Journal of the Operations Research
Society of Japan
Vol. 19, No. 3, September, 1976

A PRIMAL CUTTING PLANE ALGORITHM FOR INTEGER FRACTIONAL PROGRAMMING PROBLEMS

HIROAKI ISHII, *Osaka University*

TOSHIHIDE IBARAKI and HISASHI MINE, *Kyoto University*

(Received January 7, 1976; Revised March 30, 1976 and July 1, 1976)

Abstract

A primal cutting plane algorithm is proposed for the integer fractional programming problem:

$$\text{maximize} \quad (c_0 + \sum_{j=1}^{n} c_j x_j)/(d_0 + \sum_{j=1}^{n} d_j x_j)$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i , \quad i=1,2,\cdots,m$$

$$x_j \geq 0 , \quad \text{integer}, \quad j=1,2,\cdots,n$$

The algorithm is obtained by slightly modifying *Young's simplified primal algorithm* developed for the ordinary integer programming problem, and is based on the parametric programming approach to the fractional problem given by Jagannathan and Dinkelbach.

1. Introduction

Research on fractional programming problems has been concentrated on continuous type problems with a linear fractional objective function. This paper, however, discusses the integer programming problem with a linear fractional objective function. In addtion to the known methods ([1], [5], [6], [9])

for the integer fractional programming problem or some special cases, this paper proposes a new method, which may be considered as an integer programming version of the well known *Martos'method* for the linear fractional problems with continuous variables [8]. The adaptation to the integer problem is made possible by applying *Young's simplified primal algorithm (SPA)* for the integer programming problem, in a parametric programming manner proposed by Jagannathan [7] and Dinkelbach [2].

In Section 2, the problem description and some assumptions are given. In Section 3, a subsidiary problem $P(\lambda)$ is defined, and close relations between $P(\lambda)$ and the original integer fractional problem P are discussed. Based on these relations, an algorithm is proposed in Section 4. Following an example given in Section 5, Section 6 proves the validity of the algorithm and its finiteness. In Section 7 some further considerations are given.

2. Integer Fractional Programming Problem

This paper treats the following integer fractional problem P:

$$\text{maximize} \qquad N(x)/D(x) = \frac{\sum_{j=1}^{n} c_j x_j + c_0}{\sum_{j=1}^{n} d_j x_j + d_0}$$

$$\text{subject to} \qquad x \in S \ ,$$

where

$$S = \{x = (x_1, \cdots, x_n) \mid \sum_{j=1}^{n} a_{ij} x_j \leqq b_i \ , \quad i=1,2,\cdots,m$$

$$\text{and} \quad x_j \geqq 0, \quad \text{integer}, \quad j=1,2,\cdots,n\}$$

The followings are assumed throughout this paper.

(1) Coefficients c_j, d_j, b_i, a_{ij} are all integers.

(2) S is bounded and nonempty.

(3) $D(x) > 0$ for all $x \in S$.

3. Subsidiary Problem $P(\lambda)$ and its Relations to P

Subsidiary Problem $P(\lambda)$ is defined for a real number λ as follows.

$$P(\lambda) \ ; \ \text{maximize} \qquad N(x) - \lambda D(x) = \sum_{j=1}^{n} (c_j - \lambda d_j) x_j + c_0 - \lambda d_0$$

subject to $x \in S$.

Let $Z(\lambda)$ be the optimal value of $P(\lambda)$. It is well known that P and $P(\lambda)$ are closely related to each other as follows. (Theorem 3.1 and Theorem 3.2 are due to [2], [7].)

Theorem 3.1. Let λ^* be the optimal value of P. Then, it holds that

$$\text{(i)} \quad \lambda < \lambda^* \iff Z(\lambda) > 0 ,$$
$$\text{(ii)} \quad \lambda = \lambda^* \iff Z(\lambda) = 0 ,$$
$$\text{(iii)} \quad \lambda > \lambda^* \iff Z(\lambda) < 0 .$$

Theorem 3.2. $Z(\lambda)$ is a strictly decreasing convex function of λ.

