \ldots, n)$ a $x_j = 0$ pro $j \notin B$

1.3 Lexikografické pravidlo

Pro vektory $x, y \in R^n$, $x \neq y$, definujme $x \prec y$ jestliže platí $x_k < y_k$, kde $k = \min\{j;\, x_j \neq y_j\}$. Toto uspořádání se nazývá lexikografické (na principu uspořádání slov ve slovníku). Je zřejmé, že pro každé $x, y \in R^n$ nastává právě jedna z možností $x \prec y$, $y \prec x$, $x = y$, takže každá konečná množina $X \subset R^n$ obsahuje v tomto uspořádání nejmenší prvek, který značíme $\operatorname{lexmin} X$.

V tabulce (T) Lemkeho algoritmu

$$\begin{array}{|c||c|c|}\hline B & \overline{b} & \quad\overline{A}\quad \\ \hline\end{array}$$

označme symbolem β_j j-tý řádek její části

$$\begin{array}{|c|c|}\hline \overline{b} & \quad\overline{A}\quad \\ \hline\end{array}$$

tj.

$$\beta_j = (\overline{b}_j, \overline{a}_{j1}, \ldots, \overline{a}_{j,2n+1}) \qquad (j = 1, \ldots, n),$$

a nahraďme krok 3 algoritmu tímto tzv. lexikografickým pravidlem:

3*. Jinak urči r ze vzorce

$$\frac{\beta_r}{\overline{a}_{rs}} = \operatorname{lexmin}\left\{\frac{\beta_j}{\overline{a}_{js}};\, \overline{a}_{js} > 0\right\}, \quad \overline{a}_{rs} > 0.$$

Jak je vidět, vznikne nový vzorec ze starého formálně nahrazením čísel $\overline{b}_r$, $\overline{b}_j$ vektory β_r, β_j a minima lexikografickým minimem. Ukážeme, že při nahrazení kroku 3 krokem 3* se Lemkeho algoritmus stane konečným. K tomu nejprve zformulujeme tři základní vlastnosti lexikografického pravidla. Důkaz tohoto i dalších tvrzení této kapitoly vynecháváme.

Věta 2 *Při použití lexikografického pravidla platí:*

(i) *v každém kroku je výběr r jednoznačný,*

(ii) *v každém kroku platí $0 \prec \beta_j$ pro všechna j,*

(iii) *jestliže s vstoupí do báze a B_r vystoupí z báze, potom zavedeme-li v dalším kroku B_r do báze, potom s opět vystoupí z báze a tabulka se vrátí do původního stavu.*

1.4 Konečnost Lemkeho algoritmu

Použití lexikografického pravidla zaručuje konečnost algoritmu:

Věta 3 *Lemkeho algoritmus s použitím lexikografického pravidla je konečný (tj. po konečně mnoha krocích buď dává řešení LCP, nebo selhává, ačkoliv LCP může mít řešení).*

1.5 Lemkeho algoritmus pro pozitivně semidefinitní matice

V této části ukážeme, že z hlediska Lemkeho algoritmu hrají důležitou roli pozitivně semidefinitní matice. Matice $M \in R^{n \times n}$ se nazývá pozitivně semidefinitní jestliže $x^T M x \geq 0$ pro každé $x \in R^n$. Pro důkaz hlavní věty je potřebné následující pomocné tvrzení:

Věta 4 *Nechť M je pozitivně semidefinitní a nechť $x_0^T M x_0 = 0$ pro jisté x_0. Potom $(M + M^T)x_0 = 0$.*

Věta 5 *Nechť M je pozitivně semidefinitní a nechť soustava*

$$y = Mz + q, \tag{1.11}$$

$$y \geq 0,\ z \geq 0 \tag{1.12}$$

má řešení. Potom LCP (1.1)-(1.3) má řešení.

Povšimněme si, že soustava (1.11)-(1.12) vznikne z LCP (1.1)-(1.3) vynecháním nelineární podmínky (1.3). Řešitelnost soustavy (1.11)-(1.12) je proto možno ověřit fází I simplexového algoritmu.

Věta 6 *Je-li M pozitivně semidefinitní a má-li LCP (1.1)-(1.3) řešení, potom Lemkeho algoritmus nalezne jeho řešení (tj. nedojde k selhání).*

Pro pozitivně semidefinitní matice M je tedy selhání Lemkeho algoritmu ekvivalentní neexistenci řešení problému (1.1)-(1.3). V další kapitole ukážeme, že Lemkeho algoritmus je možno použít k řešení úlohy kvadratického programování.

