A BOUNDING TECHNIQUE FOR INTEGER LINEAR PROGRAMMING
WITH BINARY VARIABLES

BY

FREDERICK S. HILLIER and NANCY EILEEN JACQMIN

TECHNICAL REPORT NO. 86
JUNE 1979

PREPARED UNDER CONTRACT
N00014-76-C-0418  (NR-047-061)
FOR THE OFFICE OF NAVAL RESEARCH

Frederick S. Hillier, Project Director

Reproduction in Whole or in Part is Permitted
for any Purpose of the United States Government
This document has been approved for public release
and sale; its distribution is unlimited.

DEPARTMENT OF OPERATIONS RESEARCH
STANFORD UNIVERSITY
STANFORD, CALIFORNIA

79 10 26 048
A BOUNDING TECHNIQUE FOR INTEGER LINEAR PROGRAMMING
WITH BINARY VARIABLES:

BY
FREDERICK S. HILLIER and NANCY EILEEN JACQMIN

TECHNICAL REPORT NO. 86
JUNE 1979

PREPARED UNDER CONTRACT
N00014-76-C-0418 (NR-047-061)
FOR THE OFFICE OF NAVAL RESEARCH
Frederick S. Hillier, Project Director

Reproduction in Whole or in Part is Permitted
for any purpose of the United States Government

This document has been approved for public release
and sale; its distribution is unlimited.

This research was supported in part by National Science Foundation
Grant NSF/SOL No. MCS 76-81259 A01, Dept. of Operations Research
Stanford University.
1. Introduction

This paper presents a bounding technique for use in implicit enumeration
algorithms for solving the integer linear programming problem with binary
variables. This problem may be stated as follows,

\[ \text{maximize } x_0 = \sum_{j=1}^{n} c_j x_j, \]

subject to

\[ \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad (i = 1, 2, \ldots, m) \]
\[ x_j = 0 \text{ or } 1, \quad (j = 1, 2, \ldots, n). \]

The main assumptions used by this technique are (1) the linear programming
relaxation of (P),

\[ \text{maximize } x_0 = \sum_{j=1}^{n} c_j x_j, \]

subject to

\[ \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad (i = 1, 2, \ldots, m) \]
\[ 0 \leq x_j \leq 1, \quad (j = 1, 2, \ldots, n) \]

possesses a unique optimal solution, (2) the optimal solution to \((P_R)\) is not
binary, and (3) a good feasible solution to \((P)\) is available. If (2) does not
hold then the optimal solution to \((P_R)\) is also the optimal solution to \((P)\)
and the algorithm would not be required. Assumption (3) may be met by
using one of the many heuristic procedures in the literature [1,4,6,7]. An
alternative to assumption (3) requires that a lower bound on the optimal
objective function value of a related problem \((P')\), defined in Section 3, be
available.

In what follows we show that "joint" bounds can be obtained on the values
of a subset of the variables by adapting the approach used by the Bound-and-Scan
algorithm [3] in which the variables are restricted to be non-negative integers
rather than binary. In addition we give good indications that these bounds may
be used efficiently. Finally, a class of problems particularly well-suited to this bounding procedure is specified.

2. Conceptual Outline of the Bounding Technique

This technique is motivated by the following considerations: In an implicit enumeration algorithm the goal is to quickly eliminate large subsets of the $2^n$ possible combinations of values for the variables by various tests. Having a bound on the objective function value of the optimal solution provides one such test since constraining feasible points to yield a better objective function value greatly decreases the size of the feasible region. Also, this new constraint assures that any new feasible solution will decrease the size of the feasible region even further. Moreover, if we can obtain a representation for a group of variables that (1) allows us to fix the values of some of the variables in an optimal solution to (P) and (2) yields "joint" bounds on them which allow systematic generation of all relevant combinations of their values, we will have eliminated even more of the $2^n$ combinations. Further, if the number of variables in this group is large as a function of $n$ and we use the "joint" bounds in a computationally frugal way, additional time savings will be achieved as a result of having only a small number of variables to deal with in another manner. The technique presented here seeks to take advantage of these ideas.

Consider the region determined by the constraints that are binding in the optimal solution to $(P^*_R)$, after making a change of variables $(x'_j = 1 - x_j)$ for those variables, $x_j$, whose complements, $(1 - x_j)$, were nonbasic in this solution [5, pp. 659-662]. In addition, require that the value of the objective function

\[^1\text{Here we exclude constraints that are binding due to degeneracy. Hence, the binding constraints are those constraints } \sum_{j=1}^{n} a_{ij}x_j \leq b_i \text{ (} i = 1,2,\ldots,m \text{) whose slack variables are nonbasic in the optimal solution to } (P^*_R), \text{ and those non-negativity constraints that correspond, after the change of variables, to nonbasic variables in this solution.}\]
not be less than the bound generated by the known feasible solution to (P). The optimal solution to (P) clearly lies in this region.