By theorem 3.1, the problem P reduces to finding a λ such that $Z(\lambda)=0$.

Theorem 3.3. For $x, \bar{x} \in S$ and $\lambda = N(x)/D(x)$,

$$N(\bar{x}) - \lambda D(\bar{x}) > 0$$

if and only if

$$N(\bar{x}) / D(\bar{x}) > N(x) / D(x) .$$

Proof. $N(\bar{x}) - \lambda D(\bar{x}) > 0 \iff N(\bar{x}) - \frac{N(x)}{D(x)} D(\bar{x}) > 0 \iff \frac{N(\bar{x})}{D(\bar{x})} - \frac{N(x)}{D(x)} > 0$

(since $D(\bar{x}) > 0$). □

The property of Theorem 3.3 plays an important role in the development of our algorithm.

4. An Algorithm for the Integer Fractional Programming Problem

Before describing an algorithm for the integer fractional programming problem, we shall briefly review Young's SPA used to solve the ordinary integer programming problems ([3, 11, 13, 14]). Young's SPA starts with the following integer programming problem as the initial tableau:

$$\text{Maximize} \quad x_o = c_0 + \sum_{j=1}^{n} (-c_j)(-x_j)$$

$$\text{(4.1)} \quad \text{subject to} \quad x_{n+i} = b_i + \sum_{j=1}^{n} a_{ij}(-x_j) , \quad i=1,2,\cdots,m,$$

$$x_k \geq 0, \quad k=1,2,\cdots,n+m,$$

where the primal feasibility condition

$$b_i \geqq 0 , \quad i=1,2,\cdots,m$$

is assumed, and c_j, a_{ij}, b_i are all integer constants.

To deal with a tableau obtained after pivot operations, in general, let us assume that the following is the current tableau.

$$\text{maximize} \quad x_0 = \alpha_{00} + \sum_{j=1}^{n} \alpha_{0j}(-t_j)$$

$$\text{subject to} \quad u_i = \alpha_{i0} + \sum_{j=1}^{n} \alpha_{ij}(-t_j) , \quad i=1,2,\cdots,m'$$

$$t_j \geqq 0 , \quad j=1,2,\cdots,n \tag{4.2}$$

$$u_i \geqq 0 , \quad i=1,2,\cdots,m' ,$$

where the primal feasibility

$$\alpha_{i0} \geqq 0 , \quad j=1,2,\cdots,n$$

is also assumed; u_i denotes the current basic variables and t_j nonbasic variables.

If this tableau satisfies the dual feasibility condition also, i,e.,

$$\alpha_{0j} \geqq 0 , \quad j=1,2,\cdots,n,$$

then the current tableau provides an optimal solution and computation terminates. Otherwise, a cut is generated according to a certain rule, and a pivot operation is performed on this cut row. (m' in(4.2) includes the number of the generated cuts.) The resulting tableau satisfies one of the following three conditions:

(i) It is dual feasible. Then Young's SPA terminates.

(ii) It is not dual feasible, but will be able to satisfy $\alpha'_{00} > \alpha_{00}$ in the next tableau, where α'_{00} is the new coefficient α_{00} **after** the pivot operation is executed. (This case is called a *transition cycle*.) Then the same procedure is repeated by regarding the resulting tableau as (4.2) until case (i) or case (iii) is reached. (If the tableau has L-row (introduced in (iii) below), it is deleted before the pivot operation.)

(iii) It is not dual feasible, and $\alpha'_{00} = \alpha_{00}$ will hold in the next tableau. (This case is called a *stationary cycle*.) A pivot element is determined by a rule based on a special row, called *L-row*. (If the current tableau does not have *L-row*, a pivot element is selected after generating *L-row*.) Then pivot operation is performed. After repeating this process finite times, it is known that either case (i) or case (ii) is eventually reached.