Kapitola 2

Kvadratické programování

2.1 Úloha kvadratického programování

Úlohou kvadratického programování nazýváme problém

$$\min\left\{\tfrac{1}{2}x^TDx + c^Tx;\, Ax \geq b, x \geq 0\right\} \tag{QP}$$

kde $A \in R^{m\times n}$, $b \in R^m$, $c \in R^n$, $D \in R^{n\times n}$ (pro $D = 0$ je to úloha lineárního programování). Jelikož $x^TDx = x^T\left(\frac{1}{2}(D + D^T)\right)x$ pro každé x, lze předpokládat že D je symetrická (jinak dosadíme $D := \frac{1}{2}(D+D^T)$ a účelová funkce se nezmění). Navíc budeme předpokládat, že D je pozitivně semidefinitní. Označme $Q(x) = \frac{1}{2}x^TDx+c^Tx$. Potom pro každé x, x^* platí

$$Q(x) = Q(x^*) + (Dx^* + c)^T(x - x^*) + \tfrac{1}{2}(x - x^*)^TD(x - x^*) \tag{2.1}$$

(fakticky jde o Taylorův rozvoj funkce $Q(x)$). Skutečně, roznásobením pravé strany dostáváme $\frac{1}{2}x^{*T}Dx^* + c^Tx^* + x^TDx^* - x^{*T}Dx^* + c^Tx - c^Tx^* + \frac{1}{2}x^TDx - \frac{1}{2}x^TDx^* - \frac{1}{2}x^{*T}Dx + \frac{1}{2}x^{*T}Dx^* = \frac{1}{2}x^TDx + c^Tx = Q(x)$. Množinu $X = \{x;\, Ax \geq b, x \geq 0\}$ nazýváme množinou přípustných řešení úlohy (QP).

2.2 Převedení na LCP

Věta 7 *Nechť D je symetrická pozitivně semidefinitní. Potom x^* je optimální řešení (QP) právě když je optimálním řešením úlohy lineárního programování*

$$\min\left\{(Dx^* + c)^Tx;\, Ax \geq b, x \geq 0\right\}. \tag{LP}$$

Poznámka Tato věta má čistě teoretický význam, protože účelová funkce úlohy (LP) obsahuje hledaný vektor x^*, takže ji bez jeho znalosti nelze sestavit. Nicméně slouží jako východisko k důkazu další věty, v níž už je tento nedostatek odstraněn.

Důkaz Úlohy (QP) a (LP) mají stejnou množinu přípustných řešení. Nechť x^* je optimální řešení (QP). Potom pro libovolné přípustné řešení x úlohy (QP) a libovolné $\lambda \in [0,1]$ z konvexity množiny přípustných řešení plyne, že $x^* + \lambda(x - x^*)$ je opět přípustné řešení a podle (2.1)

$$Q(x^*) \le Q(x^* + \lambda(x - x^*)) = Q(x^*) + \lambda(Dx^* + c)^T(x - x^*) + \tfrac{1}{2}\lambda^2(x - x^*)^T D(x - x^*),$$

tedy

$$(Dx^* + c)^T(x - x^*) + \tfrac{1}{2}\lambda(x - x^*)^T D(x - x^*) \ge 0$$

pro každé $\lambda \in (0,1]$, z čehož limitním přechodem pro $\lambda \to 0_+$ dostáváme

$$(Dx^* + c)^T(x - x^*) \ge 0, \tag{2.2}$$

tj.

$$(Dx^* + c)^T x^* \le (Dx^* + c)^T x, \tag{2.3}$$

kde x je libovolné přípustné řešení (LP), což znamená, že x^* je optimální řešení (LP). Naopak, nechť x^* je optimální řešení (LP). Potom platí (2.3) a tedy i (2.2) pro libovolné přípustné řešení x úlohy (QP) a s ohledem na pozitivní semidefinitnost matice D je podle (2.1)

$$Q(x) = Q(x^*) + (Dx^* + c)^T(x - x^*) + \tfrac{1}{2}(x - x^*)^T D(x - x^*) \ge Q(x^*),$$

tedy x^* je optimální řešení (QP). □

Věta 8 *Nechť D je symetrická pozitivně semidefinitní. Potom x^* je optimálním řešením (QP) právě když existuje řešení y, z problému lineární komplementarity*

$$\begin{aligned} y &= Mz + q, \\ y &\ge 0,\ z \ge 0, \\ y^T z &= 0, \end{aligned} \tag{LCP$_{\mathrm{QP}}$}$$

kde

$$M = \begin{pmatrix} D & -A^T \\ A & 0 \end{pmatrix}, \quad q = \begin{pmatrix} c \\ -b \end{pmatrix},$$

v němž z je tvaru $\begin{pmatrix} x^* \\ p^* \end{pmatrix}$ pro jisté $p^* \in R^m$.

