Far East Journal of Applied Mathematics
Volume 35, Number 1, 2009, Pages 33-47
Published Online: April 24, 2009
This paper is available online at http://www.pphmj.com

A NEW IMPLICIT ENUMERATIVE SEARCH FOR THE SOLUTION TO AN INTEGER LINEAR PROGRAM

PEI-WANG GAO

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

Abstract

This paper presents a new implicit enumerative algorithm for a pure integer linear programming problem. In the algorithm, the objective function as a parameter is decreased from the optimum value of the associated linear programming relaxation problem, so that the objective function hyperplane intersects the binding constraints cone to form a simplex. So, the simplex is used to determine the intervals of the variables. If there is no integer number in the interval of any variable, the associated objective function hyperplane does not contain any feasible integer solution and then is shifted down by one unit. Otherwise, an implicit enumerative procedure is carried out to do a search in the simplex. In the implicit enumerative search process, the upper and lower bounds of the unstopped variables are improved step-by-step based on the original constraints and objective function, and thus the computational efficiency is greatly raised. Finally, the computational test on some classical and randomly-generated numerical examples is made. It shows that the algorithm presented here is convenient, efficient and potential.

2000 Mathematics Subject Classification: 90C10.

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

This research was supported by the Guangxi Scientific Fund Grant number GKZ0728260 and the Scientific Fund of GUFE Grant number 2006JYB001.

Received November 12, 2008

1. Introduction

Since the cutting plane method and the branch-and-bound principle as two classical techniques are developed for solving integer linear programming problems (ILP in short), a considerable amount of effort in trying to improve the performance of the methods has been expended by various researchers (see, for instance, Achterberg et al. [1], Balas et al. [2], Ceria et al. [4], Elhedhli and Goffin [5], and Mitchell [10]). Nevertheless, it still spends a considerable amount of computation to solve the linear programming subproblems generated by cuts or branches.

Observe that in many cases, the solution to ILP is close to an optimal solution to the associated linear programming relaxation problem, denoted by RILP (see, for instance, Baum and Trotter [3]). People presented an idea of letting the objective function varied parametrically and searching for the solution to ILP on the objective function hyperplane shifts (see, for instance, Land and Doig [9], Thompson [11], Joseph et al. [7, 8] and Gao [6]). Along this line, this paper presents a new implicit enumerative algorithm for a pure integer linear programming problem. In the algorithm, the objective function as a parameter is decreased from the optimum value of RILP so that the objective function hyperplane intersects the binding constraints cone of RILP to form a simplex. So, the simplex is used to determine the intervals of the variables. If there is no integer number in the interval of any variable, the associated objective function hyperplane does not contain any feasible integer solution and then is shifted down by one unit. Otherwise, an implicit enumerative procedure is carried out to do a search in the simplex. Compared with the bound-and-stopped algorithm by Gao [6], the intervals of the variables in the simplex are easily found with no pivot but may become wider. For this reason, further improvement on the upper and lower bounds of the unstopped variables is made at each iteration based on the original constraints and objective function. It not only raises the efficiency of the implicit enumeration but also avoids an increasing memory requirement with problem size to do so. Finally, the computational test on some classical and randomly-generated numerical examples is made. It shows that the algorithm presented here is convenient, efficient and potential.

The paper is organized as follows: In Section 2, the search region on the objective function hyperplane is established. Section 3 presents an implicit enumerative algorithm to do a search in the region. In Section 4, one numerical example is first given to illustrate the use of the algorithm and then the further computational test on some classical and randomly-generated problems is made. Finally, in Section 5, we make a brief conclusion about the algorithm.

2. The Determination of the Initial Bounds of the Variables

Consider an integer linear programming problem of the form

$$\text{(ILP)} \qquad \max\ c^T x$$

$$\text{s.t. } Ax \le b$$

$$x \ge 0, \text{ and integral,}$$

where $A = (a_{ij})$ is an $m \times n$ integer matrix and b, c are integer column vectors of appropriate dimensions.

Suppose that there is 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 basic variables and the non-basic ones, respectively. Then the objective function and the constraints can be expressed as

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

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

where B^* and N^* are the optimal basic matrix and the non-basic matrix respectively, and $\bar{c}_{N^*}$, the reduced costs corresponding to the non-basic variables. For convenience, assume $\bar{c}_{N^*} > 0$.