By repeating the above cycles, Young's SPA guarantees that case (i) is eventually reached and an optimal solution of (4.1) is obtained. Now note that the problem P(λ) is rewritten as follows.

$$
\begin{aligned}
P(\lambda): \quad & \text{maximize} && x_0 = c_0 - d_0 + \sum_{j=1}^{n} \{(-c_j) - \lambda(-d_j)\}(-x_j) \\
& \text{subject to} && z_1 = c_0 + \sum_{j=1}^{n} (-c_j)(-x_j) \\
(4.3) \qquad & && z_2 = d_0 + \sum_{j=1}^{n} (-d_j)(-x_j) \\
& && x_{n+i} = b_i + \sum_{j=1}^{n} a_{ij}(-x_j), \quad i=1,2,\cdots,m \\
& && x_j \geqq 0, \text{ integer}, \quad j=1,2,\cdots,n+m.
\end{aligned}
$$

This differs from (4.1) only in that the objective function is parametrized by λ, and rows z_1 and z_2 are augmented. The primal feasibility

$$b_i \geqq 0, \quad i=1,2,\cdots,m,$$

is also assumed. For a fixed λ, our algorithm is exactly the same as Young's SPA. Two rows z_1 and z_2 corresponding to $N(x)$ and $D(x)$ respectively, are used to compute the new objective row x_0 when λ is modified. To describe a general step of our algorithm, let the current tableau be as follows:

$$
\begin{aligned}
& \text{maximize} && x_0 = \alpha_{00}(\lambda) + \sum_{j=1}^{n} \alpha_{0j}(\lambda)(-t_j) \\
& \text{subject to} && z_1 = \beta_{00} + \sum_{j=1}^{n} \beta_{0j}(-t_j)
\end{aligned}
$$

$$z_2 = \gamma_{00} + \sum_{j=1}^{n} \gamma_{0j}(-t_j)$$

$$u_i = \alpha_{i0} + \sum_{j=1}^{n} \alpha_{ij}(-t_j) , \quad i=1,2,\cdots,m' ,$$

$$t_j \geqq 0 , \quad j=1,2,\cdots,n$$

$$u_i \geqq 0 , \quad i=1,2,\cdots,m'$$

This tableau satisfies the primal feasibility

$$\alpha_{i0} \geqq 0 , \quad i=1,2,\cdots,m'$$

and all coefficients α_{ij}, β_{0j}, γ_{0j} are integers. The objective row x_0 is related to the z_1 and z_2 rows in the following way:

$$(4.4) \qquad \alpha_{0j}(\lambda) = \beta_{0j} - \lambda\gamma_{0j} , \quad j=1,2,\cdots,n .$$

Our algorithm starts with the initial tableau (4.3) with

$$\lambda = c_0 / d_0$$

(this makes $\alpha_{00}(\lambda)=0$). If the initial tableau does not satisfy the dual feasibility condition, the above Young's SPA is applied until coefficient $\alpha_{00}(\lambda)$ strictly increases (i.e., $\alpha_{00}(\lambda) > 0$) or the dual feasibility condition (with $\alpha_{00}(\lambda)=0$) is satisfied. As noted above, Young's SPA always produces one of the two results in finite pivot operations. If a dual feasible tableau is obtained, computation terminates and the resulting tableau provides an optimal solution. If $\alpha_{00}(\lambda) > 0$† is obtained, however, λ is updated to $\lambda = \beta_{00}/\gamma_{00}$ (this is justified by Theorem 3.3) so that $\alpha_{00}(\lambda)=0$ and the objective row x_0 is recalculated by (4.4). Then the above procedure is repeated.

As shown in Theorem 6.1, the entire computation eventually terminates and an optimal solution is obtained.

---

† $\alpha_{00}(\lambda)>0$ corresponds to $\alpha'_{00} > \alpha_{00}$ (the condition to enter the transition cycle).