Poznámka Je $M \in R^{(n+m)\times(n+m)}$, $y, z, q \in R^{n+m}$. Pro $D = 0$ dostáváme novou charakteristiku optimálního řešení úlohy lineárního programování.

Důkaz 1) Nechť x^* je optimální řešení (QP), potom je podle věty 7 i optimálním řešením úlohy

$$\min\left\{(Dx^* + c)^T x;\ Ax \ge b, x \ge 0\right\}$$

a tedy podle věty o dualitě existuje optimální řešení p^* k ní duální úlohy

$$\max\left\{b^T p;\ A^T p \le Dx^* + c, p \ge 0\right\},$$

takže platí $Ax^* \ge b$, $x^* \ge 0$, $Dx^* - A^T p^* + c \ge 0$, $p^* \ge 0$, a navíc podle podmínek optimality

$$x^{*T}(Dx^* - A^T p^* + c) = 0,$$
$$p^{*T}(Ax^* - b) = 0.$$

Označíme-li

$$y = \begin{pmatrix} Dx^* - A^T p^* + c \\ Ax^* - b \end{pmatrix}, \quad z = \begin{pmatrix} x^* \\ p^* \end{pmatrix},$$

je $y \ge 0$, $z \ge 0$, $y^T z = 0$ a

$$y = \begin{pmatrix} D & -A^T \\ A & 0 \end{pmatrix} z + \begin{pmatrix} c \\ -b \end{pmatrix},$$

tj. y a z jsou řešením (LCP$_{\text{QP}}$). Naopak, jsou-li y, z řešením (LCP$_{\text{QP}}$) a píšeme-li vektor $z \in R^{n+m}$ ve tvaru $z = \begin{pmatrix} x^* \\ p^* \end{pmatrix}$, kde $x^* \in R^n$, $p^* \in R^m$, potom platí

$$Dx^* - A^T p^* + c \ge 0,\ Ax^* - b \ge 0,\ x^* \ge 0,\ p^* \ge 0,$$

$$x^{*T}(Dx^* - A^T p^* + c) + p^{*T}(Ax^* - b) = 0,$$

a vzhledem k tomu, že oba sčítanci jsou nezáporní, plyne odsud

$$x^{*T}(Dx^* - A^T p^* + c) = p^{*T}(Ax^* - b) = 0,$$

což jsou podmínky optimality pro (LP) a úlohu k ní duální. Tedy x^* je optimálním řešením (LP) a podle věty 7 i optimálním řešením (QP). □

2.3 Aplikace Lemkeho algoritmu

Věta 9 *Nechť D je symetrická pozitivně semidefinitní. Potom:*

1. *Má-li (QP) optimální řešení, potom se nalezne Lemkeho algoritmem aplikovaným na úlohu (LCP$_{\text{QP}}$) (tj. nedojde k jeho selhání).*
2. *Dojde-li při řešení (LCP$_{\text{QP}}$) Lemkeho algoritmem k selhání, potom (QP) buď nemá přípustné řešení, nebo její účelová funkce je na množině přípustných řešení neomezená zdola.*

Důkaz 1. Matice

$$M = \begin{pmatrix} D & -A^T \\ A & 0 \end{pmatrix}$$

v problému ($\mathrm{LCP_{QP}}$) je pozitivně semidefinitní, neboť pro každé $x \in R^n$, $p \in R^m$ platí

$$\begin{pmatrix} x \\ p \end{pmatrix}^T \begin{pmatrix} D & -A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ p \end{pmatrix} = \begin{pmatrix} x \\ p \end{pmatrix}^T \begin{pmatrix} Dx - A^T p \\ Ax \end{pmatrix} = x^T Dx \geq 0.$$

Má-li (QP) optimální řešení, potom podle věty 8 má ($\mathrm{LCP_{QP}}$) řešení a podle věty 6 nedojde při řešení ($\mathrm{LCP_{QP}}$) Lemkeho algoritmem k selhání a z nalezeného řešení y, $z = \begin{pmatrix} x^* \\ p^* \end{pmatrix}$ podle věty 8 dostáváme optimální řešení x^* problému (QP).

