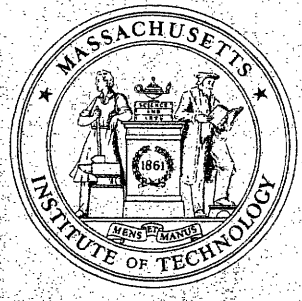
OPERATIONS RESEARCH CENTER

working paper



MASSACHUSETTS INSTITUTE OF TECHNOLOGY

PARAMETRIC INTEGER PROGRAMMING: THE RIGHT-HAND-SIDE CASE

by

Roy E. Marsten*
and
Thomas L. Morin**

OR 050-76 March 1976

*Operations Research Center
M.I.T.
Cambridge, MA

**School of Industrial Engineering
Purdue University
West Lafayette, IN

Supported in part by the U.S. Army Research Office (Durham) under Contract DAHC04-73-C-0032.

Abstract

A family of integer programs is considered whose right-hand-sides lie on a given line segment L. This family is called a parametric integer program(PIP). Solving a (PIP) means finding an optimal solution for every program in the family. It is shown how a simple generalization of the conventional branch-and-bound approach to integer programming makes it possible to solve such a (PIP). The usual bounding test is extended from a comparison of two point values to a comparison of two functions defined on the line segment L. The method is illustrated on a small example and computational results for some larger problems are reported.

Acknowledgement

The computer implementation of the algorithm reported here was done by Lee Aurich and Nancy Kaplan.

Table of Contents

1. Introduction

The purpose of this paper is to show how a simple generalization of the conventional branch-and-bound approach to integer programming makes it possible to solve a parametric integer program. Following Nauss [6] we shall call the family of programs

$$(P_\theta) \quad \max \sum_{j=1}^{n} r_j x_j$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i + \theta d_i \qquad 1 \le i \le m$$

$$x_j \in \{0,1\} \qquad 1 \le j \le n$$

for $0 \le \theta \le 1$ a single parametric integer program (PIP). By "solving" (PIP) we shall mean obtaining an optimal solution of (P_θ) for every $0 \le \theta \le 1$ for which (P_θ) is feasible. We assume that (P_θ) is feasible for at least one value of θ.

Parametric integer programming has only recently emerged as a topic of research. The pioneering papers include Noltemeier [7], Roodman [10,11], Piper and Zoltners [8,9], and Bowman [1]. Nauss [6] has reviewed this earlier work and contributed many new results for parameterizations of the objective function. The present paper, which has grown out of the authors' work on synthesizing dynamic programming with branch-and-bound [3,4,5], is devoted to the right-hand-side case.

In parametric linear programming, the first step is to solve (P_0), i.e. (P_θ) for $\theta=0$. Then the direction vector $d=(d_1, \ldots, d_m)$ is specified and the analysis is performed by driving θ from 0 to 1. Critical values of θ and new optimal solutions are identified one at a time as θ increases. In the procedure for parametric integer programming to be presented here, the direction d must be specified in advance. The (PIP) is solved in one branch-and-bound search. The usual bounding test is modified so that a partial solution is eliminated only if none of its descendants is optimal for any (P_θ), $0\leq\theta\leq1$. This means that some partial solutions must be retained that could otherwise be eliminated if only (P_0) were of interest. The severity of the resulting computational burden depends on the magnitude of d.

The organization of the paper is as follows. A prototype branch-and-bound algorithm for (P_0) is presented in Section 2. The lower bound and upper bound functions are developed in Sections 3 and 4, respectively. The modified branch-and-bound algorithm for (PIP) is given in Section 5 and applied to a sample problem in Section 6. Computational experience with the algorithm is reported in Section 7.

2. A prototype branch-and-bound algorithm

We shall draw upon the framework and terminology of Geoffrion and Marsten [2] to describe a simple linear programming based branch-and-bound algorithm for (P_0). Problem (P_0) is separated, by fixing variables at zero

and one, into smaller candidate problems (CP^q). Each candidate problem has an associated set of fixed variables $F^q \subseteq J=\{1, \ldots, n\}$ and partial solution x^q. That is, (CP^q) is defined by the conditions $x_j = x_j^q$ for $j \in F^q$. The current set of candidate problems is called the candidate list. If any feasible solutions of (P_0) are known, the best of these is called the incumbent and its value denoted by LB. If we let $J^q = J-F^q$ be the set of "free" variables and

