有理数演算による小数法の実現と数値実験

4B-5

大柳俊夫　（北海道大学）
加地郁夫　（北海道大学）

【１】はじめに

一般に，線形計画問題（ＬＰ問題）の変数すべてに整数条件を加えたものを全整数計画問題（ＩＬＰ問題）といい，この問題の解法の一つにR.Gomoryによる小数法がある[1]．この方法は，まずＩＬＰ問題の変数の整数条件を除いたＬＰ緩和問題を解き，つぎにその解をもとにすべての変数が整数になるまでカットと呼ばれる新しい制約を加えながら繰返しＬＰ問題を解くもので，

(1) 最適解を得るまでは実行可能解が１つも求まらない

(2) 一般に，最適解を得るまでに多くのカットを必要とし，浮動小数点演算では丸め誤差が蓄積され，いくらカットを加えても最適解が得られないことや，間違った解が得られることがしばしば起る

という特徴がある．そのため，この方法の汎用のアルゴリズムとしての実用性に関しては否定的な意見が大多数で，特に(2) の特徴のため，現在までこの方法に関する厳密かつ詳細な数値実験の報告はほとんど見当らない．

本稿は，小数法を有理数演算を使って実現し，この方法の有効性を厳密に調べるために行った数値実験について述べるものである．今回の実験は，種々の問題に対してカットの生成方法や除去方法を変え，最適解を得るまでに加えたカット数を調べたものである．

【２】小数法の概要

図１の形式の列タブローを用いた小数法の基本的なアルゴリズムは次のとおりである．

ｽﾃｯﾌﾟ1：ＬＰ緩和問題を解きｽﾃｯﾌﾟ2へ．

ｽﾃｯﾌﾟ2：列タブローの第０列成分がすべて整数ならば終了．そうでないときはｽﾃｯﾌﾟ3へ．

ｽﾃｯﾌﾟ3：第０列成分が非整数値をとる行（カット生成行と呼ぶ）をもとにカットを生成し，これをタブローの最下行に追加した新しいＬＰ問題を解く．そして不要になったカットを必要に応じて除去しｽﾃｯﾌﾟ2へ．

$$
\begin{array}{cc|cccc|}
 & & 1 & -x_1 & \cdots & -x_n \\
\hline
x_0 & = & a_{00} & a_{01} & \cdots & a_{0n} \\
x_1 & = & 0 & -1 & & \\
 & & 0 & & \cdot & \\
\vdots & & \vdots & & & \cdot \\
x_n & = & 0 & & & -1 \\
\hline
x_{n+1} & = & a_{n+1,0} & a_{n+1,1} & \cdots & a_{n+1,n} \\
\vdots & & \vdots & & & \\
x_{n+m} & = & a_{n+m,1} & a_{n+m,1} & \cdots & a_{n+m,n} \\
\hline
\end{array}
$$

図１　列タブロー

ただし，このアルゴリズムはＬＰ問題が実行不可能なら直ちに終了する．

このアルゴリズムの効率は，カット生成行の選択方法，カット生成方法およびカットの除去方法により大きく影響を受ける．

【３】カットの生成と除去

今回の数値実験で用いたカット生成行の選択方法，カット生成方法およびカットの除去方法は以下のとおりである．なお，

J　：非基底変数の添字集合，J'：J∪{0}

$f_{ij} = a_{ij} - [a_{ij}]$　　[ ] はガウス記号，
　　($0 \leqq f_{ij} < 1$, $j \in J'$)

frac(x)：xの小数部分を求める関数．
　　($0 \leqq \mathrm{frac}(x) < 1$)

とする．

(1) カット生成行（ｓ）の選択方法

カット生成行の選択方法としてとして以下の３つを考える．

Ⅰ：$s = \min_i \{ i \mid f_{i0} > 0 \}$ となる行ｓ．

Ⅱ：$f_{s0} = \max_i \{ f_{i0} \mid f_{i0} > 0 \}$ に対応する行ｓ．

Ⅲ：$f_{s0} = \min_i \{ f_{i0} \mid f_{i0} > 0 \}$ に対応する行ｓ．

(2) カットの生成方法

カット生成行ｓは，

$$x_s = a_{s0} + \Sigma a_{sj}(-x_j) \qquad (j \in J)$$

を表わしている．ここで，

$$a_{sj} = [a_{sj}] + f_{sj} = [a_{sj}] + n_{sj}/d_{sj} \quad (j \in J')$$

（ただし，$f_{sj}=0$のときは$n_{sj}=0$, $d_{sj}=1$とし，n_{sj}/d_{sj} は既約分数）

とし，d_{sj}($j \in J'$)の最小公倍数をＤとすると，一般にカットはｈ（$0 \leqq h \leqq D-1$）を用いて，

$$v = -\mathrm{frac}(f_{s0} \cdot h) - \Sigma\, \mathrm{frac}(f_{sj} \cdot h)(-x_j) \geqq 0 \qquad (j \in J)$$

