VÉGES CRISS-CROSS MÓDSZER A HIPERBOLIKUS PROGRAMOZÁSI FELADATRA[1]

ILLÉS TIBOR – SZIRMAI ÁKOS – TERLAKY TAMÁS
Eötvös Lóránd Tudományegyetem – Delft University of Technology

Cikkünkben a hiperbolikus (hányados) programozási feladat megoldására általánosítjuk a criss-cross módszert. A lineáris és a kvadratikus programozási problémákra megfogalmazott criss-cross algoritmusokhoz hasonlóan a hiperbolikus criss-cross módszer is tetszőleges, nem feltétlenül megengedett bázismegoldásból indítható. A hiperbolikus programozásban szokásos feltételek mellett bizonyítjuk az algoritmus végességét.

1. Bevezetés

A hiperbolikus (hányados) programozási feladat természetes általánosítása a lineáris programozási feladatnak. A lineáris feltételeket megtartva, lineáris célfüggvény szélsőértéke helyett két lineáris függvény hányadosát optimalizáljuk. Ilyen hányados függvények természetes módon jelentkeznek a közgazdaságtanban, amikor a profit/ráfordítás típusú célfüggvények optimalizálása a cél (lásd pl. [8]).

Annak ellenére, hogy a hiperbolikus programozás célfüggvénye nem lineáris, sőt nem is konvex, a feladat megoldására hatékony algoritmusok ismertek. Ez az alábbi eredménynek köszönhető. Habár a célfüggvény nem konvex, de mint azt Martos [9] bizonyította, a célfüggvény pszeudokonvex, sőt pszeudolineáris, ami elégséges feltétele annak, hogy minden lokális minimum egyben globális is legyen. A másik kedvező tulajdonság azonosítása Charnes és Cooper [4] nevéhez fűződik. Eszerint a hiperbolikus programozási feladat átfogalmazható egy ekvivalens lineáris programozási feladattá. Így nem meglepő, hogy a különböző lineáris programozási algoritmusok megfelelő adaptálás után alkalmassá válnak a hiperbolikus programozási feladat megoldására. Így a szimplex módszerek adaptálása után [3,4,5] a közelmúltban Karmarkar belsőpontos algoritmusát is általánosították [1,2] hányados programozási feladatok megoldására. A hányados programozás eredményeinek áttekintése és egy teljességre törekvő, majd 1200 elemet tartalmazó irodalomjegyzéke található Schaible [10] cikkében.

[1] Beérkezett 1996. február 18.

Cikkünkben Terlaky [11,6] criss-cross algoritmusát általánosítjuk hányados programozási feladatok megoldására. Az általánosítás megőrzi a criss-cross módszerek azon alapvető tulajdonságát, hogy tetszőleges, nem feltétlenül megengedett bazismegoldásból indítható. A továbbiakban bizonyítjuk az algoritmus végességét az irodalomban [7,9] szokásos enyhe feltételek mellett. Végül két, az irodalomból vett egyszerű numerikus példa megoldásával illusztráljuk az algoritmus lényeges lépéseit.

A következő fejezetben a hányados programozási feladat megfogalmazása után röviden összefoglaljuk a hiperbolikus programozási feladat alaptulajdonságait. Továbbá tárgyaljuk a feladathoz rendelt pivot tábla előjelstruktúrája és a feladat megoldása, nem korlátossága illetve nem megengedettsége közti kapcsolatot.

2. A hiperbolikus programozási feladat alaptulajdonságai

A továbbiakban az alábbi hiperbolikus programozási feladattal foglalkozunk:

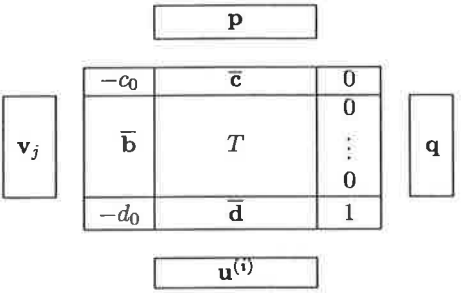
$$\left.\begin{array}{lcl} & \min & \dfrac{\mathbf{c}^T\mathbf{x}}{\mathbf{d}^T\mathbf{x}} \\ A\mathbf{x} & = & \mathbf{b} \\ \mathbf{x} & \geq & \mathbf{0} \end{array}\right\} \quad \text{(P)}\,,$$

ahol $\mathbf{c}, \mathbf{d}, \mathbf{x} \in \mathbf{R}^n$, $\mathbf{b} \in \mathbf{R}^m$, $A \in \mathbf{R}^{m \times n}$, $\operatorname{rang}(A) = m$.

Legyen $P := \{\mathbf{x} \in \mathbf{R}^n : A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$ a (P) feladat *megengedett megoldásainak* a halmaza. Közismert, hogy a hiperbolikus programozási feladat duálisa a következő lineáris programozási feladat:

$$\left.\begin{array}{lcl} & \max & -y_0 \\ A^T\mathbf{y} + \mathbf{d}y_0 & \geq & -\mathbf{c} \\ \mathbf{b}^T\mathbf{y} & \leq & 0 \end{array}\right\} \quad \text{(D)}\,.$$

A dolgozatban egy új, véges pivot algoritmust adunk a hiperbolikus programozási feladat megoldására. Módszerünk a hiperbolikus feladatok esetén szokásos táblától eltérőt használ, ezért ezt az alábbiakban ismertetjük.



