Journal of Mathematical Sciences: Advances and Applications
Volume 1, Number 2, 2008, Pages 265-278

THE STOPPED SIMPLEX ALGORITHM FOR INTEGER LINEAR PROGRAMS WITH SPECIAL CUTS

PEI-WANG GAO and XING CAI

Department of Finance
Guangxi University of Finance and Economics
Nanning 530003
P. R. China
e-mail: pwgao@yahoo.cn

Abstract

This paper presents an improved stopped simplex algorithm that aims to further decrease the stopped number of the variables. First of all, two special cuts are generated by introducing a linear transformation to cut the intersection of the objective function hyperplane and the feasible region of the linear programming relaxation problem. So, the cuts lead to the more narrow intervals of the variables on the objective function hyperplane. Secondly, the stopped simplex algorithm with the cuts is carried out to do a search on the objective function hyperplane. Finally, a test on some classical numerical examples is made. It shows that the algorithm presented here is more efficient and potential, compared with Thompson's algorithm.

1. Introduction

The classical cutting plane method and the branch-and-bound principle are the most popular algorithms for integer linear programs (ILP in short). They always solve a series of linear programming

2000 Mathematics Subject Classification: 90C05, 90C10.

Keywords and phrases: linear programming, integer programming, objective function hyperplane, simplex algorithm, stopped simplex algorithm.

This research is supported by the Guangxi Scientific Fund (Grant No. GKZ0728260).

Received May 26, 2008

subproblems generated by cuts or branches. For this reason, various researchers expended a considerable amount of effort in trying to improve the efficiency of cutting and branching. The most important surveys of Gomory's cut and branch-and-bound have been published by Balas et al. [2] and Achterberg et al. [1]. In addition, some cutting plane techniques other than Gomory's have achieved success by focusing on the generation of deep cutting planes (see for instance Boyd [3], Eckstein and Nediak [5] and Letchford [9]).

Observe that if the solution to ILP is close to an optimal solution to the associated linear programming relaxation problem, denoted by RILP, it will be a good idea letting the objective function varied parametrically and searching for the solution to ILP on the objective function hyperplane shifts. Based on this idea, various search algorithms have been presented (see for instance Thompson [10], Joseph et al. [7, 8], and Gao [6]). Of these, Thompson's stopped simplex method [10] deserves further attention due to its excellent computational characteristics. According to Thompson's statement, the stopped simplex method solves a great many of stopped linear programming problems to arrive at an answer with little computation, and has a moderate memory requirement that varies linearly with the size of the problem. Obviously, the computational efficiency of the stopped simplex method is primarily determined by the number of the stopped linear programming problems solved, that is, the stopped number used on the variables. Taking notice of that, in this paper we present an improvement of the stopped simplex algorithm by Thompson that aims to further decrease the stopped number of the variables. For this reason, two special cuts are generated by introducing a linear transformation to cut the intersection of the objective function hyperplane and the feasible region of the linear programming relaxation problem. So, the cuts lead to the more narrow intervals of the variables on the objective function hyperplane. Subsequently, the stopped simplex algorithm with the cuts is carried out to do a search on the objective function hyperplane. Finally, a test on some classical numerical examples is made. It shows that the algorithm presented here is more efficient and potential, compared with Thompson's algorithm.

The paper is organized as follows. In Section 2, the basic theory on searching for the solution on the objective function hyperplane is established. Section 3 presents an improved stopped simplex algorithm to do a search on the objective function hyperplane, and Section 4 describes the algorithm steps in detail. In Section 5, one numerical example is first given to illustrate the use of the algorithm and then the further computational study on some classical examples is made. Finally in Section 6 we make a brief conclusion about the algorithm.

2. Preliminaries

Consider a pure integer linear programming problem of the form

$$\begin{array}{lll} \text{(ILP)} & \max & c^T x \\ & \text{s. t.} & Ax \leq b \\ & & x \geq 0, \text{ and integral,} \end{array}$$

where $A = (a_{ij}) \in Z^{m \times n}$, and $b \in Z^m$, $c \in Z^n$.