2. Nechť tedy Lemkeho algoritmus skončí selháním a předpokládejme, že (QP) je přípustná. Dokážeme, že (QP) je v tom případě neomezená. Jelikož algoritmus končí selháním, nemá ($\mathrm{LCP_{QP}}$) podle věty 6 řešení a tedy podle věty 5 nemá řešení ani soustava

$$y = \begin{pmatrix} D & -A^T \\ A & 0 \end{pmatrix} z + \begin{pmatrix} c \\ -b \end{pmatrix}, \quad \begin{pmatrix} y \\ z \end{pmatrix} \geq 0,$$

tj. soustava

$$\begin{pmatrix} I & 0 & -D & A^T \\ 0 & I & -A & 0 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} c \\ -b \end{pmatrix}, \quad \begin{pmatrix} y_1 \\ y_2 \\ z_1 \\ z_2 \end{pmatrix} \geq 0.$$

To podle Farkasovy věty znamená, že existují d_1, d_2 tak, že

$$d_1 \geq 0,\ d_2 \geq 0,\ Dd_1 + A^T d_2 \leq 0,\ Ad_1 \geq 0,\ c^T d_1 - b^T d_2 < 0.$$

Dokážeme, že z toho plyne

$$Dd_1 = 0,\ c^T d_1 < 0.$$

Především

$$0 \geq d_1^T (Dd_1 + A^T d_2) = d_1^T Dd_1 + (Ad_1)^T d_2 \geq 0$$

jelikož $d_1^T Dd_1 \geq 0$, $Ad_1 \geq 0$ a $d_2 \geq 0$, tj. $d_1^T Dd_1 = 0$ a odtud $Dd_1 = 0$ (věta 4) takže $A^T d_2 \leq 0$. Dále z přípustnosti soustavy $Ax \geq b$, $x \geq 0$ plyne opět podle Farkasovy věty (aplikované na soustavu $Ax - x' = b$, $x \geq 0$, $x' \geq 0$), že pro každé $y \leq 0$ takové, že $A^T y \geq 0$, je $b^T y \geq 0$. Protože $-d_2 \leq 0$, $A^T(-d_2) \geq 0$, dostáváme odsud $b^T(-d_2) \geq 0$, tedy z $c^T d_1 - b^T d_2 < 0$ vyplývá $c^T d_1 < b^T d_2 \leq 0$ a nakonec $c^T d_1 < 0$.

Nyní, z přípustnosti (QP) plyne, že $Ax \geq b$, $x \geq 0$ má řešení $x_0 \geq 0$. Potom pro každé $\lambda \geq 0$ je $A(x_0 + \lambda d_1) = Ax_0 + \lambda Ad_1 \geq Ax_0 \geq b$, přičemž $x_0 + \lambda d_1 \geq 0$, tedy $x_0 + \lambda d_1$ je přípustné řešení pro každé $\lambda \geq 0$ a pro hodnotu účelové funkce dostáváme s přihlédnutím k faktu že $Dd_1 = 0$,

$$\begin{aligned} Q(x_0 + \lambda d_1) &= Q(x_0) + \lambda (Dx_0 + c)^T d_1 + \tfrac{1}{2} \lambda^2 d_1^T Dd_1 \\ &= Q(x_0) + \lambda c^T d_1 \stackrel{\lambda \to \infty}{\longrightarrow} -\infty, \end{aligned}$$

tedy $Q(x)$ je na množině přípustných řešení neomezená zdola. $\square$

Poznámka Přípustnost úlohy (QP) lze testovat fází I simplexového algoritmu. Existují tedy tři možnosti ukončení jako u lineárního programování.

2.4 Algoritmus

Shrnutím předchozích faktů dostáváme tento algoritmus pro řešení (QP) se symetrickou pozitivně semidefinitní maticí D:

0. Sestav úlohu (LCP$_{\mathrm{QP}}$) a řeš ji Lemkeho algoritmem.
1. Jestliže algoritmus dává řešení y, $z = \begin{pmatrix} x^* \\ p^* \end{pmatrix}$, ukonči: x^* je optimální řešení (QP).
2. Skončí-li Lemkeho algoritmus selháním, ověř neprázdnost množiny přípustných řešení $X = \{x; Ax - x' = b, x \geq 0, x' \geq 0\}$ úlohy (QP) fází I simplexového algoritmu.
3. Je-li $X = \emptyset$, ukonči: (QP) je nepřípustná.
4. Je-li $X \neq \emptyset$, ukonči: (QP) je neomezená.