$$\beta^q = \sum_{j \in F^q} A_j x_j^q$$

where A_j is the jth column of A, then a typical candidate problem may be written as

$$(CP^q) \quad \sum_{j \in F^q} r_j x_j^q + \max \sum_{j \in J^q} r_j x_j$$

$$\text{subject to} \quad \sum_{j \in J^q} a_{ij} x_j \leq b_i - \beta_i^q \qquad 1 \leq i \leq m$$

$$x_j \in \{0,1\} \qquad j \in J^q$$

An upper bound on the value of (CP^q) is obtained by solving its LP relaxation (CP_R^q). It is also helpful to compute a lower bound on the value of (CP^q). This can be done by using a heuristic to find a feasible solution of (CP^q). This feasible solution, if it is better than the incumbent, becomes the new incumbent. A prototype branch-and-bound algorithm may now be described as follows.

Step 1. Place (P_0) in the candidate list and set LB = $-\infty$.

Step 2. If the candidate list is empty, stop. If there is an incumbent, it is optimal for (P_0). Otherwise (P_0) is infeasible.

Step 3. Select a candidate problem and remove it from the candidate list. Call it (CP^q).

Step 4. Solve the linear program (CP_R^q). Let UB^q denote its optimal value.

Step 5. If $UB^q \leq LB$, go to Step 2.

Step 6. If the optimal solution of (CP_R^q) is all integer, make this solution the new incumbent, set $LB = UB^q$, and go to Step 2.

Step 7. Use a heuristic to find a feasible solution of (CP^q). Let H^q denote its value. If $H^q > LB$, then make this solution the new incumbent and set $LB = H^q$.

Step 8. Separate (CP^q) into two new candidate problems $(CP^{q'})$ and $(CP^{q''})$ by choosing $p \in J^q$ and setting $F^{q'} = F^{q''} = F^q \cup \{p\}$, $x_p^{q'} = 0$, $x_p^{q''} = 1$. Place $(CP^{q'})$ and $(CP^{q''})$ in the candidate list and return to Step 2.

A great many variations on this pattern are described in [2], but this prototype will suffice for our purposes. Step 5 is the bounding test. If this test is satisfied, then no descendant of x^q is better than the incumbent. Notice that the bounding test includes the case where (CP_R^q), and hence (CP^q), is infeasible since then $UB^q = -\infty$. If (CP^q) does not have to be separated at Step 8, then we say that it has been fathomed. This occurs if (CP^q) passes the bounding test or if (CP_R^q) has an all integer solution. Step 7, the heuristic, is optional. Its purpose is to strengthen the bounding test by improving the incumbent and increasing LB.

The modifications that must be made to this prototype algorithm to solve (PIP) are confined to Steps 5, 6, and 7. The notion of the incumbent must be generalized from a single value LB to a function $LB(\theta)$ defined on $0 \leq \theta \leq 1$. The upper bound must also be expressed as a function of θ: $UB^q(\theta)$. The bounding test then becomes a comparison of two functions on the interval $0 \leq \theta \leq 1$ rather than just a point comparison for $\theta = 0$.

3. The optimal return and lower bound functions.

In this section we shall investigate the behavior of the optimal value of an integer program as a function of its right-hand-side. Let the optimal return function

$$f(b') = \max \; rx \quad \text{subject to } Ax \leq b'$$
$$x \in \{0,1\}$$

be defined for $b' \in R^m$. It is apparent that $f(b')$ is nondecreasing in each component of b'. Let $\{x^k \mid k \in K\}$ be the set of all feasible solutions of (PIP), i.e. of all (P_θ) for $0 \leq \theta \leq 1$. For each $k \in K$, define the step function

$$f^k(b') = \begin{cases} \sum_{j=1}^{n} r_j x_j^k & \text{if } b' \geq \sum_{j=1}^{n} A_j x_j^k \\ -\infty & \text{otherwise} \end{cases}$$

for all $b' \in R^m$.

The optimal return function $f(b')$ is the pointwise maximum of this finite collection of nondecreasing step functions

$$f(b') = \max \{f^k(b') \mid k \in K\}$$

and is therefore itself a nondecreasing step function.