となる．ｖはカットのスラック変数である．カット生成方法として以下の３つを考える．

α：$h=1$とする．

$$v = -f_{s0} - \Sigma f_{sj}(-x_j) \geqq 0 \qquad (j \in J)$$

β：$h=D-1$とする．

$$v = -1 + f_{s0} - \Sigma f_{sj}{}^{C}(-x_j) \geqq 0 \qquad (j \in J)$$

$$\text{ただし，} f_{sj}{}^{C} = \begin{cases} 0 & (f_{sj}=0) \\ 1 - f_{sj} & (f_{sj} \neq 0) \end{cases}$$

γ：$h = [d_{s0}/2]$ とする．

カットαは，Gomoryによるカットで文献[1]ではｆカットと呼ばれている．また，カットβは文献[1]でf^{C}カットと呼ばれている．そして，カットγは今回の実験で新しく考えたものである．

Implementation of Fractional Algorithm in Rational Arithmetics and Its Computational Results.
Toshio OHYANAGI and Ikuo KAJI
HOKKAIDO UNIVERSITY

(3) カットの除去方法

以前に加えられたカットのスラック変数が再度基底に正の値をとって戻った場合，このカットは現在制約とはなっていないので除去することができる．こうすることにより列タブローの拡大を抑えることができる。しかし，除去したカットを後になって再度生成する必要が生じることもあり、そのまま残しておいてもよい．カットの除去に関して以下の５つを考える．

ａ：カットを除去しない．

ｂ：カットを除去可能なときに除去する．

ｃ：カットを５面加える毎に除去可能性を調べ除去する．

ｄ：カットを10面加える毎に除去可能性を調べ除去する．

ｅ：列タブローの行数がm+2n+1に達したときにカットの除去可能性を調べ除去する．

【４】プログラムの作成

プログラムの開発は，パソコン(PC-9801VX,CPU:80286)上でModula-2言語(Logitech Modula-2 V3.0)を使用して行った．その際，丸め誤差を排除するために計算をすべて有理数演算で行うようにした．Modula-2言語は，仕様として有理数型および有理数演算をサポートしていないので，まず有理数演算ライブラリーを作成した．そしてそのライブラリーをもとに，【３】で説明したカットの生成・除去方法のすべての組合せ（45通り）を実現するプログラムを文献[１]を参考にして作成した．プログラムでは，パラメータの設定によりカットの生成・除去方法を変えるようにしている．

【５】数値実験と結果

数値実験では，まず【３】で説明したカットの生成・除去方法のすべての組合せ（45通り）に対して，最適解を得るまでに加えたカット数を調べた（表１）．カットを100 面加えても最適解が得られない場合は，計算を打切ることにした（表中の＊は打切ったことを表わす）．なお，この実験で用いた問題は，①と②が文献[２]の例題，③は文献[３]で事例として用いられている問題，④～⑥は文献[４]のランダム問題である．