1. ábra

Az 1. ábrán használt jelölések értelmezése a következő:

$$c_0 := \mathbf{c}_B B^{-1}\mathbf{b}, \qquad d_0 := \mathbf{d}_B B^{-1}\mathbf{b},$$
$$\overline{\mathbf{b}} := B^{-1}\mathbf{b}, \quad \overline{\mathbf{c}} := \mathbf{c} - \mathbf{c}_B B^{-1}A, \quad \overline{\mathbf{d}} := \mathbf{d} - \mathbf{d}_B B^{-1}A,$$
$$\mathbf{q} := (1/d_0)\overline{\mathbf{b}}, \qquad \mathbf{p} := \overline{\mathbf{c}} - (c_0/d_0)\overline{\mathbf{d}},$$
$$\mathbf{u}^{(i)} := \mathbf{t}^{(i)} + (\overline{b}_i/d_0)\overline{\mathbf{d}}, \qquad \mathbf{v}_j := B^{-1}\mathbf{a}_j + (\overline{d}_j/d_0)B^{-1},$$
$$T := B^{-1}A,$$

ahol B jelöli az $A\mathbf{x} = \mathbf{b}$ lineáris egyenletrendszer egy bázisát. A B bázis vektorainak indexhalmazát jelölje J_B, továbbá $J_N = J \setminus J_B$ a bázison kívüli változók indexhalmaza.

1. Definíció *Az x_i változó az $i \in J_B$ index esetén* primál nem megengedett, *ha $\overline{b}_i < 0$, továbbá az x_j változó pedig a $j \in J_N$ index esetén* duál nem megengedett, *ha $\overline{c}_j < (c_0/d_0)\overline{d}_j$.*

Az alábbi optimalitási (1. Megjegyzés), primál nem megengedettségi (2. Megjegyzés) illetve duál nem megengedettségi (3. Megjegyzés) feltételek jól ismertek és a pivot tábla konstrukcióját figyelembe véve könnyen igazolhatók.

1. Megjegyzés *Ha $\overline{\mathbf{b}} \geq 0$ és $\mathbf{p} \geq 0$, akkor a pivot táblánk optimális.*

2. Megjegyzés *Ha van olyan $r \in J_B$ index, amely esetén $\overline{b}_r < 0$ és $\mathbf{u}^{(r)} \geq 0$ akkor nem létezik primál megengedett megoldás.*

3. Megjegyzés *Ha van olyan $r \in J_N$ index, amely esetén $p_r < 0$ és $\mathbf{v}_r \leq 0$ akkor nem létezik duál megengedett megoldás.*

A fenti 1-3. Megjegyzések a bevezetésre kerülő algoritmusunk megállási kritériumait fogalmazzák meg.

Terlaky [11] criss-cross módszerét, illetve a 3. Megjegyzést szem előtt tartva, egy lehetséges pivot szabály a következő lenne:

P. Pivot szabály

I. (i) Ha $\overline{\mathbf{b}} \geq \mathbf{0}$ és $\mathbf{p} \geq \mathbf{0}$, akkor optimális táblánál vagyunk.

(ii) Ha az (i) nem áll fenn, akkor legyen

$$k := \min\{i : \overline{b}_i < 0 \text{ vagy } p_i < 0, i = 1, 2, \ldots, n\}.$$

II. (i) Ha valamely k indexre $p_k < 0$ és $\mathbf{v}_k \leq \mathbf{0}$ akkor nem létezik duál megengedett megoldás.

(ii) Ha az(i) nem áll fenn, akkor legyen

$$s := \min\{j : v_{jk} > 0, j \in J_B\},$$

pivotáljunk az (s, k) pozición és menjünk az I.-re.

III. (i) Ha valamely k indexre $\overline{b}_k < 0$ és $\mathbf{u}^{(r)} \geq \mathbf{0}$ akkor nem létezik primál megengedett megoldás.

(ii) Ha az (i) nem áll fenn, akkor legyen

$$r := \{i : u_{ki} < 0, i \notin J_N\},$$

pivotáljunk a (k, r) pozición és menjünk a II.-ra.

A P. Pivot szabály mindenképpen finomításra szorul, hiszen a τ_{sk} illetve τ_{kr} helyen állhat nulla is. Ez egyértelműen annak a következménye, hogy a táblán kívüli mesterségesen kiszámított vektorok előjele alapján határozzuk meg a döntésünket. Mielőtt az 1. ábrán bemutatott táblán a P. Pivot szabály alkalmazása során felmerülő hátrányokat mutatnánk be, talán érdemes felhívni a figyelmet bizonyos lényeges előnyökre is.

1. Állítás *Legyen a $d'_0 \neq 0$ az aktuális B' bázis esetén. Válasszuk ki az (s, r) pivot pozíciót a $\overline{\mathbf{b}}, \mathbf{u}^{(s)}$ (illetve a $\mathbf{p}, \mathbf{v}_r$) vektorok segítségével. Tegyük fel, hogy a pivot pozíción lévő $\tau'_{sr} \neq 0$. Ekkor a pivot transzformáció után előálló $d''_0 \neq 0$.*

Bizonyítás. A P. Pivot szabály alapján $\overline{b_s} < 0$ és $u_{sr} = \tau_{sr} \cdot \overline{b}_s \frac{\overline{d}_r}{d'_0} < 0$. Tegyük fel, indirekt módon, hogy a $d''_0 = 0$. Ekkor a feltételünk alapján a