An Algorithm for the Integer Fractional Programming Problem

Step 1 (Initialize)††: Let $\lambda \leftarrow N(x^I)/D(x^I)$,

$$x_j^I = 0 , \quad j=1,2,\cdots,n .$$

Step 2 (Check the optimality)†††: If

$$\alpha_{0j}(\lambda) \geqq 0 , \quad j=1,2,\cdots,n ,$$

go to Step 4. Otherwise, add a cut and apply a pivot operation according to the rule for the transition cycle of Young's SPA (see Appendix for details).

Step 3 : If $\alpha_{00}(\lambda) > 0$, let $\lambda \leftarrow \frac{\beta_{00}}{\gamma_{00}}$ and update the x_0-row by (4.4). Return to Step 2. Otherwise return to Step 2 directly.

Step 4 (Terminate) : Terminate. The current λ is the optimal value of P and

$$u_i = \alpha_{i0} , \quad i=1,2,\cdots,m'$$

$$t_j = 0 , \quad j=1,2,\cdots,n$$

is an optimal solution of P.

---

†† We assume that the tableau (4.3) obtained from the above λ is primal feasible, i.e.,

$$b_i \geqq 0 , \quad i=1,2,\cdots,m .$$

It obviously satisfies $\alpha_{00}(\lambda) = 0$. If (4.3) does not provide a primal feasible tableau, a primal feasible tableau has to be obtained by some means. This point is not discussed in this paper, since it is the same as the ordinary integer programming problem.

††† Whenever Step 2 is entered, $\alpha_{00}(\lambda) = 0$ is always satisfied.

5. Example

Consider the following problem P (*see Fig. 5.1*).

P :

$$\text{Maximize} \quad \frac{N(x)}{D(x)} = \frac{3 + 7x_1 + 9x_2}{2 + 3x_1 + 4x_2}$$

