FIATAL MŰSZAKIAK TUDOMÁNYOS ÜLÉSSZAKA XVII.

Kolozsvár, 2012. március 22–23.

PRIMÁL-DUÁL ALGORITMUSOK OBJEKTUMORIENTÁLT MEGVALÓSÍTÁSA MATLABBAN

DARVAY Zsolt, PATAKI Előd

Abstract

In this paper we provide solutions for linear optimization problems, which are used in engineering applications. We use different versions of primal-dual interior-point algorithms. These algorithms generally solve the optimization problem in polynomial time. We exploit the possibilities of the object oriented MATLAB, and thus we can compare easily the solving strategies. We have accomplished the design of the program in such a way to become reusable, flexible and easily extensible. Therefore, we have used the template method, strategy and memento behavioral design patterns.

Key words:

linear optimization, interior-point method, primal-dual, path-following, behavioral design pattern.

Mathematics Subject Classification (2010): 90C05, 90C51, 68N19.

Összefoglalás

Ebben a cikkben olyan lineáris optimalizálási feladatokra szolgáltatunk megoldást, amelyek műszaki jellegű problémákból származnak. A primál-duál belsőpontos algoritmusok különböző változatait használjuk, melyek általában polinom időben határozzák meg a feladat egy adott pontosságú megoldását. Az objektumorientált MATLAB lehetőségeit kihasználva könnyen összehasonlíthatjuk az egyes megoldási stratégiákat. Úgy valósítottuk meg a tervezést, hogy a program rugalmassá, újrafelhasználhatóvá és könnyen bővíthetővé váljon. Ezért a sablonfüggvény, stratégia és emlékeztető viselkedési tervezési minták kerültek felhasználásra.

Kulcsszavak:

lineáris optimalizálás, belsőpontos módszer, primál-duál, trajektóriakövető, viselkedési tervezési minta.

1. Bevezetés

Műszaki jellegű problémák gyakran optimalizálási feladatokhoz vezetnek. Egy függvény szélsőértékeit keressük valamely megszorító feltételek mellett. A célfüggvény és a feltételek jellegétől függően különböző típusú feladatokat határozunk meg. Ebben a cikkben a lineáris programozás belsőpontos algoritmusaival foglalkozunk, amelyekről általában bizonyítható, hogy egy adott pontosságú közelítést polinom időben szolgáltatnak. Az első belsőpontos algoritmust Karmarkar [4] vezette be, és egy potenciálfüggvény segítségével igazolta a módszer polinomiális jellegét. Ezt követően a belsőpontos algoritmusoknak nagyon sok változata jelent meg [1,3,5-7]. A trajektóriakövető algoritmusok egy sajátos útvonalat, a centrális utat használják annak érdekében,

hogy az optimális megoldáshoz közelebb kerüljünk. A továbbiakban egy primál-duál trajektóriakövető algoritmus megvalósításával foglalkozunk különböző tervezési mintákra alapozva.

2. A lineáris programozási feladat

A lineáris programozási feladatok különböző, egymással ekvivalens formában adhatóak meg. A lineáris programozás szakirodalmában a primál (P) és duál (D) feladatokat leggyakrabban a következő standard alakban határozzuk meg:

$$(P) \begin{cases} \min c^T x, \\ Ax = b, \\ x \geq 0, \end{cases} \quad (D) \begin{cases} \max b^T y, \\ A^T y + s = c, \\ s \geq 0, \end{cases}$$

ahol A egy olyan mátrix, amely m sorral és n oszloppal rendelkezik, $\text{rang}(A) = m$ és b illetve c m illetve n dimenziós oszlopvektorok. Továbbá 0 is egy vektor, melynek minden eleme 0 , tehát $0^T = [0, \dots, 0]$. A primál feladatban a *célfüggvény* ($c^T x$) minimumát, míg a duál feladatban a *célfüggvény* ($b^T y$) maximumát keressük. Mindkét felírás esetén a további feltételeket *megszorításoknak* nevezzük. Ha a primál-duál feladatpárnak van megoldása, akkor a *célfüggvényértékek* megegyeznek és a kapott vektorokra $x^T s = 0$. Ha x és s két olyan vektor, amely nem szolgáltat optimális megoldást, akkor az $x^T s$ mennyiség határozza meg a *célfüggvényértékek* közti különbséget, melyet *dualitási résnek* nevezünk.