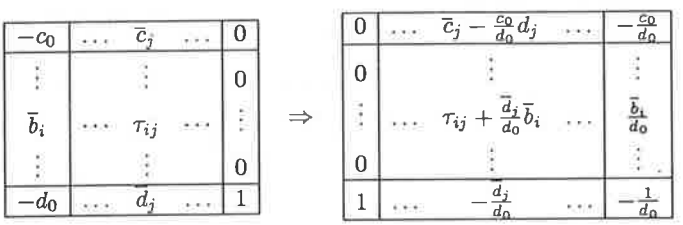
$$-d''_0 = -d'_0 - \overline{d}_r \frac{\overline{b}_s}{\tau_{sr}} = 0\,,$$

azaz $d'_0 = -\bar{d}_r \frac{\bar{b}_s}{\tau_{sr}}$ adódik. Felhasználva a $d'_0 \neq 0$ és $\tau_{sr} \neq 0$ feltételeket, kapjuk, hogy

$$\tau_{sr} + \bar{b}_s \frac{\bar{d}_r}{d'_0} = 0 \,,$$

ami ellentmond az $u_{sr} < 0$ feltételnek. ■

A $\mathbf{p}$, $\mathbf{v}_j$ és $\mathbf{u}^{(i)}$ vektorok használatából származó előnyök a 2. fejezetben definiálásra kerülő algoritmus végességének igazolásakor lesznek szembeötlők. Ha a P. Pivot szabályt szeretnénk megőrizni (bizonyos finomítások mellett), akkor a $\tau_{sk} = 0$ (illetve $\tau_{kr} = 0$) eseteket áthidaló megoldást — a pivot tábla struktúrájának megőrzése mellett — kell találnunk. Nyilvánvaló, hogy a szabály által kijelölt helyen nem pivotálhatunk. Célszerűnek látszik a $-c_0$ illetve $-d_0$ pozíciók valamelyikén egy *külső pivot*ot végrehajtani. A $\mathbf{p}$, $\mathbf{v}_j$ és $\mathbf{u}^{(i)}$ vektorok definíciója alpján nem igazán meglepő, hogy a $-d_0$ pozíció az igazán alkalmas pivot-pozíció. Ennek hatását a következő ábrán mutatjuk be.



2. ábra

A transzformáció feltétele $d_0 \neq 0$, de ez nyilván teljesül, hiszen egy olyan állapotnál voltunk, amikor a $\mathbf{p}$ illetve $\bar{\mathbf{b}}$ vektorok kiszámíthatók voltak. A $-d_0$ pozíción való pivotálás után nyert tábla a 3. ábrán látható.

0	$\mathbf{p}$	$-c_0/d_0$
0	$T = V = U$	$\mathbf{q}$
1	$-1/d_0\bar{\mathbf{d}}$	$-1/d_0$

3. ábra

Vegyük észre, hogy itt $\mathbf{p}$ és $\mathbf{q}$ a tábla részévé vált, míg a $\bar{\mathbf{b}}$ vektor már nem található meg a bázistáblán. Ha ezen a táblán újra döntünk a pivot pozíció

megválasztásáról, akkor ismét az (s,k) illetve (k,r) pozíciókat jelölné ki a P. Pivot szabály, de jelenleg ezeken a pozíciókon a

$$\tau_{sk} + \frac{\overline{d}_k}{d_0}\overline{b}_s = \frac{\overline{d}_k}{d_0}\overline{b}_s \quad \text{illetve a} \quad \tau_{kr} + \frac{\overline{d}_r}{d_0}\overline{b}_k = \frac{\overline{d}_r}{d_0}\overline{b}_k$$

értékek állnak. Akár $\mathbf{v}_k$ akár $\mathbf{u}^{(k)}$ definícióját nézzük meg, ezek az értékek nem lehetnek nullák. Tehát az előírt pivot végrehajtható. (Igaz, ezt megelőzően egy speciális pivotot is végre kellett hajtanunk.)

Mivel az eredetileg meghatározott pivotálást csak egy további segítségével lehet végrehajtani, ezért *kettős pivot*álásról beszélünk. A kettős pivotálás szembetűnő előnye: utána ugyanolyan vektorok alapján döntünk a pivot pozíció kiválasztásáról, mint előtte, azaz az előjelstruktúra szempontjából a kettős pivot *nem* zavarja meg az eljárást.

Ha a P. Pivot szabályt azzal pontosítjuk, hogy szükség esetén kettős pivotot hajtunk végre, akkor igaz a következő állítás:

1. Lemma *Kettős pivot legfeljebb egyszer fordulhat elő.* ■

Ha az aktuális bázisunk B' olyan, hogy a $g(\mathbf{x}) = \mathbf{d}^T\mathbf{x}$ függvénynek nem nulla szintvonalán vagyunk, akkor a P. Pivot szabállyal választott báziscsere esetén $g(\mathbf{x}'') \neq 0$ lesz, vagyis nem kerülhetünk a nulla szintvonalra, amely gondot okozhatna. A kettős pivot elvégezhetőségének a feltétele szintén az, hogy az aktuális bázismegoldásunk esetén a $d'_0 \neq 0$ teljesüljön. Így az alábbi feltételezéssel élünk.

1. Feltétel

$$\{\mathbf{x} \in \mathbf{R}^n : A\mathbf{x} = \mathbf{b}\} \cap \{\mathbf{x} \in \mathbf{R}^n : \mathbf{d}^T\mathbf{x} = 0\} = \emptyset.$$

Az 1. Feltétel lényegében az elfajuló esetet zárja ki, vagyis azt, amikor a megoldáshalmaz része a $g(\mathbf{x})$ nulla helyeinek a halmazának (amely tulajdonképpen egy hipersík). Az 1. Feltétel nem igazán erős megkötés.