If the optimal basic solution to RILP is integral, it is also optimal for ILP. Otherwise, the optimum value for ILP is certainly smaller than f^*, and an implicit enumerative algorithm will be presented to solve ILP. In

this case, let the objective function f be a parameter varied down beginning with the optimum value. The associated objective function hyperplane would intersect the binding constraints cone, represented by $x_{N^*} \geq 0$, to form an $(n-1)$-simplex, denoted by S_f. Obviously if any, a feasible solution to ILP constantly lies in the simplex associated with the integral values of the objective function. Based on the fact, our implicit enumerative search will be carried out in the simplex.

Now, in (2.1) and (2.2), fixing an index $j \in \{1, ..., n\}$ and letting $x_{N^*i} = 0$ for $i = 1, ..., n$, $i \neq j$, we obtain the components of the jth vertex $x^j(f)$, labelled by $x^j(f) = (x_1^j(f), ..., x_{n+m}^j(f))^T$ $(j = 1, ..., n)$, as follows:

$$x^j_{N^*j}(f) = \frac{1}{\bar{c}_{N^*j}}(f^* - f), \quad j = 1, ..., n,$$

$$x^j_{N^*i}(f) = 0, \quad i = 1, ..., n, \text{ but } i \neq j,$$

$$x^j_{B^*k}(f) = b_k^* + d_{kj}^* x^j_{N^*j}(f), \quad k = 1, ..., m,$$

where d_{kj}^* is the element of the matrix $-B^{*-1}N^*$ on row k and column j. Letting $\alpha_{N^*j} = -1/\bar{c}_{N^*j}$, $\beta_{N^*j} = f^*/\bar{c}_{N^*j}$ and substituting the expression of $x^j_{N^*j}(f)$ into that of $x^j_{B^*k}(f)$, we have

$$x^j_{N^*j}(f) = \alpha_{N^*j} f + \beta_{N^*j}, \quad j = 1, ..., n, \tag{2.3}$$

$$x^j_{B^*k}(f) = \alpha_{B^*k, N^*j} f + \beta_{B^*k, N^*j}, \quad k = 1, ..., m, \tag{2.4}$$

where $\alpha_{B^*k, N^*j} = d_{kj}^* \alpha_{N^*j}$, $\beta_{B^*k, N^*j} = b_k^* + d_{kj}^* \beta_{N^*j}$.

In terms of reference 6, $\min_{1 \leq j \leq n}(x_i^j(f))$ and $\max_{1 \leq j \leq n}(x_i^j(f))$ are lower and upper bounds of the variable x_i $(i = 1, ..., n + m)$ in the simplex S_f, respectively. Thus, by using (2.3) and (2.4) to compute the components of the vertices in the simplex S_f, we can obtain initial lower

and upper bounds of the variables. Furthermore, observe that a feasible solution to ILP is non-negative and integral. Let $x_i^{IL}(f)$ be the smallest integer number greater than or equal to $\min_{1\le j\le n}(x_i^j(f))$, and $x_i^{IU}(f)$ be the greatest integer number smaller than or equal to $\max_{1\le j\le n}(x_i^j(f))$, for any $i = 1, ..., n$. Then $x_i^{IL}(f)$ and $x_i^{IU}(f)$ are initial integral lower and upper bounds of the variable x_i $(i = 1, ..., n + m)$.

Next, let f take the integral values with $f < f^*$ from large to small in turn. In this way, as soon as a feasible solution to ILP is found in a simplex S_f, the algorithm terminates according to the following optimality rule.

Theorem 2.1. *If there is a feasible solution $x = \bar{x}$ to ILP in a simplex 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.*

The theorem above-mentioned is obviously satisfied. The new implicit enumerative search algorithm will be presented in the next section.

3. The Implicit Enumerative Algorithm

According to the theory above, given a fixed integral value of f with $f < f^*$. If there is no integer number in the interval $[x_i^{IL}(f), x_i^{IU}(f)]$ for any $i \in \{1, ..., n + m\}$, no feasible solution to ILP exists on the associated simplex S_f and the objective function hyperplane will be shifted down by one unit. Otherwise, a new implicit enumerative search procedure is carried out below.