Now suppose that the solutions $\{x^k \mid k \in \bar{K}\}$ are known, where $\bar{K} \subseteq K$. A lower approximation of $f(b')$ may be constructed from these known solutions, namely

$$\bar{f}(b') = \max \{f^k(b') \mid k \in \bar{K}\}.$$

Clearly $\bar{f}(b')$ is also a nondecreasing step function and is a lower bound function for $f(b')$, i.e. $\bar{f}(b') \leq f(b')$ for all $b' \in R^m$. The approximation can be improved as new feasible solutions are discovered.

We are interested in a particular "slice" of $f(b')$ and $\bar{f}(b')$: the line segment $\{b+\theta d \mid 0 \leq \theta \leq 1\}$ where b is the right-hand-side of (P_0) and d is the given direction vector. Define $g(\theta)=f(b+\theta d)$ and $LB(\theta)=\bar{f}(b+\theta d)$ for $0 \leq \theta \leq 1$. Then $g(\theta)$ and $LB(\theta)$ are both step functions and $LB(\theta) \leq g(\theta)$ for all $0 \leq \theta \leq 1$. If $d \geq 0$, then $g(\theta)$ and $LB(\theta)$ are both nondecreasing. (See Figure 1). There is at least one optimal solution of (PIP) associated with each step of $g(\theta)$. Solving (PIP) is equivalent to constructing $g(\theta)$ by finding at least one x solution for each of its steps.

The procedure for constructing $LB(\theta)$ from the known feasible solutions is as follows. For each $k \in \bar{K}$ define

$$\theta_1^k = \min \{\theta \mid \sum_{j=1}^{n} A_j x_j^k \leq b + \theta d\} \tag{3.1}$$

$$\theta_2^k = \max \{\theta \mid \sum_{j=1}^{n} A_j x_j^k \leq b + \theta d\} \tag{3.2}$$

where $\theta_1^k = \theta_2^k = +\infty$ if the indicated set is empty. Then

$$LB^k(\theta) = \begin{cases} \sum_{j=1}^{n} r_j x_j^k & \text{if } \theta_1^k \leq \theta \leq \theta_2^k \\ -\infty & \text{otherwise} \end{cases} \tag{3.3}$$

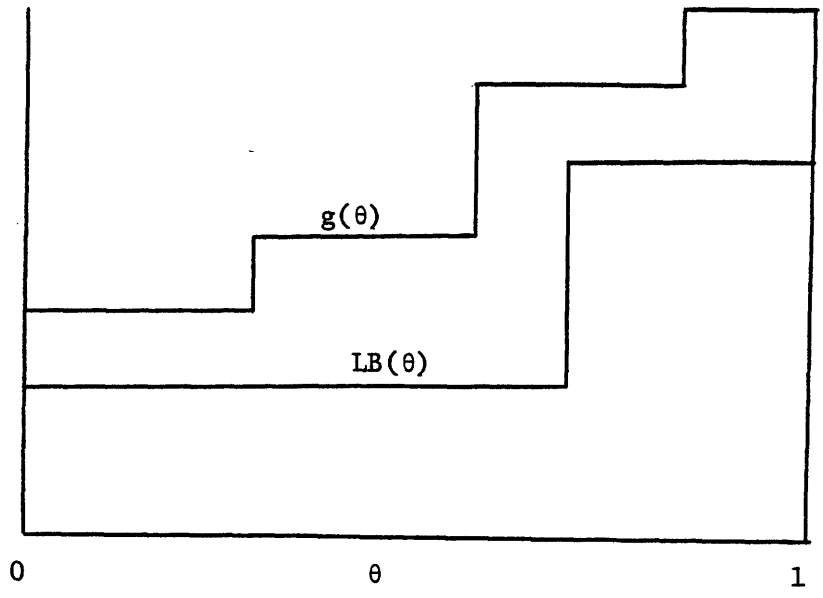


Figure 1. Typical $g(\theta)$ and $LB(\theta)$ functions.