3. Egy criss-cross típusú algoritmus

Ebben a fejezetben bemutatjuk a hiperbolikus programozási feladat megoldására megfogalmazott criss-cross típusú algoritmusunkat. Igazoljuk végességét és két egyszerű példán illusztráljuk működését is. Mielőtt rátérnénk az algoritmus részletezésére, rójuk ki a hiperbolikus programozási feladatra szokásos *pozitivitási megkötés*t, amelyre a későbbiek során szükségünk lesz:

2. Feltétel

$$P \subset \{\mathbf{x} \in \mathbf{R}^n : \mathbf{d}^T\mathbf{x} > 0\}\,.$$

1. Algoritmus

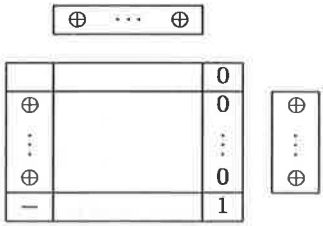
Legyen adott egy B_0 induló bázis, amely kielégíti az (1.) feltételt.

1. lépés: Számítsuk ki a $d_0 = \mathbf{d}_B^T B^{-1}\mathbf{b}$ és $c_0 = \mathbf{c}_B^T B^{-1}\mathbf{b}$ értékeket illetve a $\mathbf{p} := \overline{\mathbf{c}}_N - \frac{c_0}{d_0}\overline{\mathbf{d}}_N$ és $\mathbf{q} := 1/d_0\overline{\mathbf{b}}$ vektorokat.

Legyen $I := \{i \in J_B : \overline{b}_i < 0\} \cup \{i \notin J_B : p_i < 0\}$.

Ha $I = \emptyset$ **akkor** az alábbi esetek fordulhatnak elő:

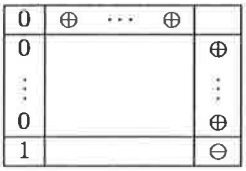
(1)



4. ábra

Optimális megoldás. STOP.

(2)



5. ábra

Optimális megoldás. STOP.

különben legyen $r := \min_{i \in I} i$ és menjünk a 2. lépésre.

2. lépés: Ha $r \in J_B$

akkor (*Duál iteráció*)

számoljuk ki a $\mathbf{u}^{(r)}$ vektort és legyen

$J := \{j \notin J_B : u_{rj} < 0\}$.

Ha $J = \emptyset$ **akkor** $P = \emptyset$, STOP

különben legyen $s := \min_{j \in J} j$.

Ha $\tau_{rs} = 0$ **akkor** hajtsunk végre egy kettős

pivotot a d_0 és az (r, s) pozíción,

különben pivotáljunk az (r, s) pozíción.

különben (*Primál iteráció*)

számoljuk ki a $\mathbf{v}_r$ vektort és legyen

$J := \{j \in J_B : v_{jr} > 0\}$.

Ha $J = \emptyset$ **akkor** $D = \emptyset$, STOP

különben legyen $s := \min_{j \in J} j$.

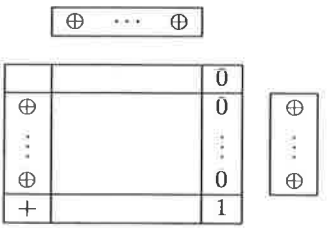
Ha $\tau_{sr} = 0$ **akkor** hajtsunk végre egy kettős pivotot

a d_0 és az (s, r) pozíción,

különben pivotáljunk az (s, r) pozíción.

Menjünk az 1. lépésre. □

Az algoritmus végességének a bizonyítása előtt vizsgáljuk meg a megállási kritériumait. A 4. ábrán lévő előjelstruktúra primál megengedett megoldást jelent és a $-d_0 < 0$ miatt a 2. Feltétel (is) teljesül. (A $d_0 = 0$ esetet a 2. Feltétel kizárja.)



6. ábra

A 6. ábrán lévő előjelstruktúra előfordulását kizárja a 2. Feltétel, mert a $-d_0 > 0$ nem lehetséges, ha primál megengedett megoldásunk van.

Az 5. ábrán látható előjelstruktúrával leírt megállási feltételt két részletben tárgyaljuk attól függően, hogy a $-1/d_0 < 0$ vagy pedig a $-1/d_0 = 0$. Kezdjük az egyszerűbb esettel, amikor is $-1/d_0 < 0$. Figyelembe véve a $\mathbf{q}$ vektor definícióját, kapjuk, hogy a $\overline{\mathbf{b}} \geq \mathbf{0}$, azaz primál megengedett megoldásunk van (és a 2. Feltétel is teljesül). A $\mathbf{p} \geq \mathbf{0}$ összefüggést (is) figyelembe véve, megoldásunk optimális.

A $-1/d_0 = 0$ esetet a következő lemma segítségével oldjuk meg.

2. Lemma *Legyen adott a következő előjelstruktúrájú tábla:*

0	$\oplus \ldots \oplus$	
0		$\oplus$
$\vdots$		$\vdots$
0		$\oplus$
1		0

Ekkor a (P) feladatnak létezik megengedett megoldása, a célfüggvény korlátos és az optimális célfüggvényérték csak határértékben állítható elő.