$$\text{subject to} \quad x_3 = 6 + 2(-x_1) + 3(-x_2)$$

$$x_4 = 5 + 3(-x_1) + 2(-x_2)$$

$$x_1, x_2, x_3, x_4 \geqq 0 \ , \quad \text{integer} \ .$$

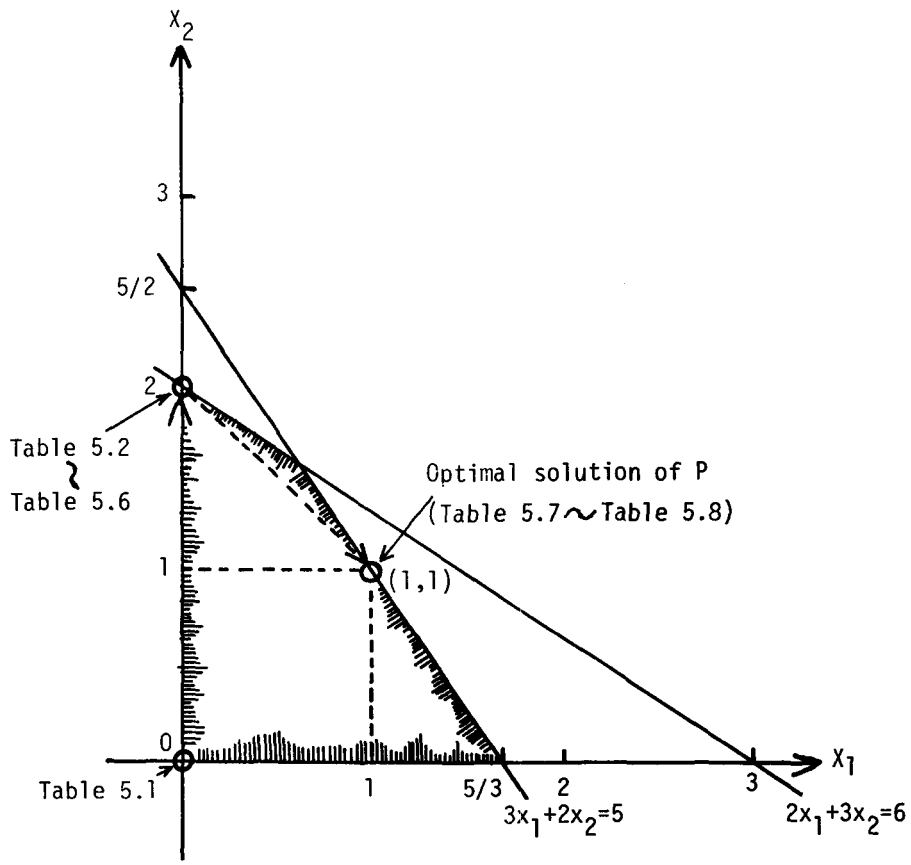

Figure 5.1 Illustration of computation process for the example in Section 5

From the constraints we have

$$(5.1) \qquad 0 \leq x_1 \leq \min\left[\left\lfloor \frac{6}{2} \right\rfloor, \left\lfloor \frac{5}{2} \right\rfloor\right] = 1 \ (= d_1),$$

$$0 \leq x_2 \leq \min\left[\left\lfloor \frac{6}{3} \right\rfloor, \left\lfloor \frac{5}{2} \right\rfloor\right] = 2 \ (= d_2)$$

$$0 \leq x_3 \leq 6 \ (= d_3), \qquad 0 \leq x_4 \leq 5 \ (= d_4),$$

where $\lfloor x \rfloor$ denotes the integer part of x.

Step 1 : Let $\lambda \leftarrow \frac{N(x^I)}{D(x^I)} \ \left(= \frac{c_0}{d_0} = -\frac{3}{2}\right)$ and construct the initial tableau *(Table 5.1)* for

$$x^I = (x_1^I, x_2^I, x_3^I, x_4^I) = (0, 0, 6, 5).$$

Step 2 : Since $\alpha_{01}(\lambda)=-5/2$, $\alpha_{02}(\lambda)=-6/2$, the dual feasibility condition is not satisfied. We have $J=\{1,2\}$ according to *Substep 1* of Young's SPA since $\theta_1=\min\{6/2, 5/3\}=5/3\geq 1$, $\theta_2=\min\{6/3, 5/2\}=2\geq 0$. In *Substep 2*, x_2 column is chosen as a pivot column though x_1 column is also possible. In *Substep 5*, cut

$$(5.2) \qquad s_1 = 2 + 0(-x_1) + 1(-x_2)$$

is generated and added to the tableau (see s_1-row of *Table 5.1*) and execute a pivot operation (* denotes the pivot element). The resulting tableau is given in *Table 5.2*. Obviously

$$0 \leq s_1 \leq 2$$

follows from (5.1) and (5.2).

Step 3 : $\alpha_{00}(\lambda)=6>0$. Thus let $\lambda \leftarrow \beta_{00}/\gamma_{00}=21/10$, and recalculate the x_0-row of *Table 5.2* by using the new λ. *Table 5.3* shows the resulting tableau. Return to Step 2.

Step 2 : $\alpha_{01}(\lambda)=-7/10<0$; the dual feasibility is not satisfied. Since $J=\phi$ in this case and the tableau does not have L-row, L-row

$$X_L = 3 + 1(-x_1) + 1(-s_1)$$

Table 5.1. First tableau

	1	$-x_1$	$-x_2$
x_0 =	0	-5/2	-6/2
z_1 =	3	-7	-9
z_2 =	2	-3	-4
x_3 =	6	2	3
x_4 =	5	3	2
s_1 =	2	0	1*

Table 5.2. Second tableau

	1	$-x_1$	$-s_1$
x_0 =	6	-5/2	6/2
z_1 =	21	-7	9
z_2 =	10	-3	4
x_3 =	0	2	-3
x_4 =	1	3	-2
x_2 =	2	0	1

Table 5.3. Third tableau

	1	$-x_1$	$-s_1$
x_0 =	0	-7/10	6/10
z_1 =	21	-7	9
z_2 =	10	-3	4
x_3 =	0	2	-3
x_4 =	1	3	-2
x_2 =	2	0	1
x_L =	3	1	1
s_2 =	0	1*	-2