First of all, let $a_{ij}^+ = \max(a_{ij}, 0)$, $a_{ij}^- = \min(a_{ij}, 0)$, $c_j^+ = \max(c_j, 0)$, and $c_j^- = \min(c_j, 0)$ for $i = 1, ..., m$, $j = 1, ..., n$. ILP can be rewritten as

$$\max f = \sum_{j=1}^{n} (c_j^+ + c_j^-)x_j \tag{3.1}$$

$$\text{s.t.} \sum_{j=1}^{n} (a_{ij}^{+} + a_{ij}^{-}) x_j \leq b_i, \quad i = 1, \ldots, m \tag{3.2}$$

$$x_j \geq 0, \text{ and integral } j = 1, \ldots, n. \tag{3.3}$$

Obviously, $c_j^{+} \geq 0$, $c_j^{-} \leq 0$, $a_{ij}^{+} \geq 0$ and $a_{ij}^{-} \leq 0$ for $i = 1, \ldots, m$, $j = 1, \ldots, n$.

Next, our implicit enumerative search procedure also uses the stopped rule to assign the values of the variables as in the stopped simplex algorithm by Thompson [11], but differently, needs no simplex pivot to determine the bounds. In the stopped process, once a variable is assigned an integral value, we call it *stopped* and otherwise, *unstopped*. In what follows, S is used to represent the subscript set of the stopped variables and T, the subscript set of the unstopped ones. After some variables are stopped in a stopped course, the bounds of all the unstopped variables may be improved by the following theorems.

Theorem 3.1. *Suppose that in a stopped course, the variable x_s for any $s \in S$ is assigned a fixed integral value x_s^0, and the variable x_t for any $t \in T$ has a lower bound $x_t^{IL}(f)$ and an upper bound $x_t^{IU}(f)$. Then the value of an unstopped variable x_t $(t \in T)$ in a feasible solution in the simplex S_f is such that*

$$x_t \leq \min_{1 \leq i \leq m} \left\{ \frac{b_i - \sum_{s \in S} a_{is} x_s^0 - \sum_{j \in T} a_{ij}^{-} x_j^{IU}(f) - \sum_{j \in T \setminus \{t\}} a_{ij}^{+} x_j^{IL}(f)}{a_{it}^{+}} \mid a_{it}^{+} > 0 \right\} \tag{3.4}$$

and

$$x_t \geq \max_{1 \leq i \leq m} \left\{ \frac{b_i - \sum_{s \in S} a_{is} x_s^0 - \sum_{j \in T} a_{ij}^{+} x_j^{IL}(f) - \sum_{j \in T \setminus \{t\}} a_{ij}^{-} x_j^{IU}(f)}{a_{it}^{+}} \mid a_{it}^{-} < 0 \right\}. \tag{3.5}$$

Proof. Note that the values of the unstopped variables in a feasible

solution to ILP in the simplex S_f satisfy

$$x_j^{IL}(f) \le x_j \le x_j^{IU}(f), \quad j \in T.$$

For any $t \in T$, we have by the constraints (3.2),

$$\sum_{s\in S} a_{is}x_s^0 + \sum_{j\in T} a_{ij}^- x_j^{IU}(f) + \sum_{j\in T\setminus\{t\}} a_{ij}^+ x_j^{IL}(f) + a_{it}^+ x_t \le b_i, \quad i = 1, \ldots, m$$

and

$$\sum_{s\in S} a_{is}x_s^0 + \sum_{j\in T} a_{ij}^+ x_j^{IL}(f) + \sum_{j\in T\setminus\{t\}} a_{ij}^- x_j^{IU}(f) + a_{it}^- x_t \le b_i, \quad i = 1, \ldots, m.$$

Therefore, we obtain for $a_{it}^+ > 0$,

$$x_t \le \frac{b_i - \sum_{s\in S} a_{is}x_s^0 - \sum_{j\in T} a_{ij}^- x_j^{IU}(f) - \sum_{j\in T\setminus\{t\}} a_{ij}^+ x_j^{IL}(f)}{a_{it}^+},$$

$$i = 1, \ldots, m$$

and for $a_{it}^- < 0$,

$$x_t \ge \frac{b_i - \sum_{s\in S} a_{is}x_s^0 - \sum_{j\in T} a_{ij}^+ x_j^{IL}(f) - \sum_{j\in T\setminus\{t\}} a_{ij}^- x_j^{IU}(f)}{a_{it}^-},$$

$$i = 1, \ldots, m.$$

Summing up the $2m$ inequalities above from $i = 1$ to m, respectively, we lead to the inequalities (3.4) and (3.5), and therefore, complete the proof of the theorem.