Bizonyítás. A fent látható előjelstruktúra azt jelenti, hogy a

$$\left.\begin{array}{rcrcr} & & & \min & \mathbf{c}^T\mathbf{u} \\ A\mathbf{u} & - & \tau\mathbf{b} & = & \mathbf{0} \\ \mathbf{d}^T\mathbf{u} & & & = & 1 \\ \mathbf{u} & & & \geq & \mathbf{0} \\ & & \tau & \geq & 0 \end{array}\right\} \quad \text{(SP)},$$

feladat egy optimális bázismegoldásánál vagyunk, ($\hat{\varphi} := \mathbf{c}^T\hat{\mathbf{u}}$ optimumértékkel) amely esetén $\hat{\tau} = 0$, vagyis előállítottuk az

$$\begin{array}{rcl} A\mathbf{u} & = & \mathbf{0} \\ \mathbf{d}^T\mathbf{u} & = & 1 \\ \mathbf{u} & \geq & \mathbf{0} \end{array}$$

feltételrendszer egy megengedett megoldását, az $\hat{\mathbf{u}} := \mathbf{q} \geq 0$ vektort. Ekkor poliéderünk nem korlátos és egy olyan irányhoz jutottunk, amely esetén az

$$\mathbf{x}_k := \mathbf{x}_0 + k\hat{\mathbf{u}} \in P$$

pontsorozat ($\mathbf{x}_0 \in P$ és $\mathbf{d}^T\mathbf{x}_0 = 1$) a következő tulajdonsággal rendelkezik:

$$\varphi_{0,k} := \frac{\mathbf{c}^T\mathbf{x}_k}{\mathbf{d}^T\mathbf{x}_k} = \frac{\mathbf{c}^T\mathbf{x}_0 + k\mathbf{c}^T\hat{\mathbf{u}}}{\mathbf{d}^T\mathbf{x}_0 + k\mathbf{d}^T\hat{\mathbf{u}}} = \frac{\mathbf{c}^T\mathbf{x}_0 + k\mathbf{c}^T\hat{\mathbf{u}}}{\mathbf{d}^T\mathbf{x}_0 + k} = \frac{\mathbf{c}^T\mathbf{x}_0 + k\hat{\varphi}}{k+1}.$$

Könnyen belátható, hogy a $\varphi_{0,k}$ monoton nem növekvő sorozat. Ha $\varepsilon > 0$ pontossággal akarjuk előállítani a feladat megoldását, akkor

$$k(\mathbf{x}_0, \varepsilon) := \frac{c_0 - d_0\hat{\varphi}}{\varepsilon}$$

küszöbszám adódik. ■

Az 1. lépés (2) esetéhez (5. ábra) kapcsolódik még az alábbi előjelstruktúra is (7. ábra), amelynek az előfordulását a 2. Feltétel kizárja.

0	$\oplus \cdots \oplus$	
0		$\ominus$
$\vdots$		$\vdots$
0		$\ominus$
1		$+$

7. ábra

Az 1. Algoritmus végességének bizonyítása során mindvégig a 3. ábrán látható táblát használjuk, hiszen a pivot szabályt is ehhez kapcsolódóan definiáltuk, azzal a kiegészítéssel, hogy a **q** vektor helyett mindenkor a **b** vektorral dolgozunk. Az indoklásunk az un. *ortogonalitási tételre* épül ([6], 2.5. Tétel). Mivel a kettős pivot nem változtatja meg a belépő és távozó vektor kiválasztásának a szabályát, ezért a végesség bizonyításánál nem játszik szerepet (1. Állítás és 1.6 Lemma). A bizonyítás menete követi Terlaky [11] bizonyítását.

1. Tétel *A hiperbolikus criss-cross módszer (1. Algoritmus) véges.*

Bizonyítás. Indirekt bizonyítást alkalmazunk, azaz tegyük fel, hogy eljárásunk nem véges. Mivel véges sok bázisunk lehet, ezért ez azt jelenti, hogy valamelyik végtelen sokszor előfordul. Ez csak úgy lehetséges, ha az eljárás ciklizál. Jelölje az I^* halmaz a ciklus során belépő változók indexeinek a halmazát (ezek nyilván pontosan azok, amelyek egyben a ciklus során távoznak is). Legyen $l := \max_{i \in I^*} i$. Figyeljük az l. változó mozgását. Ekkor négy eset lehetséges:

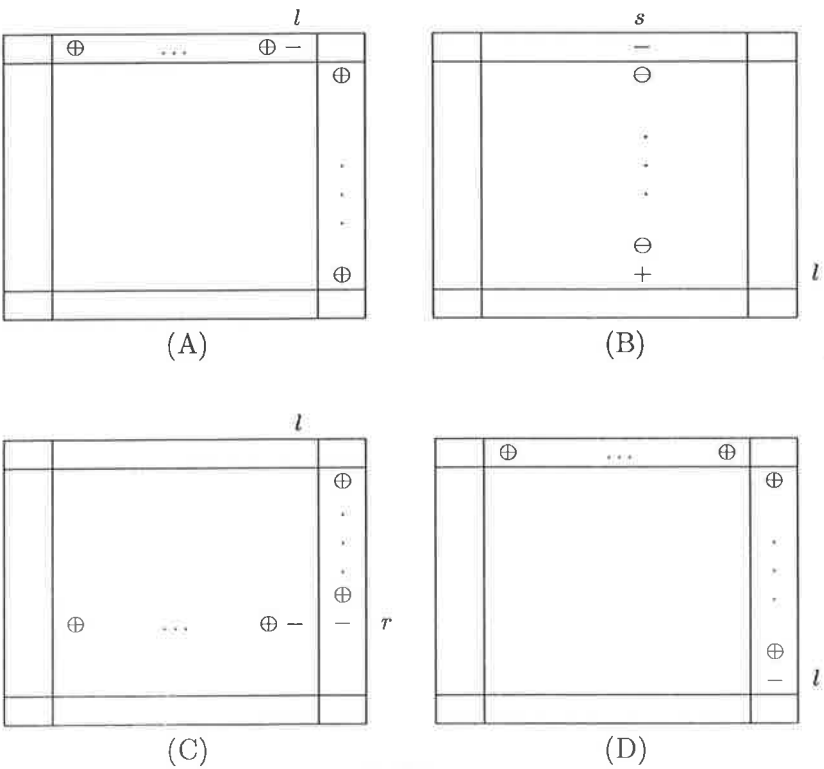
(a) az l. változó primál iterációnál lép be a bázisba és primál iterációnál távozik a bázisból;

(b) az l. változó primál iterációnál lép be a bázisba és duál iterációnál távozik a bázisból;

(c) az l. változó duál iterációnál lép be a bázisba és primál iterációnál távozik a bázisból;

(d) az l. változó duál iterációnál lép be a bázisba és duál iterációnál távozik a bázisból.