Table 5.4. Fourth tableau

	1	$-s_2$	$-s_1$
x_0 =	0	7/10	-8/10
z_1 =	21	7	-5
z_2 =	10	3	-2
x_3 =	0	-2	1*
x_4 =	1	-3	4
x_2 =	2	0	1
x_L =	3	-1	3
x_1 =	0	1	-2

is added in *Substep 3* (see x_L-row of *Table 5.3*; $\alpha_{L0}=3(=1+2)$ since $0\leqq x_1 \leqq 1$ and $0\leqq s_1 \leqq 2$). According to the rule for the stationary cycle of Young's SPA, x_1-column is selected as the pivot column in *Substep 4*. Then cut row is added to the tableau (see s_2-row of *Table 5.3*).

$$0 \leqq s_2 \leqq 4$$

follows from s_2-row and $0\leqq s_1 \leqq 2$. The resulting tableau after a pivot operation is given in *Table 5.4*.

Step 3 : $\alpha_{00}(\lambda)=0$. Return to Step 2.

Step 2 : $\alpha_{00}(\lambda)=-8/10<0$; the dual feasibility is not satisfied.

Although $J=\phi$, *Substep 3* is skipped since *Table 5.4* has L-row. In *Substeps 4* and *5*, a cut is generated. In this case, however, the generated cut is the same as x_3-row, and x_3-row is used as the pivot row. After a pivot operation, *Table 5.5* is obtained.

Step 3 : $\alpha_{00}(\lambda)=0$. Return to Step 2.

Step 2 : $\alpha_{01}(\lambda)=-9/10<0$. According to *Substeps 4* and *5*, add a cut

$$s_3 = 0 + 1(-s_2) + (-1)(-x_3)$$

to the tableau, and execute a pivot operation, to obtain *Table 5.6*.

Step 3 : $\alpha_{00}(\lambda)=0$. Return to Step 2.

Step 2 : $\alpha_{02}(\lambda)=-1/10<0$. Since $J=\{2\}$, in this case, a cut is generated according to *Substeps 2* and *5*. (Note that L-row is deleted). The generated cut is the same as x_4-row. After a pivot operation, *Table 5.7* is obtained.

Step 3 : $\alpha_{00}(\lambda)=1/10>0$. Then let $\lambda \leftarrow \beta_{00}/\gamma_{00}=19/9$ and recalculate x_0-row. *Table 5.8* is obtained.

Step 2 : $\alpha_{01}(\lambda)=3/9>0$, $\alpha_{02}(\lambda)=1/9>0$; the dual feasibility is satisfied. Go to Step 4.

Step 4 : Terminate. An optimal solution x^0 is given by

$$x_1^0 = x_2^0 = x_3^0 = 1 , \quad x_4^0 = 0$$

$N(x^0)/D(x^0)=\lambda=19/9$.

Table 5.5. Fifth tableau

	1	$-s_2$	$-x_3$
x_0 =	0	-9/10	8/10
z_1 =	21	-3	5
z_2 =	10	-1	2
x_4 =	1	5	4
x_2 =	2	2	-1
x_1 =	0	-3	2
x_L =	3	5	-3
s_3 =	0	1*	-1

Table 5.6. Sixth tableau

	1	$-s_3$	$-x_3$
x_0 =	0	9/10	-1/10
z_1 =	21	3	2
z_2 =	10	1	1
x_4 =	1	-5	1*
x_2 =	2	-2	1
x_1 =	0	3	-1
x_L =	3	-5	2

Table 5.7. Seventh tableau

	1	$-s_3$	$-x_4$
x_0 =	1/10	4/10	1/10
z_1 =	19	13	-2
z_2 =	9	6	-1
x_3 =	1	-5	1
x_2 =	1	3	-1
x_1 =	1	-2	1

Table 5.8. Eighth tableau

	1	$-s_3$	$-x_4$
x_0 =	0	3/9	1/9
z_1 =	19	13	-2
z_2 =	9	6	-1
x_3 =	1	-5	1
x_2 =	1	3	-1
x_1 =	1	-2	1

6. Proof of Finiteness and Validity

Theorem 6.1 (Finiteness and Validity). The procedure given in Section 4 terminates in a finite number of iterations and, upon termination, produces an optimal solution of P.