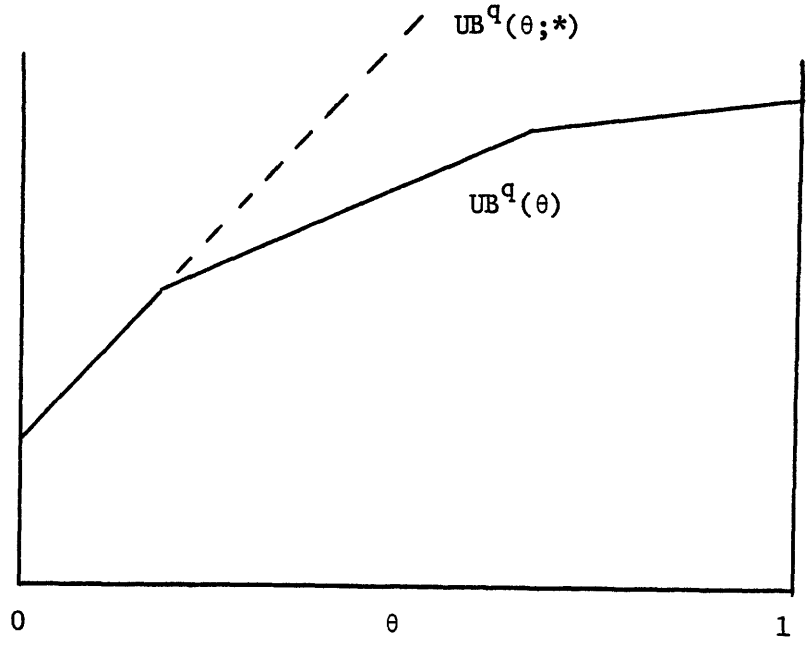


Figure 2. Typical $UB^q(\theta)$ and $UB^q(\theta;*)$ functions.

and

$$LB(\theta) = \max\{LB^k(\theta) \mid k \in \bar{K}\}. \tag{3.4}$$

The solutions which determine $LB(\theta)$ will be called the incumbents. Each one is incumbent for a particular interval of θ.

4. The upper bound functions.

Consider a given partial solution x^q. In order to demonstrate that no descendant of x^q could be optimal for any (P_θ), we need an upper bound on the return achieved by any descendant and this upper bound must depend on θ. Such an upper bound can be obtained by introducing (θd) into the relaxed candidate problem (CP_R^q). Define

$$UB^q(\theta) = \sum_{j \in F^q} r_j x_j^q + \max \sum_{j \in J^q} r_j x_j$$

subject to

$$\sum_{j \in J^q} a_{ij} x_j \le b_i + \theta d_i - \beta_i^q \qquad 1 \le i \le m$$

$$0 \le x_j \le 1 \qquad j \in J^q$$

so that $UB^q(0) = UB^q$. It is well known that $UB^q(\theta)$ is concave and piecewise linear on $0 \le \theta \le 1$. The function $UB^q(\theta)$ could be obtained explicitly by ordinary parametric linear programming. The computational burden involved in doing this for every candidate problem could be quite substantial, however. Fortunately any dual feasible solution of (CP_R^q) can be used to construct a linear upper bound function for $UB^q(\theta)$. An optimal dual solution of (CP_R^q), barring degeneracy, yields the first linear segment of $UB^q(\theta)$.

By linear programming duality we know that;

$$UB^q(\theta) = \sum_{j\in F^q} r_j x_j^q + \min \sum_{i=1}^{m} u_i(b_i+\theta d_i-\beta_i^q) + \sum_{j=1}^{n} v_j$$

$$\text{subject to} \quad \sum_{i=1}^{m} u_i a_{ij} + v_j \geq r_j \qquad j \in J^q$$

$$u_i \geq 0 \qquad 1\leq i\leq m$$

$$v_j \geq 0 \qquad 1\leq j\leq n$$

For notational convenience we have included all of the v_j variables, even though $v_j=0$ for $j\epsilon F^q$ in any optimal solution. Let D^q denote the dual feasible region

$$D^q = \{(u,v)\geq 0 \mid \sum_{i=1}^{m} u_i a_{ij} + v_j \geq r_j \text{ for } j\epsilon J^q\}.$$

Since the primal variables are all bounded and at least one (P_θ) is feasible, we may conclude that D^q is non-empty. Let $\{(u^t, v^t) \mid t\epsilon T^q\}$ and $\{(y^s, z^s) \mid s\epsilon S^q\}$ denote the sets of extreme points and extreme rays, respectively, of D^q. Taking $e=(1, \ldots, 1)$ we have

$$UB^q(\theta) \leq \sum_{j\in F^q} r_j x_j^q + u^t(b+\theta d-\beta^q) + v^t e$$

for all $t\epsilon T^q$, with equality if (u^t, v^t) is optimal for the objective function $u(b+\theta d-\beta^q) + ve$. As a function of θ then, the return achieved by any descendant of x^q is bounded above by:

$$UB^q(\theta;t) = (u^t d)\theta + [\sum_{j\in F^q} r_j x_j^q + u^t(b-\beta^q) + v^t e]$$

for any $t\epsilon T^q$. This is a linear function of θ and, since $u^t\geq 0$, it is nondecreasing if $d\geq 0$.