A 8. ábra mutatja a négy pivot tábla előjelstruktúráját amikor az l. változó belép illetve távozik a bázisból (felül a **p** vektor sora található, míg jobbra az utolsó oszlopban a **q** vektor helyén (3. ábra), a **b** vektoré). Az (A) tábla azt mutatja, amikor az l. változó primál iterációnál belép, a (B) tábla azt a helyzetet, amikor az l. változó primál iterációnál távozik a bázisból. A (C) tábla azt mutatja, amikor az l. változó duál iterációnál belép, és végül a (D) tábla azt a helyzetet, amikor az l. változó duál iterációnál távozik a bázisból.



8. ábra

Vizsgáljuk meg sorra a lehetséges eseteket. A könnyebb érthetőség kedvéért

a megfelelő sor- illetve oszlopvektorok előjelstruktúráját is megadjuk. (Ezek definíciója megtalálható Klafszky és Terlaky cikkében ([6], 433. oldal). Természetesen nem szabad elfelejteni, hogy bizonyításaink – lévén hiperbolikus programozásról szó és nem lineáris programozásról – az $(A, -\mathbf{b})$ mátrix és $(\mathbf{c}, 0)$ vektor által kifeszített altér helyett a $\mathrm{Lin}\big((A, -\mathbf{b}, \mathbf{0}); (\mathbf{c}, 0, 0); (\mathbf{d}, 0, 1)\big)$ altérben történnek.)

Mivel az (a), (c) és (d) esetek bizonyítása azonos technikát igényel, míg a (b) eseté ezektől kissé eltérőt, ezért először az (a), (c) és (d) esetek bizonyításával foglalkozunk.

(a) Az l. változó primál iterációnál lép be a bázisba illetve az l. változó primál iterációnál távozik a bázisból és a belépő változó az s. Ekkor a 8. ábra (A) és (B) tábláját kell figyelembe vennünk. Az ortogonalitási tételt használva azt kell kapnunk, hogy az (A) pivot tábla $\mathbf{p}$ sora merőleges a (B) tábla s. oszlopára.

Az ortogonalitási tételt a $\mathbf{p}$ vektor sorára és az s. változó oszlopára kell alkalmaznunk. A $\mathbf{p}$ vektor sorának előállításakor a $\mathbf{p}$ vektor definícióját vesszük figyelembe, azaz $\mathbf{t}^{(\mathbf{p})} = \mathbf{t}^{(\mathbf{c})} - \frac{c_0}{d_0}\,\mathbf{t}^{(\mathbf{d})}$. (Ebből azonnal adódik, hogy a $\mathbf{c}$ és $\mathbf{d}$ oszlopában 1 illetve $-\frac{c_0}{d_0}$ áll.)

	l	$\mathbf{b}$	$\mathbf{c}$	$\mathbf{d}$
$\mathbf{t}^{(\mathbf{p})} =$	$\oplus \ \dots \ \oplus$	$-$	0	1	$-\frac{c_0}{d_0}$

	l	$\mathbf{b}$	$\mathbf{c}$	$\mathbf{d}$
$\mathbf{t}_s =$	$\ominus \ \dots \ \ominus$	$+$	0	$\bar{c}_s$	$\bar{d}_s$

Ekkor a $\mathbf{t}_s{}^T\mathbf{t}^{(\mathbf{p})} < 0$ adódik, figyelembe véve a p_s előjelét is, hiszen $p_s = c_s - \frac{c_0}{d_0}\,d_s < 0$, ami újra ellentmondást eredményez.

(c) Az l. változó duál iterációnál lép be a bázisba, a távozó változó az r. illetve az l. változó primál iterációnál távozik a bázisból. Ekkor a 8. ábra (C) és (B) tábláját kell figyelembe vennünk. Az ortogonalitási tételt használva azt kell kapnunk, hogy a (C) (teljes) pivot tábla r. sora merőleges a (B) tábla s. oszlopára. Tehát a $\mathbf{t}^{(r)}$ és a $\mathbf{t}_s$ vektorok merőlegesek.

	l	$\mathbf{b}$	$\mathbf{c}$	$\mathbf{d}$
$\mathbf{t}^{(r)} =$	$\oplus \ \dots \ \oplus$	$-$	$-$	0	0

	l	$\mathbf{b}$	$\mathbf{c}$	$\mathbf{d}$
$\mathbf{t}_s =$	$\ominus \ \dots \ \ominus$	$+$	0	$*$	$*$

Ekkor nyilvánvalóan $\mathbf{t}_s{}^T\mathbf{t}^{(r)} < 0$, ami ellentmond az ortogonalitási tételnek.

(d) Az l. változó duál iterációnál lép be a bázisba, a távozó változó az r. illetve az l. változó duál iterációnál távozik a bázisból. Ekkor a 8.