Proof. *Finiteness* : (a) From Assumption 2 of Section 2, the number of feasible solution in S is finite.

(b) Each $P(\lambda)$ has the same feasible region S, and each tableau represents a feasible solution since the primal feasiblity is assumed. When λ is updated in Step 3 or Step 5, the new λ satisfies,

$$\lambda = \frac{\beta_{00}}{\gamma_{00}} = \frac{N(\bar{x})}{D(\bar{x})} ,$$

where $\bar{x}$ is the feasible solution represented by the tableau. By Theorem 3.3, this new λ is strictly greater than the old λ.

(c) It is known that, for each λ, condition $\alpha_{00}(\lambda)>0$ (then λ is updated) or a dual feasibility (termination) is satisfied after a finite number of pivot operations. (This property was first proved by Young [13, 14] under the assumption that all coefficients in a tableau are integers. Salkin, Schroff and Mehta [12] generalized this property to the case in which each α_{ij} is rational. Note that $\alpha_{0j}(\lambda)$ is rational in our algorithm.) (a) (b) (c) together prove the finiteness.

Validity : When Step 6 is reached, the tableau satisfies both primal and dual feasibility conditions, and hence it gives an optimal solution of $P(\lambda)$ for the current λ. Since the tableau also satisfies $\alpha_{00}(\lambda)=0$, this solution is an optimal solution of P by Theorem 3.1. □

7. Discussion

It is shown in this paper that the primal cutting plane algorithm, Young's SPA, for the ordinary integer programming problem can be easily modified to handle the integer fractional programming problem. In view of this result, it appears possible to modify other primal algorithms such as *Glover's simplified primal algorithm* [4] (which is favorably compared to Young's SPA in [10] from the view point of computational efficiency) in a similar manner to accept the fractional problem. However, a straightforward application of the present technique to Glover's SPA seems to cause a difficulty that the required property of the reference equation (which plays in Glover's SPA, a role similar to the L-row of Young's SPA) is no longer preserved when λ is updated.

Thus it would be a subject for future research to find a generation method of the reference equation when λ is updated.

Acknowledgement

The authors thank Professor T. Hasegawa of Kyoto University and Professor T. Nishida of Osaka University for their suggestion and comment.

References

[1] Anzai, Y., "On Integer Fractional Programming", *Journal of Operations Research Society of Japan*, 17 (1974), 49-66.

[2] Dinkelbach, W., "On Nonlinear Fractional Programming", *Management Science*, 13 (1967), 492-498.

[3] Garfinkel, R.S. and G.L. Nemhauser, *Integer Programming*, John Wiley & Sons, New York, N. Y., 1972.

[4] Glover, F., "A New Foundation for a Simplified Primal Integer Programming Algorithm", *Operations Research* 16 (1968), 727-740.

[5] Gruspan, M., "Hyperbolic integer programming", *Naval Research Logistics Quarterly*, 20 (1973), 341-356.

[6] Hammer, P.L. and S. Redeanu, *Boolean Method in Operations Research and Related Areas*, Springer-Verlag, New York, 1968.

[7] Jagannathan, R., "On Some Properties of Programming Problems in Parametric Forms Pertaining to Fractional Programming", *Management Science*, 12 (1966), 609-615.

[8] Martos, B., "Hyperbolic Programming", *Naval Research Logistics Quaterly*, 11 (1964), 135-155.

[9] Robillard, P., "(0,1) Hyperbolic Programming Problems", *Naval Research Logistics Quarterly*, 18 (1971), 47-57.

[10] Salkin, H., "A Note Comparing Glover's and Young's Simplified Primal Algorithms", *Naval Research Logistics Quarterly*, 19 (1972), 399-402.

[11] Salkin, H., *Integer Programming*, Addison Wesley, Menlo Park, California, 1975.