Suppose that by applying the simplex method (see for instance Dantzig [4]), we obtain an optimal basic solution to RILP, $x_{B^*} = b^*$, $x_{N^*} = 0$ with the optimum value f^*, where x_{B^*} and x_{N^*} are the optimal basic and non-basic variables, respectively. Let $\bar{c}_{N^*}$ denote the reduced costs corresponding to the non-basic variables. Then the objective function and the constraints can be expressed as

$$f = f^* - \bar{c}_{N^*}^T x_{N^*}, \tag{2.1}$$

$$x_{B^*} = b^* - B^{*-1} N^* x_{N^*}, \tag{2.2}$$

$$x_{B^*} \geq 0,\ x_{N^*} \geq 0, \tag{2.3}$$

where B^* and N^* are the optimal basic and non-basic matrices, respectively. For convenience, assume $\bar{c}_{N^*} > 0$.

If the optimal basic solution to RILP is integral, it is also an optimal one to ILP. Otherwise, the optimum value for ILP is certainly smaller than f^*. In this case, let the objective function f as a parameter varied down beginning with f^*. The associated objective function hyperplane shift is represented by S_f. Obviously if any, a feasible solution to ILP constantly lies on the objective function hyperplane with an integral objective value. As soon as a feasible solution to ILP emerges on an objective function hyperplane shift S_f, the algorithm can be terminated according to the following optimality rule.

Theorem 2.1. *If there is a feasible solution* $x = \bar{x}$ *to ILP on an objective function hyperplane shift* S_f *with* $f = \bar{f}$ *and no feasible solution to ILP yields for any integral value of f with* $f > \bar{f}$, *then* $x = \bar{x}$ *is optimal for ILP with the objective value* $\bar{f}$.

In the next section we will use the stopped simplex algorithm to perform a search on the objective function hyperplane S_f.

3. The Stopped Simplex Algorithm with Special Cuts

According to Thompson's stopped simplex method [10], once a variable is assigned an integral value in a stopped search course, we call it "stopped" and otherwise, "unstopped" or "free". In what follows, S is used to represent the subscript set of the stopped variables and T, the subscript set of the unstopped ones. When a stopped course is changed, the value of the last stopped variable is increased by one and other stopped variables keep unchanged.

Observe that it is significant for improving the efficiency of the stopped simplex algorithm how to make them more narrow the intervals of the variables on the objective function hyperplane. For this reason, we introduce a linear transformation of the non-basic variables into a new variable by (2.1) as follows.

$$y = \sum_{j=1}^{n} d_j x_{N^*j} \tag{3.1}$$

with $d_j = \left[\dfrac{\bar{c}_{N^*j}}{\bar{c}^L_{N^*}}\right]$ for $j = 1, 2, \ldots, n,$ and $\bar{c}^L_{N^*} = \min_{1 \le j \le n} \{\bar{c}_{N^*j}\},$ where $[\bullet]$ stands for the greatest integer number smaller than or equal to $\bullet$. Obviously, all $d_j (j = 1, \ldots, n)$ are integer numbers greater than or equal to one. Therefore, the value of the variable y associated with a feasible solution to ILP is a nonnegative integer number. Furthermore, we have the following useful conclusion for the interval of the variable y.

Theorem 3.1. *Let* $\alpha_j = \dfrac{\bar{c}_{N^*j}}{d_j}$ *for* $j = 1, 2, \ldots, n$ *and* $\alpha^U = \max_{1 \le j \le n} \{\alpha_j\},$ $\alpha^L = \min_{1 \le j \le n} \{\alpha_j\}.$ *Then for a fixed integral value of f, the variable y by* (3.1) *with a feasible solution has an interval*

$$\frac{f^* - f}{\alpha^U} \le y \le \frac{f^* - f}{\alpha^L}. \tag{3.2}$$

Proof. Observe that (2.1) holds for a fixed integral value of f and a feasible solution on the objective function hyperplane S_f, that is,

$$\sum_{j=1}^{n} \alpha_j (d_j x_{N^*j}) = f^* - f.$$

Due to $\alpha^U = \max_{1 \le j \le n} \{\alpha_j\},$ $\alpha^L = \min_{1 \le j \le n} \{\alpha_j\},$ and $d_j x_{N^*j} \ge 0$ for all $j = 1, 2, \ldots, n,$ we have

$$\alpha^L \sum_{j=1}^{n} d_j x_{N^*j} \le \sum_{j=1}^{n} \alpha_j (d_j x_{N^*j}) \le \alpha^U \sum_{j=1}^{n} d_j x_{N^*j}$$

or

$$\alpha^L y \le f^* - f \le \alpha^U y.$$

The expressions above are equivalent to the inequalities (3.2). This completes the proof of the theorem.