ábra (C) és (D) tábláját kell figyelembe vennünk. Az ortogonalitási tételt használva azt kell kapnunk, hogy a (C) (teljes) pivot tábla r. sora merőleges a (D) tábla **b** oszlopára.

$$\begin{array}{c|c|c|c|c|c|c} & & l & \mathbf{b} & \mathbf{c} & \mathbf{d} \\ \mathbf{t}^{(r)} = & \oplus \ \ldots \ \oplus & - & - & 0 & 0 \end{array}$$

$$\begin{array}{c|c|c|c|c|c|c} & & & & & & \\ \mathbf{t_b} = & \oplus \ \ldots \ \oplus & - & -1 & * & * \end{array}$$

A $\mathbf{t}^{(r)}$ **c**-vel és **d**-vel jelölt pozícióin azért áll nulla, mert a **c**-nek és **d**-nek sorvektorok felelnek meg („bázisban" vannak), míg a $\mathbf{t}_b$ vektornál a $*$ mindkét esetben azt jelenti, hogy akármilyen érték állhat ott. Ekkor a $\mathbf{t_b}^T\mathbf{t}^{(r)} > 0$ adódik, ami ellentmond az ortogonalitási tételnek.

Végül térjünk rá a (b) eset tárgyalására.

(b) Az l. változó primál iterációnál lép be a bázisba illetve az l. változó duál iterációnál távozik a bázisból. Ekkor a 8. ábra (A) és (D) tábláját kell figyelembe vennünk.

Az (A) táblából nyert vektorokat $'$-vel, míg a (D) táblából nyerteket $''$-vel jelöljük, és így nem okoz gondot, hogy mindkét táblából előállítjuk a **p**-nek megfelelő sor-, illetve a **b**-nek megfelelő oszlopvektorokat. A **b**-nek megfelő oszlopvektorokat normáljuk úgy, hogy a **d**-hez tartozó koordináta -1 legyen. Ekkor az alábbi vektorokat nyerjük:

$$\begin{array}{c|c|c|c|c|c|c} & & l & \mathbf{b} & \mathbf{c} & \mathbf{d} \\ \mathbf{t}^{(\mathbf{p})'} = & \oplus \ \ldots \ \oplus & - & 0 & 1 & -\frac{c'_0}{d'_0} \end{array}$$

$$\begin{array}{c|c|c|c|c|c|c} & & & & & & \\ \mathbf{t}''_{\mathbf{b}} = & \oplus \ \ldots \ \oplus & - & \frac{-1}{d''_0} & \frac{-c''_0}{d''_0} & -1 \end{array}$$

illetve a

$$\begin{array}{c|c|c|c|c|c|c} & & l & \mathbf{b} & \mathbf{c} & \mathbf{d} \\ \mathbf{t}^{(\mathbf{p})''} = & \oplus \ \ldots \ \oplus & 0 & 0 & 1 & -\frac{c''_0}{d''_0} \end{array}$$

$$\begin{array}{c|c|c|c|c|c|c} & & & & & & \\ \mathbf{t}'_{\mathbf{b}} = & \oplus \ \ldots \ \oplus & - & \frac{-1}{d'_0} & \frac{-c'_0}{d'_0} & -1 \end{array}$$

Ekkor a merőlegességi tulajdonság miatt $\mathbf{t}''^{T}_{\mathbf{b}}\mathbf{t}^{(\mathbf{p})'} = 0$ és $\mathbf{t}'^{T}_{\mathbf{b}}\mathbf{t}^{(\mathbf{p})''} = 0$ adódik, azaz $0 = \mathbf{t}''^{T}_{\mathbf{b}}\mathbf{t}^{(\mathbf{p})'} + \mathbf{t}'^{T}_{\mathbf{b}}\mathbf{t}^{(\mathbf{p})''}$. Azonban a fenti előjelstruktúrát figyelembe véve a két szorzat összege pozitívnak adódik, ami ellentmodás. A tétel bizonyítását befejeztük. ■

Végezetül két példával illusztráljuk az algoritmus működését. Példáinkat Martos [9] könyvéből vettük és így azonnal alkalmunk nyílik az algoritmusunkat összehasonlítani egyéb jól ismert módszerekkel. Igazán nem meglepő, hogy algoritmusunk más bázisokon keresztül közelíti meg az optimális megoldást, hiszen rendelkezik az összes criss-cross algoritmus azon tulajdonságával, hogy sem nem primál, sem nem duál megengedett bázisokat is meglátogat, illetve a célfüggvény változása nem monoton.

Tekintsük az első példát Martos ([9], 170. oldal) könyvéből, amellyel saját módszerét illusztrálja.

1. Példa *Keressük meg a φ függvény minimumát.*

$$\begin{array}{rcl} \varphi(x_1,x_2) & = & \dfrac{24x_1+6}{5x_1+x_2+1} \\ -x_1+x_2 & \leq & 1 \\ x_1-x_2 & \leq & 1 \\ x_1,x_2 & \geq & 0 \end{array}$$

A feladat megengedett megoldás halmazának a csúcsai a következők: $\hat{\mathbf{x}}_1 = (1,0)$, $\hat{\mathbf{x}}_2 = (0,0)$, $\hat{\mathbf{x}}_3 = (0,1)$. A függvényértékek pedig: $\hat{\varphi}_1 = 6$, $\hat{\varphi}_2 = 6$, $\hat{\varphi}_3 = 3$. Martos módszerével az $\hat{\mathbf{x}}_1$ csúcsból az $\hat{\mathbf{x}}_3$ csúcsba nem lehet közvetlenül eljutni, mert nincsen közbülső, primál megengedett megoldás, ezért kénytelenek vagyunk az ún. *regularizációt* [9] végrehajtani, ami nem jelent egyebet, mint egy speciális új feltétel hozzávételét a feladathoz. Esetünkben ez az

$$x_1 + x_2 \leq 2$$

feltétel és így további két primál megengedett megoldást nyerünk $\hat{\mathbf{x}}_4 = (1/2, 3/2)$, $\hat{\mathbf{x}}_5 = (3/2, 1/2)$. Az $\hat{\mathbf{x}}_1$ csúcsból kiindulva az $\hat{\mathbf{x}}_5, \hat{\mathbf{x}}_4$ csúcsokon keresztül jutunk el az optimális megoldáshoz, az $\hat{\mathbf{x}}_3$ csúcsba, azaz *három* báziscserére és regularizációra volt szükégünk a megoldás során.