Az alkalmazott primál-duál algoritmus egy belső pontból indul, melyről feltételezzük, hogy a centrális út közelében helyezkedik el. Az elmozdulásvektorokat a Newton módszer segítségével határozzuk meg, egy lineáris egyenletrendszer megoldva. A lépés nagyságát úgy számoljuk ki, hogy a kapott pont a megszorító feltételek által meghatározott poliéder belsejében maradjon. Mindezt addig ismétljük, amíg a dualitási rés egy adott pontossági paraméternél kisebbé válik.

A gyakorlatból származó feladatok általában nincsenek standard alakban, tehát a megszorítások esetén nincs mindenütt egyenlőség. A lineáris programozási feladatot általánosan így írhatjuk:

$$\begin{cases} \min c^T x, \\ \sum_{j=1}^n a_{ij} x_j \geq b_i, i \in I_1, & (i) \\ \sum_{j=1}^n a_{ij} x_j \leq b_i, i \in I_2, & (ii) \\ \sum_{j=1}^n a_{ij} x_j = b_i, i \in I_3, & (iii) \\ x_j \leq \alpha, j \in I_1', & (iv) \\ x_j \geq \alpha, j \in I_2', & (v) \\ x_j \in \mathbb{R}, j \in I_3'. & (vi) \end{cases}$$

Itt I_1, I_2, I_3 és I_1', I_2', I_3' nem mind üres, diszjunkt indexhalmazok.

A továbbiakban azt vizsgáljuk meg, hogy milyen átalakításra van szükség annak érdekében, hogy a feladatot standard alakra hozzuk.

Amint láthatjuk, a (iii) egyenleten nem kell átalakításokat végezni. Az (i) és (ii) esetén úgy kellene alakítani az összefüggéseket, hogy egyenlőséget kapjunk. Konkrétan, (i) esetén a megfelelő egyenlőtlenségekből kivonjuk a $z_i \geq 0, i \in I_1$ változókat. Ekkor a következőt kapjuk:

$$\sum_{j=1}^n a_{ij}x_j - z_i = b_i, i \in I_1. \quad (1)$$

Hasonlóan, a (ii) esetén a $\sum_{j=1}^n a_{ij}x_j + z_i = b_i, i \in I_2$ összefüggéshez jutunk, tehát az egyes egyenlőtlenségekhez hozzáadjuk a $z_i \geq 0, i \in I_2$ változókat.

A felső illetve alsó korlátokat meghatározó feltételeket is szabványos alakra kell hozni. Az (v) egyenlőtlenség esetén az ismeretlen változónak van egy alsó korlátja, mely α -val egyenlő. Ezért a változót kicseréljük egy másikkal, mely egyenlő a régi változó és alsó korlátjának különbségével:

$$x_j' = x_j - \alpha, j \in I_2'. \quad (2)$$

Hasonlóan járunk el a (iv) egyenlőtlenséggel is:

$$x_j' = -x_j + \alpha, j \in I_1'. \quad (3)$$

Ha szabad változónk van, mint a (vi) esetén, akkor két új változó bevezetésére van szükség:

$$x_j = x_j' - x_j'', x_j' \geq 0, x_j'' \geq 0, j \in I_3'. \quad (4)$$

3. A tervezési minták használata

A program a sablonfüggvény, a stratégia, és az emlékeztető tervezési mintákat használja. A *sablonfüggvény* tervezési minta egy algoritmus vázát határozza meg, de lehetőséget teremt arra, hogy egyes lépéseket a származtatott osztályokban adjunk meg. A *stratégia* tervezési minta egy algoritmus-családot határoz meg úgy, hogy minden egyes algoritmust külön modulba zár, megadva a felcserélhetőség lehetőségét. Az *emlékeztető* tervezési minta egy objektum belső állapotáról készít másolatot a zártság megsértése nélkül [2].