Actually, the left and right inequalities of (3.2) are two special cuts with f as a parameter, which can be used to cut the intersection of the objective function hyperplane and the feasible region of RILP. Therefore, the cuts generated by the linear transformation (3.1) make them more narrow the intervals of the variables on the objective function hyperplane.

Now for a fixed integral value of f with $f < f^*$, let y^{IL} be the smallest integer number greater than or equal to $\frac{f^* - f}{\alpha^U}$, and y^{IU}, the greatest integer number smaller than or equal to $\frac{f^* - f}{\alpha^L}$. According to the above theorem, y^{IL} and y^{IU} are integral lower and upper bounds of the variable y with a feasible solution, respectively. If $y^{IL} > y^{IU}$, then no feasible solution exists on the associated objective function hyperplane in terms of the number theoretic properties. Otherwise, the stopped simplex algorithm is performed to do a search on the objective function hyperplane S_f below.

First, if x_{N^*j} for any $j \in \{1, ..., n\}$ is a slack variable of (ILP), it will be expressed as the linear function of the decision variables by the associated constraint of (ILP) and then substituted into (3.1). So, (3.2) is equivalently transformed into the equality constraint on the decision variables, represented by

$$y'^{IL} \leq \sum_{j=1}^{n} d'_j x_j \leq y'^{IU}.$$

Next, we carry out the stops on the decision variables x_j in the sequence of $j = 1, ..., n.$ Suppose that the variables $x_1, ..., x_{i-1}$ $(i \geq 1)$ are stopped in a stopped course, where $i = 1$ means that no variable is stopped. Taking the minimum value of x_i as the objective function, we construct the i-th stopped linear programming problem, labeled by (SLP-i), below.

$$\begin{aligned}
\min \quad & x_i \\
\text{s.t.} \quad & Ax \le b \\
& c^T x = f \\
& y'^{IL} \le \sum_{j=1}^{n} d'_j x_j \le y'^{IU} \\
& x_1, \ldots, x_{i-1} \quad \text{stopped} \\
& x_k \ge 0, \; k = i, \ldots, n.
\end{aligned}$$

The stopped problem above can be solved by the dual simplex algorithm. Obviously if any, the optimum value of (SLP-i) is a lower bound of x_i. Let x_i^{IL} be the smallest integral number greater than or equal to the lower bound of x_i. Then x_i^{IL} is an integral lower bound of x_i. In the forward search, stop the i-th variable x_i at x_i^{IL}. If (SLP-i) has no feasible solution, the forward stopped search will produce no feasible solution to ILP by the following theorem, in which a backtracking is performed to make x_{i-1} become free.

Theorem 3.2. *Suppose that for a given integral value of f, the variables*

$$(\,*\,)\; x_1, \ldots, x_{i-2} \; (i \ge 2)$$

are assigned fixed integral values, and correspondingly, $(SLP\text{-}(i-1))$ *has a feasible solution with the objective value* x_{i-1}^0. *If there is no optimal solution to the problem* (*SLP*-i) *with the stopped variables* $(\,*\,)$ *and* $x_{i-1} = x_{i-1}^1$, *where* x_{i-1}^1 *is an integer number greater than* x_{i-1}^0, *then the stopped variables* $(\,*\,)$ *and* x_{i-1} *with* $x_{i-1} > x_{i-1}^1$ *produce no feasible solution to ILP*.

Similar to the proof of Theorem 4 by Thompson [10], the proof is easily completed. Concretely, our stopped simplex search procedure on the objective function hyperplane S_f is devised as follows.