表１　最適解を得るまでに加えたカット数
ただし，＊は100 面カットを加えても解が得られなかったことを表わす．

	カットの生成・除去の方法			問題番号およびサイズ（行数，列数）					
				①(1,4)	②(3,2)	③(21,8)	④(10,7)	⑤(15,10)	⑥(15,10)
1	Ⅰ	α	a	4	4	36	6	39	＊
2	Ⅰ	α	b	4	4	＊	6	45	＊
3	Ⅰ	α	c	4	4	30	6	＊	＊
4	Ⅰ	α	d	4	4	＊	6	＊	＊
5	Ⅰ	α	e	4	4	＊	6	＊	＊
6	Ⅰ	β	a	2	2	12	5	20	13
7	Ⅰ	β	b	2	2	82	5	28	13
8	Ⅰ	β	c	2	2	12	5	27	13
9	Ⅰ	β	d	2	2	12	5	20	13
10	Ⅰ	β	e	2	2	12	5	20	13
11	Ⅰ	γ	a	2	2	9	4	18	26
12	Ⅰ	γ	b	2	2	9	4	14	20
13	Ⅰ	γ	c	2	2	9	4	21	18
14	Ⅰ	γ	d	2	2	9	4	18	26
15	Ⅰ	γ	e	2	2	9	4	18	18
16	Ⅱ	α	a	2	2	5	4	15	4
17	Ⅱ	α	b	2	2	5	4	14	4
18	Ⅱ	α	c	2	2	5	4	12	4
19	Ⅱ	α	d	2	2	5	4	15	4
20	Ⅱ	α	e	2	2	5	4	15	4
21	Ⅱ	β	a	4	4	＊	11	＊	＊
22	Ⅱ	β	b	4	4	＊	39	＊	＊
23	Ⅱ	β	c	4	4	＊	11	＊	＊
24	Ⅱ	β	d	4	4	＊	11	＊	＊
25	Ⅱ	β	e	4	4	＊	11	＊	＊
26	Ⅱ	γ	a	2	2	20	4	27	16
27	Ⅱ	γ	b	2	2	17	4	24	18
28	Ⅱ	γ	c	2	2	20	4	24	18
29	Ⅱ	γ	d	2	2	20	4	27	16
30	Ⅱ	γ	e	2	2	20	4	24	18
31	Ⅲ	α	a	4	4	＊	11	＊	＊
32	Ⅲ	α	b	4	4	＊	17	＊	＊
33	Ⅲ	α	c	4	4	＊	11	＊	＊
34	Ⅲ	α	d	4	4	＊	11	＊	＊
35	Ⅲ	α	e	4	4	＊	11	＊	＊
36	Ⅲ	β	a	2	2	2	4	13	4
37	Ⅲ	β	b	2	2	2	4	14	4
38	Ⅲ	β	c	2	2	2	4	12	4
39	Ⅲ	β	d	2	2	2	4	13	4
40	Ⅲ	β	e	2	2	2	4	13	4
41	Ⅲ	γ	a	2	2	21	4	9	50
42	Ⅲ	γ	b	2	2	40	4	9	18
43	Ⅲ	γ	c	2	2	32	4	9	18
44	Ⅲ	γ	d	2	2	32	4	9	18
45	Ⅲ	γ	e	2	2	17	4	9	18

この結果，すべての組合せに対していつでも容易に最適解が得られるとは限らず，カットを100 面加えても解が得られない場合もあり，カットの生成・除去の方法が重要であることがわかった．この実験で用いた問題６題に対しては，カット生成行の選択と生成方法の組合せが，(Ⅰ, β), (Ⅰ, γ), (Ⅱ, α), (Ⅱ, γ), (Ⅲ, β), (Ⅲ, γ) のときに，カット除去の方法によらずにすべて最適解が得られた．特に，カット生成方法がγの場合には，カット生成行選択方法にもよらずに最適解が得れれた．これは興味深い結果である．また，(Ⅱ, α), (Ⅲ, β) の組合せが他の組合せよりも加えるカット数が少なくて済むことがわかる．

そこでつぎに，この(Ⅱ, α), (Ⅲ, β) ２つの組合せに対してさらにランダム問題を30題（サイズ15×10を20題，25×15を５題，35×20を５題）解き，これらの組合せが多くの問題に対して有効であるかどうかを調べた．その結果，組合せが(Ⅱ, α) の場合，カットの除去方法によらずすべての問題に対して最適解を得ることができた．特に，カット除去方法としてｂを用いると，加えるカット数が30面を越えることはないという結果が得られた．これに対して(Ⅲ, β) の組合せでは，カットの除去方法によっても異なるが解を得られない問題が数題あった．この２つの組合せは，ともに，それぞれのカット生成方法に対してカットの定数項 $-\mathrm{frac}(f_{s0}\cdot h)$ を最小にする行を生成行として選択するものであるが，今回行った実験では，(Ⅱ, α) の組合せのほうが良いことになった．

以上のことより，現在までの実験からはカット生成・除去方法の組合せとして(Ⅱ, α, b)が多くの問題に対して有効であり，この組合せにした場合，少ないカット数で最適解が得られる可能性が大きいと予想される．ただし，すべての問題に対して少ないカット数で最適解が得られる保証はない．

【６】おわりに

今回，小数法を有理数演算を使って実現し，小数法の有効性を調べる数値実験を行った．その結果，カット生成・除去方法の組合せとして(Ⅱ, α, b)が有効であるという結果が得られた．今後さらに多くの問題に対してこの組合せに対する実験を行い，この組合せ方法の有効性を確認する予定である．また，今回はあまり行えなかったがカットγに対する実験も行う予定である．

なお，数値実験の詳細および実行時間については，講演当日説明を行う．

【参考文献】

[1]今野他著:「整数計画法と組合せ最適化」日科技連

[2]今野著:「整数計画法」産業図書

[3]ﾏｸﾐﾗﾝ著，前田他訳:「数理計画法２」産業図書

[4]W.Y.L.Benjamin,L.R.Ronald:"Development of a Parametric Generating Procedure for Integer Programming TestProblems",J.ACM,Vol24,No3, pp465-472