[12] Salkin, H., P. Shroff and S. Mehta, "Primal All-Integer Integer Programming Applied to Tableaux with Rational Coefficents". *Technical Memorandum, No. 298*, Department of Operations Research, Case Western Reserve University April 1974.

[13] Young, R., "A Primal (All-Integer) Integer Programming Algorithm", *Journal of Research of the National Bureau of Standards*, 69B (1963),

213-249.

[14] Young, R., "A Simplified (All-Integer) Integer Programming Algorithm", *Operations Research*, 16 (1968), 750-782.

(The Authors' address : Department of Applied Physics, Faculty of Engineering, Osaka University, Suita Japan.
Department of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University, Kyoto, Japan.)

Appendix. Description of Cut Generation Procedure by Young's SPA ([14])

The generation of a cut and the selection of a pivot element in Young's SPA are done according to the following rule. α_j $(j=1,2,\cdots,n)$ is used to denote the j-th column of a tableau. J computed during computation determines whether the present cycle is transitive or stationary; $J\neq\phi$ implies that $\alpha'_{00}>\alpha_{00}$ holds in the next tableau (i.e., it is a transition cycle), while $J=\phi$ implies that $\alpha'_{00}=\alpha_{00}$ holds in the next tableau (i.e., a stationary cycle).

Substep 1 : (This substep is entered from Step 2 of the main algorithm in Section 4): Calculate $\theta_j=min\{\alpha_{i0}/\alpha_{ij} \mid \alpha_{ij}>0$, i does not correspond to x_0-row, z_1-row, z_2-row or L-row$\}$ for each j satisfying $\alpha_{0j}(\lambda)<0$. $J\leftarrow\{j \mid \theta_j\geq 1\}$.

Go to Substep 2 if $J\neq\phi$, and go to Substep 3 if $J=\phi$.

Substep 2 : If the current tableau has *L-row*, delete it from the tableau. Select any column α_{j_0}, $j_0\in J$, as the pivot column. Go to Substep 5.

Substep 3 : If the current tableau has *L-row*, go to Substep 4. Otherwise add the following *L-row* to the tableau, and go to Substep 4.

$$x_L = \alpha_{L0} + \sum_{j=1}^{n} (-t_j) \quad ,$$

where

$$\alpha_{L0} = \sum_{j=1}^{n} d_j$$

and d_j is an upper bound of t_j, i.e., $0\leq t_j\leq d_j$ for any $x\in S$. (The example in Section 5 includes a method for obtaining d_j.)

Substep 4 : Let α_{Lj} (j=1,2,$\cdots$,n) denote the elements in the *L-row*. For each α_j, $j=1,2,\cdots,n$, that has $\alpha_{Lj}>0$, calculate column

$$R_j = (\alpha_{0j} / \alpha_{Lj} \ , \ \alpha_{1j} / \alpha_{Lj} \ , \cdots , \ \alpha_{m'j} / \alpha_{Lj})^T \ ,$$

where T denotes transpose, and select the lexicographically smallest column among them as the pivot column α_{j_0}. Go to Substep 5.

Substep 5 : Calculate i_0 such that

$$\alpha_{i_0 0} / \alpha_{i_0 j_0} = \min \{\alpha_{i0} / \alpha_{ij_0} \mid \alpha_{ij_0} > 0 , \quad i=1,2,\cdots,m'\} ,$$

and generate the following Gomory cut to adjoin the tableau.

$$s = \lfloor \alpha_{i_0 0}/\mu \rfloor + \sum_{j=1}^{n} \lfloor \alpha_{i_0 j}/\mu \rfloor (-t_j) ,$$

where $\mu=\alpha_{i_0 j_0}$ (>0) and $\lfloor y \rfloor$ denotes the integer part of y.

Execute a pivot operation on the pivot element $\lfloor \alpha_{i_0 j_0}/\mu \rfloor$ (=1) of the *s-row*.

(Then return to Step 3 of the main algorithm).