At the outset, set $S = \phi, T = \{1, \ldots, n\}$, and then compute the lower bound y^{IL} and upper bound y^{IU} of the variable y by the linear transformation (3.1). If $y^{IL} > y^{IU}$, set $f - 1$ to f and go back to the outset. Otherwise, find the lower bound x_1^{IL} of the variable x_1 by solving the stopped problem (SLP-1). If (SLP-1) has no feasible solution, set $f - 1$ to f and go back to the outset. If (SLP-1) has an optimal solution, stop the variable x_1 at its lower bound x_1^{IL}. Suppose that this forward search course goes on until the $(i - 1)$-th variable, $x_{i-1} (i \geq 2)$, is stopped. By now $S = \{1, \ldots, i - 1\}, T = \{i, \ldots, n\}$. Find x_i^{IL} by solving the i-th stopped problem (SLP-i). If (SLP-i) has no feasible solution, a backtracking will be started by setting $S = S \setminus \{i - 1\}$ and $T = T \cup \{i - 1\}$. If (SLP-i) has an optimum value, the next variable x_i is stopped at its lower bound x_i^{IL}. Continue in this way. When a backtracking arrives at the first stopped variable x_1 whose value is either beyond its upper bound or makes (SLP-2) not feasible, set $f - 1$ to f and go back to the outset. When the last variable x_n is stopped at its lower bound, the stopped simplex search procedure ends with a feasible solution on the objective function hyperplane S_f.

4. The Algorithm Steps

According to the algorithm theory above-mentioned, we can describe the computational steps of the stopped simplex algorithm in detail as follows.

Step 1. Solve RILP to get the optimal solution $x_{B^*} = b^*, x_{N^*} = 0$ with the optimum value f^*, and go to next step.

Step 2. If $x_{B^*} = b^*$ is integral, the algorithm is terminated. Otherwise, introduce the linear transformation (3.1), and compute the lower and upper bounds, y^{IL} and y^{IU}, of the variable y by (3.2) and then go to next step.

Step 3. Let f^{IU} be the greatest integer number smaller than f^* and M, a given adequate small integer number with $M < f^{IU}$, set $f = f^{IU}$ and go to next step.

Step 4. Check if $f < M$. If it is so, the algorithm terminates with no feasible solution. Otherwise, go to next step.

Step 5. Set $i = 0$, $s = (0, \ldots, 0) \in R^n$, $S = \phi$, $T = \{1, \ldots, n\}$, and then go to next step.

Step 6. If $y^{IL} > y^{IU}$ holds, set $f - 1$ to f and go back to Step 4. Otherwise, go to next step.

Step 7. If $i = n$, the algorithm terminates with the output of an optimal solution. Otherwise, go to next step.

Step 8. Find the lower bound x_{i+1}^{IL} of x_{i+1} by solving the stopped problem $(\text{SLP-}(i+1))$. If $(\text{SLP-}(i+1))$ has an optimum value, go to next step. Otherwise, that is, if $(\text{SLP-}(i+1))$ has no feasible solution, let $S = S \setminus \{i\}$, $T = T \cup \{i\}$, $i = i - 1$, and go to Step 10.

Step 9. Set $i + 1$ to i, and stop the variable x_i at its lower bound x_i^{IL}, and let $S = S \bigcup \{i\}$, $T = T \setminus \{i\}$, and go back to Step 7.

Step 10. See if $S = \phi$. If it is so, set $f - 1$ to f and go back to Step 4. Otherwise, set $x_i = x_i + 1$, and go back to Step 8.

By carrying out the algorithm steps above, we obtain either an optimal solution or the fact that there is no feasible solution to ILP.

5. A Numerical Example and Further Computational Study

First of all, we illustrate the use of our algorithm in detail with the following example.

Example 1. The problem considered is:

$$\begin{array}{lll} \text{(ILP)} & \max & -x_3 \\ & \text{s.t.} & -5x_1 - 8x_2 + 7x_3 \le 89 \\ & & 6x_1 - 5x_2 - x_3 \le -11 \\ & & -3x_1 + 5x_2 - 2x_3 \le -29 \\ & & x_1,\ x_2,\ x_3 \ge 0, \text{ and integral.} \end{array}$$

Introducing the slack variables $x_j \ge 0$ $(j = 4, 5, 6)$ and then solving the corresponding linear programming relaxation problem by the dual simplex method, we obtain

$$x_1 = 1.344 + 0.167x_4 + 0.211x_5 + 0.478x_6$$

$$x_2 = 0.878 + 0.167x_4 + 0.344x_5 + 0.411x_6$$

$$x_3 = 14.678 + 0.167x_4 + 0.544x_5 + 0.811x_6.$$

Obviously, the optimal solution to RILP, $x_1 = 1.344$, $x_2 = 0.878$, $x_3 = 14.678$, is non-integral. Therefore, we perform the stopped simplex search on the objective function hyperplane S_f below.