Identify the extreme points of this region, and represent each feasible point (as defined by this region) as a convex combination of these extreme points; then use the remaining constraints to perhaps eliminate some of these convex combinations from further consideration. This provides a convenient way of identifying the relevant combinations of values for the variables that were nonbasic, at least after the change of variables, in the optimal solution to (PR). Each time a relevant set of values is determined we exit from this technique to the underlying algorithm to see if this partial set of values for the xj can yield a completion which satisfies all the constraints of (P). If we find such a feasible solution it yields a new lower bound on the optimal objective function value of (P). We then seek to identify another relevant set of values for this same group of variables. Hence, an iteration corresponds to a relevant set of values for these variables and the algorithm terminates when all such sets have been considered.

3. Notation and Initialization

This section introduces the notation and terminology that will be used in what follows. An alternative representation of (P) is

\[
\text{maximize } \quad x_0 = c^T x \\
\text{subject to } \\
Ax \leq b \\
x_j = 0 \text{ or } 1, \quad (j = 1, \ldots, n)
\]

where c is a 1 x n vector, x is an n x 1 vector, b is an m x 1 vector and A is an m x n matrix. An n x 1 vector x is a feasible solution to (P) if it is in the region defined by the constraints; it is an optimal
solution if it is feasible and maximizes the objective function value among all feasible solutions. The value of $x_0^*$ for an optimal solution will be called the optimal objective function value. If values have been assigned to only certain of the variables (components of $x$), this specification of values will be called a partial solution for these variables. Given a partial solution, a "completion" is a solution resulting from specification of values for the remaining variables. An eligible partial solution is one that does not violate any known bounds on the variables involved, i.e., one that, based on our present information, could yield a completion that is an optimal solution.

The optimal solution to $(P_R)$ may be included in the card input stream or $(P_R)$ may be solved during initialization if the underlying algorithm is simplex-based. In either case, we will denote this solution as $x^* = [x_1^*, \ldots, x_n^*]^T$ and its objective function value as $x_0^* = c \cdot x^*$. Also we define

- $J_B = \{ j | x_j^* \text{ is basic} \}$,
- $J_0 = \{ j | x_j^* = 0 \text{ and } x_j^* \text{ is nonbasic} \}$,
- $J_1 = \{ j | x_j^* = 1 \text{ and } 1 - x_j^* \text{ is nonbasic} \}$,
- $J_N = J_0 \cup J_1$,
- $K_B = \{ i | i \in \{ 1, \ldots, m \} \text{ and the } i\text{th constraint is binding at } x^* \}$,
- $K_N = \{ 1, \ldots, m \} - K_B$.

We wish to make the following change of variables to define $x' \in \mathbb{R}^n$ for any $x \in \mathbb{R}^n$.

$$x'_j = \begin{cases} 1-x_j, & \text{if } j \in J_1 \\ x_j, & \text{otherwise.} \end{cases}$$

This change of variables directly induces the problem

$$\begin{align*}
\text{maximize} & \quad x_0' = c' \cdot x' \\
\text{subject to} & \quad \Lambda' x' \leq b' \\
& \quad x'_j = 0 \text{ or } 1
\end{align*}$$

$(P')$
from (P). Note that an optimal solution to (P'), \( \mathbf{x}^{\text{opt}} \), yields an optimal solution, \( \mathbf{x}^{\text{opt}} \), to (P) via

\[
\mathbf{x}_j^{\text{opt}} = \begin{cases} 
1 - \mathbf{x}_j^{\text{opt}}, & \text{if } j \in J_1 \\
\mathbf{x}_j^{\text{opt}}, & \text{otherwise} 
\end{cases}
\]

Henceforth we will direct our attention to solving (P').