Note that a feasible solution in the simplex S_f satisfies (3.1). Similar to the proof of Theorem 3.1, we also have

Theorem 3.2. *Suppose that in a stopped course, the variable x_s for any $s \in S$ is assigned a fixed integral value x_s^0, and the variable x_t for any $t \in T$ has a lower bound $x_t^{IL}(f)$* and an upper bound $x_t^{IU}(f)$. *Then the value of an unstopped variable x_t $(t \in T)$ in a feasible solution in the*

simplex S_f *is such that for* $c_t^+ > 0$,

$$\frac{f - \sum_{s\in S} c_s x_s^0 - \sum_{j\in T} c_j^- x_j^{IL}(f) - \sum_{j\in T\setminus\{t\}} c_j^+ x_j^{IU}(f)}{c_t^+} \le x_t$$

$$\le \frac{f - \sum_{s\in S} c_s x_s^0 - \sum_{j\in T} c_j^- x_j^{IU}(f) - \sum_{j\in T\setminus\{t\}} c_j^+ x_j^{IL}(f)}{c_t^+} \tag{3.6}$$

and for $c_t^- < 0$,

$$\frac{f - \sum_{s\in S} c_s x_s^0 - \sum_{j\in T} c_j^+ x_j^{IL}(f) - \sum_{j\in T\setminus\{t\}} c_j^- x_j^{IU}(f)}{c_t^-} \le x_t$$

$$\le \frac{f - \sum_{s\in S} c_s x_s^0 - \sum_{j\in T} c_j^+ x_j^{IU}(f) - \sum_{j\in T\setminus\{t\}} c_j^- x_j^{IL}(f)}{c_t^-}. \tag{3.7}$$

According to Theorems 3.1 and 3.2, the new integral lower bound, denoted by $x_t^{NIL}(f)$, and upper bound, denoted by $x_t^{NIU}(f)$, of an unstopped variable x_t $(t \in T)$ in a feasible solution can be generated by (3.4), (3.5), (3.6) and (3.7), and the initial bounds $x_t^{IL}(f)$, $x_t^{IU}(f)$. If $x_t^{NIU}(f) < x_t^{NIL}(f)$ for any $t \in T$ holds in a stopped course, the stopped course should be changed. Otherwise, the forward search will go on. Concretely, our implicit enumerative search procedure in the simplex S_f is devised as follows.

At the outset, see if $x_i^{IL}(f) > x_i^{IU}(f)$ holds for any $i \in \{1, ..., n + m\}$. If it is so, set $f - 1$ to f and repeat the process above. Otherwise, set $S = \varnothing$, $T = \{1, 2, ..., n\}$ and find the new bounds $x_j^{NIL}(f)$ and $x_j^{NIU}(f)$ of the variables x_j for all $j \in T$ by the iterative computation of (3.4), (3.5), (3.6), and (3.7). If $x_j^{NIL}(f) > x_j^{NIU}(f)$ for any $j \in T$, set $f = f - 1$ and repeat the above process. Otherwise, pick an index $s1$ with $s1 =$