Due to $d_1 = 1$, $d_2 = \left[\frac{0.544}{0.167}\right] = 3$, $d_3 = \left[\frac{0.811}{0.167}\right] = 4$, we introduce a linear transformation

$$y = x_4 + 3x_5 + 4x_6 = -60 - x_1 + 3x_2 + 4x_3$$

and rewrite the objective function as

$$x_3 - 14.678 = 0.167x_4 + 0.181(3x_5) + 0.203(4x_6)$$

and therefore obtain two special cutting inequalities

$$\frac{x_3 - 14.678}{0.203} \le y \le \frac{x_3 - 14.678}{0.167}. \tag{5.1}$$

Subsequently, taking $x_3 = 15$, we have $1.586 \le y \le 1.928$ by (5.1). Thus, there is no feasible solution on the associated objective hyperplane with $f = -15$.

Again setting $x_3 + 1 = 16$ to x_3, we have $y^{IL} = y^{IU} = 7$. In this case, the two cutting inequalities are $67 \leq -x_1 + 3x_2 + 4x_3 \leq 67$ (see Fig. 1). Solving the stopped problem (SLP-1), we obtain the optimum value $x_1^* = 1.9565$ of x_1, and thus $x_1^{IL} = 2$. Letting $x_1 = 2$ and then solving (SLP-2), we have $x_2^* = 1.6633$. Stopping $x_2 = 2$ leads to no feasible solution, and next, a backtrack to $x_1 = 3$ leads to no feasible solution. Therefore it is concluded that no feasible solution to ILP exists on the associated objective function hyperplane.

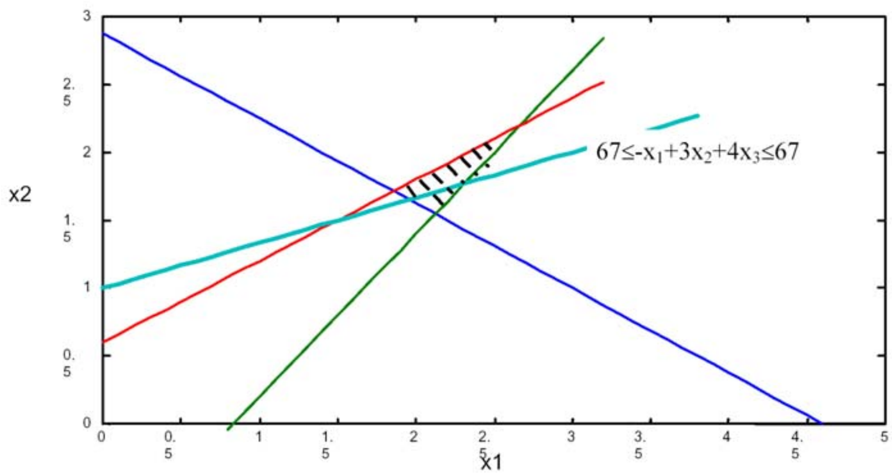

Figure 1. Two special cuts on the objective function hyperplane $x_3 = 16$.

Similarly, the algorithm also finds no feasible solution on the objective function hyperplane with $x_3 = 17$ after solving 3 stopped problems and carrying out 3 stops.

Finally, when x_3 is increased by 1 up to 18, it produces $x_1^* = 2.6327$ solving (SLP-1). Therefore, by stopping $x_1 = 3$ and solving (SLP-2), we obtain a solution to the problem below.

$$x_1 = 3,\ x_2 = 3,\ x_3 = 18.$$

This is a demonstrating example of Section 6 due to Thompson [10]. Our algorithm only makes 8 stops to obtain the answer to the problem. However, the problem was solved after making 12 stops by Thompson's method.