For notational convenience it will be assumed that the original indexing of the \( x_j \) and \( b_1 \) was such that if \( n_B = |J_B| \), then

\[
J_B = \{1, \ldots, n_B\}, \\
K_B = \{1, \ldots, n_B\}, \\
J_N = \{n_B+1, \ldots, n\}, \\
K_N = \{n_B+1, \ldots, m\}.
\]

Finally denote the initial good feasible solution to (P) by \( \mathbf{x}^{(F)} = [x_1^F, \ldots, x_n^F]^T \). Now define \( \mathbf{x}_i^{(F)} \) by

\[
x_j^{(F)} = \begin{cases} 
1 - x_j^{(F)}, & \text{if } j \in J_1 \\
x_j^{(F)}, & \text{otherwise}, 
\end{cases}
\]

and let \( x_0^{(F)} = c^T \mathbf{x}^{(F)} \). Note that \( x_0^{(F)} \) is a lower bound on the optimal objective function value of (P').

4. Technique for Bounding \( \{x_j \mid j \in J_N\} \)

Define the \( n \times n \) matrix \( A^* \) whose ith row is

\[
A_i^* = \begin{cases} 
A_i', & i = 1, \ldots, n_B \\
-e_i, & i = n_B+1, \ldots, n, 
\end{cases}
\]

where \( e_i \) is the unit \( n \)-vector with unity assigned to component \( i \). Also define \( b^* \in \mathbb{R}^n \) whose ith component is
b_i^* = \begin{cases} 
  b_i', & i = 1,\ldots,n_B \\
  0, & i = n_B+1,\ldots,n 
\end{cases}

Now consider the set of points satisfying

(i) \( A^* \mathbf{x} \leq b^* \),

(ii) \( \mathbf{c}' \mathbf{x} \geq x'_0(F) \).

Since (i) is the set of constraints that are binding on \( \mathbf{x}'^* \) (induced by \( \mathbf{x}^* \) via the change of variables) and \( x'_0(F) \) is a lower bound on the optimal objective function value of \( (P') \), any optimal solution to \( (P') \) must be in this set. If \( (\mathbf{c}'\mathbf{x}'^* - \mathbf{c}'\mathbf{x}'_0(F)) \) is small, this set will contain few binary points and the search for an optimal binary point may not be difficult.

We now seek to find the \((n+1)\) extreme points of the \(n\)-simplex defined by (i) and (ii). The first \(n\) extreme points are determined by the system of equations,

\[
A^{(1)} \mathbf{x}^{(1)} = b^{(1)}, \quad (i = 1,\ldots,n)
\]

where \( A^{(1)} \) is obtained by replacing row \(i\) of \( A^* \) by \( \mathbf{c}' \), and \( b^{(1)} \) is the vector obtained by replacing component \(i\) of \( b^* \) by \( x'_0(F) \). The \((n+1)\)st extreme point is \( \mathbf{x}^{(0)} = \mathbf{x}'^* \) [3, p. 646]. It can be shown that \( A^{(1)} \) is nonsingular \((i = 1,\ldots,n)\) [3, p. 646] and, hence, we have explicitly

\[
\mathbf{x}^{(1)} = (A^{(1)})^{-1} b^{(1)}, \quad (i = 1,\ldots,n).
\]

We wish to represent the feasible points as convex combinations of these extreme points. To this end define

\[
\mathbf{M} = \begin{bmatrix} 
  \mathbf{x}^{(0)} & \mathbf{x}^{(1)} & \cdots & \mathbf{x}^{(n)} \\
  1 & 1 & \cdots & 1 
\end{bmatrix}
\]

and

\[
\mathbf{\rho} = [\rho_0, \ldots, \rho_n]^T.
\]
Then setting

\[ M_0 = \begin{bmatrix} x \\ 1 \end{bmatrix} \]

yields such a representation for \( x \) via \( \varphi \). \( x \) is in the simplex determined by (i) and (ii) if and only if \( \varphi \geq 0 \). For an arbitrary \( x \), \( \varphi \) exists by virtue of the fact that \( M \) is nonsingular [3, pp. 648, 649], so

\[ \varphi = M^{-1} x \]

provides the relevant vector \( \varphi \).

Fortunately, this representation can be obtained directly without explicitly performing a matrix inversion, since \( A_1^* x^{(j)} = b_1^* \) for \( j \neq i \) (by definition of \( A^{(1)} \) and \( b^{(1)} \), \( i = 1, \ldots, n \)), so

\[
\begin{align*}
    b_1^* - A_1^* x &= b_1^* - \sum_{j=0}^{n} \rho_j A_1^* x^{(j)} \\
    &= b_1^* - (1 - \rho_1) b_1^* - \rho_1 A_1^* x^{(1)} \\
    &= \rho_1 (b_1^* - A_1^* x^{(1)}).
\end{align*}
\]

Therefore,

\[ \rho_1 = \frac{b_1^* - A_1^* x}{b_1^* - A_1^* x^{(1)}}, \quad i = 1, \ldots, n \]

and, similarly,

\[ \rho_0 = \frac{c^* x - x_0^{(F)}}{c^* x^{(0)} - x_0^{(F)}}. \]

Solving \((P')\) has now been reduced to finding those \( x \) with \( \rho \geq 0 \) which satisfy the constraints of \((P')\) that are not binding on \( x^* \) and identifying the one which maximizes \( c^* x \).