$\arg\min_{t\in T}\{x_t^{NIU}(f) - x_t^{NIL}(f)\}$, and then stop the variable x_{s1} at its lower bound $x_{s1}^{NIL}(f)$. Suppose that, this stopped course goes on until the ith variable, say x_{si} $(i \geq 1)$, is stopped. Then let $S = S \cup \{si\}$, $T = T \backslash \{si\}$ and find the new bounds $x_j^{NIL}(f)$ and $x_j^{NIU}(f)(j \in T)$ of the variables x_j for all $j \in T$ by the iterative computation. Check if $x_j^{NIL}(f) > x_j^{NIU}(f)$ holds for any $j \in T$. If it is so and $x_{si} < x_{si}^{NIU}(f)$, the current stopped variable x_{si} is increased by one and then the bounds of the unstopped variables are again computed. If it is so and $x_{si} = x_{si}^{NIU}(f)$, then x_{si} is unstopped (correspondingly, $S = S \backslash \{si\}$, $T = T \cup \{si\}$), and the value of the preceding stopped variable $x_{s,i-1}$ increased by one. If it is not so, the next stopped variable $x_{s,i+1}$, with $s, i+1 = \arg\min_{t\in T}\{x_t^{NIU}(f) - x_t^{NIL}(f)\}$ is picked. Continues in this way. When the first stopped variable x_{s1} is valued beyond its upper bound, set $f-1$ to f and repeat the above process. And when the last variable x_{sn} is stopped at its lower bound, a feasible solution to ILP is achieved in the simplex S_f.

Detailed computational steps of the implicit enumerative search algorithm can be described as follows.

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

Step 2. If x^* is integral, then the algorithm is terminated. Otherwise, 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 3. Check if $f < M$. If it is so, then the algorithm terminates with no feasible solution. Otherwise, compute $x_i^{IL}(f)$ and $x_i^{IU}(f)$ for $i = 1, \ldots, n+m$ by (2.6) and (2.7), and then go to next step.

Step 4. If $x^{IL}_{B^*i}(f) > x^{IU}_{B^*i}(f)$ holds for any $i \in \{1, ..., n+m\}$, set $f-1$ to f and go back to Step 3. Otherwise, go to next step.

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

Step 6. If $k = n$, the algorithm terminates with the output of an optimal solution. Otherwise, go to next step.

Step 7. Compute the bounds $x_t^{NIL}(f)$ and $x_t^{NIU}(f)$ of x_t for all $t \in T$. If $x_t^{NIU}(f) < x_t^{NIL}(f)$ holds for any $t \in T$, go to Step 9. Otherwise, go to next step.

Step 8. If $x_t^{NIL}(f) = x_t^{IL}(f)$ and $x_t^{NIU}(f) = x_t^{IU}(f)$ for all $t \in T$, go to Step 11. Otherwise, set $x_t^{IL}(f) = x_t^{NIL}(f)$ and $x_t^{IU}(f) = x_t^{NIU}(f)$ for all $t \in T$, and go back to Step 7.

Step 9. See if $S = \varnothing$. If it is so, set $f-1$ to f and go back to Step 3. Otherwise, go to next step.

Step 10. Check if $x_{s(k)} < x^{IU}_{s(k)}(f)$. If it is so, set $x_{s(k)} = x_{s(k)} + 1$, and go back to Step 7. Otherwise, let $S = S \backslash \{s(k)\}$, $T = T \cup \{s(k)\}$, $k = k-1$, and go back to Step 9.

Step 11. Set $k+1$ to k, and pick an index $s(k)$ with $s(k) = \arg\min_{t \in T}\{x_t^{IU}(f) - x_t^{IL}(f)\}$ as the subscript of next stopped variable. Stop the variable $x_{s(k)}$ at its lower bound $x^{IL}_{s(k)}(f)$, and let $S = S \cup \{s(k)\}$, $T = T \backslash \{s(k)\}$, and go back to Step 6.

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

4. A Numerical Example and Further Computational Study

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

Example 4.1. The problem considered is

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

Solving the corresponding linear programming relaxation problem by the dual simplex method, we obtain

$$\begin{aligned}
x_1 &= 1.344 + 0.167x_4 + 0.211x_5 + 0.478x_6, \\
x_2 &= 0.878 + 0.167x_4 + 0.344x_5 + 0.411x_6, \\
x_3 &= 14.678 + 0.167x_4 + 0.544x_5 + 0.811x_6.
\end{aligned}$$