Our stopped simplex algorithm was programmed by MATLAB V6.5 and conducted on a HΛSEE S262C to solve the classical examples given by Thompson [10] in Section 7. A comparison between our improved stopped simplex algorithm and the stopped simplex algorithm by Thompson is performed. The results on the number of the stopped problems solved (labeled by problems), the number of the stops needed (labeled by stops), and the number of the pivots needed (labeled by pivots) are shown in Table 1.

Table 1. A comparison between our algorithm and Thompson's algorithm

Example	Our algorithm			Thompson's algorithm		
No.	problems	stops	pivots	problems	stops	pivots
1	2	2	5	17	18	∗
2	3	3	7	> 17	> 18	∗
3	63	37	298	173	∗	194
4	31	30	179	∗	∗	∗
5	125	26	424	255	∗	102
6	455	227	1138	260	∗	105
7	434	210	1384	255	∗	103
8	8	8	43	13	∗	84
9	239	232	2222	2038	∗	10875

Note. ∗ indicates that the number for the problem was not reported in the literature.

It is seen from Table 1 that our algorithm greatly improves Thompson's stopped simplex algorithm except Examples 5, 6 and 7.

However, the problem like Examples 5, 6 and 7 has a long narrow feasible region in geometry. In this case, the cuts generated in our algorithm always do not cut through the intersection of the objective function hyperplane and the feasible region of RILP, and therefore are meaningless for our algorithm. It should be pointed out here that most of applications are not the case.

Limited to the low performance of the computer, we do not make a numerical test on large-scaled problems, such as ones from MIPLIB, ORlibrary. This work, together with analysis of the algorithm's complexity, will be done later.

6. Concluding Remarks

An important feature in the algorithm is that the two special cuts are generated by introducing a linear transformation. The cuts make them more narrow the intervals of the variables on the objective function hyperplane, and therefore greatly improve the stopped simplex algorithm by Thompson [10]. It was proved by the numerical test in Section 5. Since the stopped simplex algorithm has the high performance with little computation and moderate memory requirements from Thompson’s statement, our algorithm is of more practical interests.

Just as we see in Section 5, the cuts do not improve the efficiency of the stopped simplex algorithm on Examples 5, 6, and 7. This is because the cuts do not cut through the intersection of the objective function hyperplane and the feasible region of the problem like those examples. In this case, we call the cuts “non-effective”. Thus, it becomes a topic how to construct “effective” cuts for the problem with a long narrow feasible region.

In addition, when the problem has the solution far from the optimal solution to RILP, shown as Examples 5, 6, and 7, it takes the stopped simplex algorithm considerable computation due to the objective function hyperplane shifts. Thus, a better improvement is worth doing for such a problem.

References

[1] T. Achterberg, T. Koch and A. Martin, Branching rules revisited, Operations Research Letters 33 (2005), 42-54.

[2] E. Balas, S. Ceria, G. Cornuejols and N. Natraj, Gomory cuts revisited, Operations Research Letters 19 (1996), 1-9.

[3] E. A. Boyd, Fenchel cutting planes for integer programs, Operations Research 42 (1994), 53-64.

[4] G. B. Dantzig, Linear Programming and Extensions, Princeton University Press, 1963.

[5] J. Eckstein and M. Nediak, Depth-optimized convexity cuts, Annals of Operations Research 139 (2005), 95-129.

[6] P. W. Gao, An efficient bound-and-stopped algorithm for integer linear programs on the objective function hyperplane, Applied Mathematics and Computation 185 (2007), 301-311.

[7] A. Joseph, S. I. Gass and N. A. Bryson, A computational study of an objective hyperplane search heuristic for the general integer linear programming problem. Mathematical and Computer Modeling 25 (10) (1997), 63-76.

[8] A. Joseph, S. I. Gass and N. A. Bryson, An objective hyperplane search procedure for solving the general all-integer linear programming problem, European Journal of Operational Research 104 (1998), 601-614.

[9] A. N. Letchford, Binary clutter inequalities for integer programs, Mathematical Programming (Ser.B) 98 (2003), 201-221.

[10] G. L. Thompson, The stopped simplex method: Basic theory for mixed integer programming; integer programming, Revue Francaise de Recherche Operationnelle 8 (1964), 159-182.

■