It can be shown that each element of \( \{ x_j | j \in J_n \} \) has the property that it has value zero at all of the extreme points of our \( n \)-simplex except one, where
it has a strictly positive value. In particular, \( x_j^{(j)} > 0 \) for each \( j \in J_N \).

This follows from Lemma 3 [3, p. 646], which for completeness is restated here.

**Lemma:** Under the assumptions of Section 1, \( x_i^{(1)} (i = 0, \ldots, n) \) satisfies constraints (i) and (ii). Furthermore, \( A_i^{*} x^{(i)} < b_i^{*} \) for \( i = 1, \ldots, n \) and \( c' x^{(0)} > x_0 (F) \).

Hence, for \( j \in J_N \),

\[
\rho_j = \frac{b_j^{*} - A_j^{*} x}{b_j^{*} - A_j^{*} x^{(j)}}
\]

\[
= \frac{x_j}{x_j^{(j)}}.
\]

This result, combined with the fact that \( \rho_0 \) increases as the best known objective function value of \((P')\) increases, yields the following theorem [3, p. 652].

**Theorem:** A necessary condition for \( [x_{n+B+1}^{*}, \ldots, x_n^{*}]^T \) to be an eligible partial solution to \((P')\) for the variables whose indices are in \( J_N \) is that

\[
\sum_{j=n+B+1}^{n} \frac{x_j}{x_j^{(j)}} \leq 1 - \rho_0^B,
\]

where \( \rho_0^B \) is the value of \( \rho_0 \) for the current best feasible solution to \((P')\), \( x^B = [x_{n+B+1}^{B}, \ldots, x_n^{B}]^T \) (with objective function value \( x_0^B \)).

This leads immediately to the following corollary.

**Corollary:** If \( x_j^{(j)} < 1 \) where \( j \in J_N \), then \( x_j' = 0 \) in any optimal solution to \((P')\).

**Proof:** \( x_j^{(j)} < 1 \) implies that \( 1/x_j^{(j)} > 1 \), since \( x_j^{(j)} > 0 \). Hence, \( \rho_j > 1 \) when \( x_j = 1 \) and this partial solution for \( x_j \) is not eligible.

Thus we may set \( x_j' = 0 \) for those \( j \) where \( x_j^{(j)} < 1 \). For notational convenience let this occur for \( j = n_B + 1, \ldots, n_0 \), and our necessary condition is now reduced to
Thus, by calculating \( \frac{1}{x_j(j)} \) for \( j = n_0+1, \ldots, n \) once at the outset and updating \( \rho_0 \) whenever a new feasible solution to \((P')\) is found, a single addition and comparison are needed to check whether a current partial solution for these variables remains eligible if a particular value is increased from zero to one. Hence, it is simple to generate all the eligible partial solutions one at a time, stopping each time to explore its completions using the underlying algorithm.

A possible enumeration scheme is initiated by starting at \( x_{n_0+1} \cdots, x_n \) \( ^T = 0 \). Each subsequent partial solution is obtained by first searching the set \( \{ x_{n_0+1} \cdots, x_n \} \) for the first zero value, \( x_q \), and then setting \( x_{n_0+1} = x_{n_0+2} = \ldots = x_{q-1} = 0 \) and \( x_q = 1 \) and checking the necessary condition for eligibility. If this new partial solution is not eligible, again search for the first zero value, \( x_p \), and set \( x_{n_0+1} = \ldots = x_{p-1} = 0 \) and \( x_p = 1 \) and check for eligibility, etc.

When there does not exist a zero-valued variable in \( \{ x_{n_0+1} \cdots, x_n \} \) the eligible partial solutions have been exhausted and the algorithm terminates.

5. Computational Considerations

The bounding technique presented in the preceding section has not yet been tested for computational efficiency. However, we discuss below some indications that it will be efficient, and then consider which kinds of problems can best take advantage of it.