A sablonfüggvény mintát alkalmazva az algoritmus lényeges lépései változatlanok maradnak, de az elmozdulásvektorokat meghatározó lineáris egyenletrendszerben a jobb oldalon szereplő vektor változhat. Ennek megfelelően az egyes algoritmusváltozatok könnyen felcserélhetővé válnak. A csere a stratégia minta segítségével valósítható meg.

Nézzük, miért volt szükséges az emlékeztető minta használata. A bemeneti állományokból származó feladat általában nincs standard alakban, az algoritmus alkalmazásához viszont erre az alakra van

szükségünk, tehát át kell alakítanunk a feladatot. Az algoritmus lefutása után megkapjuk a standard alakra hozott feladat megoldását. Nekünk viszont az eredeti megoldásra van szükségünk, tehát *vissza kell állítani az előző állapotot*. Ezért került beépítésre az emlékeztető minta.

4. Eredmények

Az alábbiakban a Netlib könyvtár néhány MPS állományára kapott eredményeket mutatjuk be.

1. táblázat. MPS állományokra kapott eredmények

Állomány neve	Állomány mérete (sorokban)	Iterációszám	Célfüggvény minimuma
afiro.mps	84	13	-464.7531
adlittle.mps	575	21	225494
bandm.mps	3089	28	-158.628
beaconfd.mps	3723	19	33592.4858
blend.mps	360	16	-30.8121
brandy.mps	2431	28	1518.5098

5. Következtetések

Ebben a cikkben a lineáris optimalizálási feladatok megvalósításának egy olyan új típusú tervezését mutattuk be, amely a viselkedési minták használatára alapszik. Ez által a kapott program jobban igazodik a változtatásokhoz, mint egy hagyományos alkalmazás.

Köszönetnyilvánítás

A szerzők köszönetüket fejezik ki az Erdélyi Múzeum-Egyesületnek a kutatási munkához nyújtott támogatásért (554.1.1 / P.2 (B.G.A. - 512/6/2011)).

Irodalom

- [1] Darvay, Zs.: *Belsőpontos módszerek a lineáris programozásban*, ELTE, 1997.
- [2] Gamma, E., Helm, R., Johnson, R., Vlissides, J.: *Programtervezési minták. Újrahasznosítható elemek objektumközpontú programokhoz*, Kiskapu Kiadó, 2004.
- [3] Illés, T., Nagy, M., Terlaky, T.: Belsőpontos algoritmusok. In: Iványi Antal (szerk.) *Informatikai algoritmusok II*, ELTE Eötvös Kiadó, 2005, 1230-1297.
- [4] Karmarkar, N. K.: A new polynomial-time algorithm for linear programming, *Combinatorica*, 1984, 4:373-395.
- [5] Roos, C., Terlaky, T., Vial, J.-P.: *Interior Point Methods for Linear Optimization*, Springer, 2006.
- [6] Wright, S.J.: *Primal-Dual Interior-Point Methods*, SIAM, 1997.
- [7] Ye, Y.: *Interior Point Algorithms, Theory and Analysis*, John Wiley & Sons, 1997.

dr. Darvay Zsolt, egyetemi adjunktus
Babeş-Bolyai Tudományegyetem
Matematika és Informatika Kar
Kolozsvár
E-mail: darvay@cs.ubbcluj.ro

Pataki Előd, egyetemi hallgató
Babeş-Bolyai Tudományegyetem
Matematika és Informatika Kar
Kolozsvár
E-mail: mr.pataki@yahoo.com