Ezzel szemben, ha a criss-cross módszert alkalmazzuk a feladatra és az $\hat{\mathbf{x}}_1$ csúcsból indulunk el, akkor az algoritmusunk szerint azonnal egy kettős pivotot kell csinálnunk, amelynek a végén az $\hat{\mathbf{x}}_2$ csúcsra érkezünk. A következő lépésben az optimális $\hat{\mathbf{x}}_3$ csúcsra lépünk. A megoldás során két lépést tettünk és a kettős pivot miatt három báziscserét hajtottunk végre, de más útvonalon jutottunk az optimális megoldáshoz.

Térjünk rá a második példára, amely Martos Béla könyvének [9] a 177. oldalán található meg és a hiperbolikus szimplex módszer illusztrálására szolgál.

2. Példa *Keressük meg a φ függvény minimumát.*

$$\begin{array}{rcl} \varphi(x_1,x_2) & = & \dfrac{-6x_1-5x_2}{2x_1+7} \\ x_1+2x_2 & \leq & 3 \\ 3x_1+2x_2 & \leq & 6 \\ x_1,x_2 & \geq & 0 \end{array}$$

Az $x_1 = x_2 = 0$ megengedett megoldása a feladatnak és ekkor az induló bázisváltozók az y_1 és y_2 (mesterséges változók). A hiperbolikus szimplex módszer segítségével a következő bázisok nyerhetők: $\{y_1,y_2\}$, $\{x_2,y_2\}$ és $\{x_2,x_1\}$. A harmadik bázis optimális megoldást szolgáltat. Két iterációra volt szükség és természetesen (lévén szimplex típusú módszerről szó) a célfüggvény monoton csökkent és a közbülső bázisok primál megengedettek voltak.

A hiperbolikus criss-cross módszert alkalmazva a feladatra a következő bázisokhoz jutunk: $\{y_1,y_2\}$, $\{x_1,y_2\}$ és $\{x_1,x_2\}$. Esetünkben a második bázis duál megengedett és primál nem megengedett. A célfüggvény változása nem monoton, vagyis a megoldásunk magán viseli a criss-cross módszerek jellemvonásait és szintén két iterációban oldottuk meg a feladatot, mint a hiperbolikus szimplex módszerrel.

Köszönetnyilvánítás. A cikkünk anyagának a jelentős része Illés Tibor 1994-es delfti látogatása során fogalmazódott meg (Peregrinatio II. Ösztöndíjjal töltött három hónapot Hollandiában). A dolgozat írott változatának elkészítését jelentősen segítette az első szerző koppenhágai tanulmányuta (Állami Eötvös Ösztöndíj), amely során a szerzők az Interneten keresztül cserélték ki a kézirat újabb és újabb változatait. Az eredmények hazai és nemzetközi konferenciákon való bemutatását az OTKA T 14302-es pályázata támogatta. Ezúton (is) kifejezzük köszönetünket kutatásunk támogatóinak.

Irodalom

1. Anstreicher, K. M., Analysis of Karmarkar's algorithm for fractional linear programming, *Technical Report*, (1985), November, Yale School of Management, Yale University, New Haven, CT 06520, USA.
2. Anstreicher, K. M., A monotonic projective algorithm for fractional linear programming, *Algorithmica*, 1, No. 4, (1986) 483-498.
3. Bitran, G. R., Novaes, A. G., Linear programming with a fractional objective function, *Operations Research*, 21 (1973) 22-29.
4. Charnes, A., Cooper, W. W., Programming with linear fractionals, *Naval Research Quaterly*, 9 (1962) 181-186.

5. Gilmore, P. C., Gomory, R. E., A linear programming approach to the cutting stock problem, part II., *Operations Research*, (1963) 863-888.
6. Klafszky E., Terlaky T., A pivot technika szerepe a lineáris algebra néhány alapvető tételének a bizonyításában, *Alkalmazott Matematikai Lapok*, 14 (1989) 425-448.
7. Martos B., Hiperbolikus programozás, *Az MTA Matematikai Kutató Intézetének Közleményei*, 5 Budapest (1960) 383-406.
8. Martos B., Nem-lineáris programozási módszerek hatóköre, *Az MTA Közgazdaságtudományi Intézetének Közleményei*, 20 Budapest, 1966.
9. Martos B., *Nonlinear Programming: Theory and Methods*, Akadémiai Kiadó, Budapest, 1975.
10. Schaible, S., Fractional Programming, in Eds. Pardalos, P. and Horst, R. *Handbook of Global Optimization*, Kluwer Academic Publishers, 1995.
11. Terlaky T., Egy új, véges criss-cross módszer lineáris programozási feladatok megoldására, *Alkalmazott Matematikai Lapok*, 10 (1984) 289-296.

THE FINITE CRISS-CROSS METHOD FOR HYPERBOLIC PROGRAMMING

In this paper the finite criss-cross method is generalized to solve hyperbolic programming problems. Like in the case of linear or quadratic programming the criss-cross method can be initialized with any, not necessarily feasible basis solution. The finiteness of the procedure is proved under the usual mild assumptions. Finally, some small numerical examples illustrate the main features of the algorithm.