It was mentioned in Section 4 that a small value of \( (c'x^* - c'x'(F)) \) is a key to the efficiency of the technique. Several heuristic procedures for obtaining \( x'(F) \) have been tested and shown to give good results [9], so a relatively small value of the above seems to be a reasonable expectation. Such
a value also tends to yield relatively small values for the $x_j^{(j)}$, which greatly reduces the number of partial solutions that are eligible. For example, if $1 \leq x_j^{(j)} < 2$ for all $j = n_0 + 1, \ldots, n$, then the number of eligible partial solutions grows only linearly with $(n - n_0)$, whereas this number could grow essentially exponentially if the $x_j^{(j)}$ are sufficiently large.
In identifying the extreme points $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$ it is not really necessary to find the $(A^{(1)})^{-1}$ or even to construct the $A^{(1)}$ (or, in fact, $A^*$) in a simplex-based code. Beginning with the inverse of the final basis matrix for $(P_R)$ (recalling that the change of variables does not affect the basic columns of $A$), the extreme points may be computed by performing a succession of pivots. First append the constraint $c'x' \geq x'_0^{(F)}$ to $(P_R)$ (induced by $(P_R)$ via the change of variables), introducing its slack variable as the additional basic variable in the optimal solution. Now successively introduce each nonbasic variable $(x_j : j \in J_N)$ into the basis while removing the variable that was introduced into the preceding basis. The first $n$ components of the resulting $n$ basic solutions are the desired extreme points.

An explicit representation for the $\rho_1$ has already been given, obviating the need for finding $M^{-1}$ or even constructing $M$.

Hence, thought the theory involved in arriving at this approach is rather involved, the only additional storage (beyond that of the underlying algorithm) is $\mathbf{x}^{(1)}$ ($i = 0, \ldots, n$), $\rho_0^B$, $1/x_j^{(1)} (j = n_0 + 1, \ldots, n)$, $n_0^B$, and a vector indicating which $j$ are in $J_1$. The latter is needed to deal with the required change of variables. The added computation involves the change of variables, the addition of the constraint $c'x' \geq x'_0^{(F)}$ to $(P_R')$, $n$ pivots to find $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$, the computation of $1/x_j^{(j)} (j = n_0 + 1, \ldots, n)$ and then one addition and comparison for each partial solution in the enumeration scheme delineated in the previous section.

These modest computational and storage requirements suggest that we should usually be able to efficiently investigate a relatively large number of variables with this bounding technique. In fact, a similar technique embedded in Hillier's Bound-and-Scan algorithm for the general integer linear programming problem has proven to be quite efficient under computational testing [3].
We know that \( n_B \), the number of variables that are not handled by this technique, is bounded above by \( m \), the number of functional constraints. This result follows from Weingartner's proof that at most \( m \) of the original variables can be basic in a solution to (P) [8, pp. 35-37]. Therefore, the bounding technique seems particularly promising for problems where \( m \) is small, since relatively few variables would then need to be handled by the underlying algorithm.

In summary, in a simplex-based implicit enumeration procedure to solve (P), the use of this bounding technique for problems where the number of constraints is small in comparison to the number of variables seems very promising computationally. It will be implemented and tested in the near future and the results reported.

References


A BOUNDING TECHNIQUE FOR INTEGER LINEAR PROGRAMMING WITH BINARY VARIABLES

FREDERICK S. HILLIER and NANCY E. JACQMIN

DEPT. OF OPERATIONS RESEARCH
STANFORD UNIVERSITY, STANFORD, CALIF.  

OPERATIONS RESEARCH PROGRAM
OFFICE OF NAVAL RESEARCH  CODE 434
ARLINGTON, VA. 22217

This research was supported in part by National Science Foundation Grant NSF/NSF No. MCS 76-81259 A01, Dept. of Operations Research, Stanford University.

INTEGER PROGRAMMING
IMPLICIT ENUMERATION ALGORITHMS
A Bounding Technique for Integer Linear Programming with Binary Variables

by

Frederick S. Hillier and Nancy E. Jacqmin
Department of Operations Research
Stanford University

ABSTRACT

We present a bounding technique for use in implicit enumeration algorithms for solving the integer linear programming problem with binary variables. The main assumptions used by this technique are (1) the associated linear program, obtained by dropping the integrality constraints on the variables, possesses a unique optimal solution, (2) this optimal solution is not binary, and (3) a good feasible solution to the original problem is available. An alternative to assumption (3) which is weaker is also presented.

We show that joint bounds can be obtained on the values of a subset of the variables. In addition we give an efficient method to implement this bounding technique. Finally, a class of problems particularly well-suited to this bounding procedure is specified.

Report No. 86