Obviously, the optimal solution to RILP, $x_1 = 1.344$, $x_2 = 0.878$, $x_3 = 14.678$ is non-integral. Therefore, for a fixed integral value of x_3 with $x_3 \ge 14.678$, we can find three extreme vertices of the simplex $S_f = \{x \mid x_4 \ge 0,\ x_5 \ge 0,\ x_6 \ge 0 \text{ and } f = -x_3\}$, represented as

$$x^1(f) = (x_3 - 13.334,\ x_3 - 13.800,\ x_3,\ 5.988x_3 - 87.892,\ 0,\ 0)^T,$$

$$x^2(f) = (0.388x_3 - 4.349,\ 0.632x_3 - 8.404,\ x_3,\ 0,\ 1.838x_3 - 26.982,\ 0)^T,$$

$$x^3(f) = (0.589x_3 - 7.307,\ 0.507x_3 - 6.561,\ x_3,\ 0,\ 0,\ 1.233x_3 - 18.099)^T,$$

and determine the initial bounds of the variables in a feasible solution below

$$x_1^{IL}(f) = \langle \max(0,\ 0.388x_3 - 4.349) \rangle, \quad x_1^{IU}(f) = [x_3 - 13.334],$$

$$x_2^{IL}(f) = \langle \max(0,\ 0.507x_3 - 6.561) \rangle, \quad x_2^{IU}(f) = [x_3 - 13.800],$$

where $\langle \bullet \rangle$ stands for the smallest integral number greater than or equal to $\bullet$, and $[\bullet]$, the greatest integral number smaller than or equal to $\bullet$.

Subsequently, we perform the implicit enumerative search in the simplex S_f. Taking $f = -15$ produces $x_1^{IL}(f) = 2 > x_1^{IU}(f) = 1$. Thus no feasible solution to ILP exists in the associated simplex with $f = -15$.

Setting $f - 1 = -16$ to f, we have $x_1^{IL}(f) = x_1^{IU}(f) = 2$, $x_2^{IL}(f) = x_2^{IU}(f) = 2$. In this case, by the iterative computation, we find the new bounds such that $x_1^{NIL}(f) = 3 > x_1^{NIU}(f) = 2$, $x_2^{NIL}(f) = 2 > x_2^{NIU}(f) = 1$, and therefore conclude that no feasible solution to ILP exists in the associated simplex.

Setting $f - 1 = -17$ to f. Correspondingly, $x_1^{IL}(f) = x_1^{IU}(f) = 3$, $x_2^{IL}(f) = x_2^{IU}(f) = 3$. Computing the new bounds, we have $x_1^{NIL}(f) = 4$, $x_1^{NIU}(f) = 3$, $x_2^{NIL}(f) = 3$ and $x_2^{NIU}(f) = 2$, and therefore conclude that no feasible solution to ILP exists in the associated simplex due to either $x_1^{NIL}(f) > x_1^{NIU}(f)$ or $x_2^{NIL}(f) > x_2^{NIU}(f)$.

Finally, f is decreased by 1 up to -18, we have $x_1^{IL}(f) = 3 \le x_1^{IU}(f) = 4$, $x_2^{IL}(f) = 3 \le x_2^{IU}(f) = 4$. By the iterative computation, the new bounds are $x_1^{NIL}(f) = 3$, $x_1^{NIU}(f) = 4$, $x_2^{NIL}(f) = 3$ and $x_2^{NIU}(f) = 3$. Therefore, pick $s(1) = \arg\min_{t \in T}\{x_t^{NIU}(f) - x_t^{NIL}(f)\} = 2$ as the subscript of the coming stopped variable and stop x_2 at $x_2^{NIL}(f) = 3$. Again computing the new bounds of the unstopped variables, we have $x_1^{NIL}(f) = 3$, $x_1^{NIU}(f) = 3$. Up to here, stopping x_1 at $x_1^{NIL}(f) = 3$, and setting $x_3 = -f$, we obtain an optimal solution to the original problem

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

This is a demonstrating example of Section 6 due to Thompson [11]. Our algorithm only makes 4 tests on the feasibility in the simplex and 2 stops to solve the problem.

Our implicit enumerative algorithm was programmed by MATLAB V6.5 and conducted on a HΛSEE S262C to solve the following examples.

First of all, the classical examples by Thompson were solved for comparison between our implicit enumerative algorithm (IE algorithm for short) and the classical branch-and-bound algorithm (BB algorithm for short). Table 1 gives the computational results, including the number of the objective function hyperplane shifts (labelled by SHTnum.), the number of the stops used (labelled by Snum.) and the executive time spent (labelled by Time) in the implicit enumerative algorithm, and the number of the linear programming subproblems branched (labelled by LPnum.), the number of the pivots needed (labelled by ITERnum.) and the executive time spent.

Table 1. Comparison between the IE algorithm and the BB algorithm

	The IE algorithm			The BB algorithm		
No.	SHTnum.	Snum.	Time (Sec.)	LPnum.	PVnum.	Time (Sec.)
1	19	1	0.078	35	120	0.359
2	19	2	0.109	37	159	0.468
3	27	2	0.156	111	604	1.843
4	2	9	0.469	47	463	1.375
5	99	2	0.984	201	600	2.485
6	552	2	1.485	203	708	2.797
7	548	2	1.469	203	708	2.797
8	1	8	0.297	125	1088	3.938
9	8	1168	68.89	*	*	*

Note: * indicates that the optimal solution is not found after 2000 subproblems are solved.

It is seen from Table 1 that our algorithm spends less executive time than the pure branch-and-bound algorithm, and improves the computational stability in that the number of the stops used in our algorithm is also fewer than the number of the linear programming subproblems solved in the pure branch-and-bound algorithm.

Next, we made a further numerical test on the 7 randomly-generated problems. Although, they may be of tiny-size, it is enough to indicate how the computational efficiency of the algorithm is affected. When tabulating the computational results (shown in Table 2), we list the number of the shifts of the objective function hyperplane, the number of

the stops used, and CPU times spent by the simplex method to solve RILP (labelled by RILP) and the implicit enumerative algorithm to find a solution to ILP (labelled by IE), respectively.

Table 2. Computational results for the randomly-generated problems

	Size				Execution times (Sec.)	
No.	m	n	SHTnum.	Snum.	RILP	IE
1	5	5	1	5	0.015	0.094
2	10	5	2	5	0.031	0.234
3	5	10	114	34	0.046	2.563
4	10	10	3	53	0.016	2.843
5	15	10	1	40	0.031	4.626
6	10	15	1	15	0.015	1.470
7	20	20	2	511	0.047	70.968

From Table 2, it is found that the execution time is primarily determined by the number of the shifts of the objective function hyperplane and the number of the stops used in the implicit enumerative search process, no matter how big the size of the problem is.

5. Concluding Remarks

One of most attractive features in the algorithm is that the bounds of the unstopped variables are improved step-by-step based on the original constraints in the implicit enumerative process. So, the algorithm can use the fewer stops to find the answer to a problem. Although, the objective function hyperplane is shifted down by one unit in the implicit enumerative search process, the branching in the branch-and-bound algorithm often leads to a slower decrease of the objective value in most cases. Thus, our algorithm generally spends less time than the branch-and-bound algorithm.

In addition, we can combine cuts and branches with the implicit enumeration to improve the bounds of the variables and decrease the number of the stops used in the implicit enumerative search process. Further research would be done thereafter.

References

[1] T. Achterberg, T. Koch and A. Martin, Branching rules revisited, Oper. Res. Lett. 33 (2005), 42-54.

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

[3] S. Baum and L. E. Trotter, Integer rounding for polymatroid and branching optimization problems, SIAM J. Algebraic and Discrete Methods 2 (1981), 241-245.

[4] S. Ceria, C. Cordier, H. Marchand and L. A. Wolsey, Cutting planes for integer programs with general integer variables, Math. Program. 81 (1998), 201-214.

[5] S. Elhedhli and J. L. Goffin, The integration of an interior-point cutting plane method with a branch-and-price algorithm, Math. Program. 100(2) (2004), 267-294.

[6] P. Gao, An efficient bound-and-stopped algorithm for integer linear programs on the objective function hyperplane, Appl. Math. Comput. 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, Math. Comp. Model. 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 J. Oper. Res. 104 (1998), 601-614.

[9] A. H. Land and A. G. Doig, An automatic method of solving discrete programming problems, Econometrica 28 (1960), 497-520.

[10] J. E. Mitchell, Fixing variables and generating classical cutting planes when using an interior point branch and cut method to solve integer programming problems, European J. Oper. Res. 97 (1997), 139-148.

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