In the modified branch-and-bound algorithm for (PIP), linear programming is applied to (CP_R^q) as usual. The functions $UB^q(\theta;t)$ are obtained at no extra cost. The function obtained from an optimal dual solution will be denoted $UB^q(\theta;*)$. Barring degeneracy, $UB^q(\theta;*)$ coincides with the first linear segment of $UB^q(\theta)$. (See Figure 2). As in conventional branch-and-bound, if the dual simplex method is used, then suboptimal dual solutions can be used to perform additional weaker tests.

If (CP_R^q) is infeasible, then the simplex method will terminate with an extreme point $(u^t, v^t) \geq 0$ and an extreme ray $(y^s, z^s) \geq 0$, such that

$$y^s(b-\beta^q) + z^s e < 0.$$

If $y^s d \leq 0$, then we may conclude that $UB^q(\theta) = -\infty$ for all $0 \leq \theta \leq 1$. If $y^s d > 0$, then $UB^q(\theta) = -\infty$ for $0 \leq \theta < \theta^*$ and $UB^q(\theta) \leq UB^q(\theta;t)$ for $\theta^* \leq \theta \leq 1$, where

$$\theta^* = \frac{-y^s(b-\beta^q) - z^s e}{y^s d}.$$

5. A branch-and-bound algorithm for (PIP).

Now that the upper and lower bound functions have been derived, the generalized bounding test may be stated. The partial solution x^q does not have a descendant that is better than an incumbent if

$$UB^q(\theta) \le LB(\theta) \qquad \text{for all} \quad 0 \le \theta \le 1$$

or if

$$UB^q(\theta;t) \le LB(\theta) \qquad \text{for all} \quad 0 \le \theta \le 1$$

for some $t \in T^q$. This test is the basis for a modified branch-and-bound algorithm that can solve (PIP).

Step 1. Place (P_0) in the candidate list and set $LB(\theta) = -\infty$ for $0 \le \theta \le 1$.

Step 2. If the candidate list is empty, stop.
$LB(\theta) = g(\theta)$ for $0 \le \theta \le 1$.

Step 3. Select a candidate problem and remove it from the candidate list. Call it (CP^q).

Step 4. Solve (CP_R^q). If it is infeasible, obtain the appropriate dual extreme point (u^*, v^*) and extreme ray (y^*, z^*). Otherwise obtain an optimal dual solution (u^*, v^*).

Step 5. I. (CP_R^q) infeasible.

(a) $y^*d \le 0$. Go to Step 2.

(b) $y^*d > 0$. Set $\theta^* = [-y^*(b-\beta^q)-z^*e] \,/\, y^*d$.
If $UB^q(\theta;*) \le LB(\theta)$ for all $\theta^* \le \theta \le 1$, go to Step 2.

II. (CP_R^q) feasible.

If $UB^q(\theta;*) \le LB(\theta)$ for all $0 \le \theta \le 1$, go to Step 2.

Step 6. If the optimal primal solution of (CP_R^q) is all integer, use it to update $LB(\theta)$.

Step 7. Use a heuristic to find feasible solutions of (CP^q) with right-hand-side $(b+\theta d)$ for several values of θ. Use these feasible solutions to update $LB(\theta)$.

Step 8. Separate (CP^q) into two new candidate problems $(CP^{q'})$ and $(CP^{q''})$ by choosing $p \in J^q$ and setting $F^{q'} = F^{q''} = F^q \cup \{p\}$ $x_p^{q'} = 0$, $x_p^{q''} = 1$. Place $(CP^{q'})$ and $(CP^{q''})$ in the candidate list and return to Step 2.

The validity of the generalized bounding test insures that an optimal solution for every (P_θ), $0 \le \theta \le 1$, will be found by the search. At worst, an optimal solution may not be discovered until the bottom of the branch-and-bound tree is reached ($F^q = J$). This guarantees that $LB(\theta)$ will coincide with $g(\theta)$ by the time the algorithm is finished. It remains only to show how the optimal solutions are identified.

Let $\{x^k | k \in \bar{K}\}$ be the set of incumbents when the algorithm terminates. Let $\theta \in [0,1]$ and suppose that (P_θ) is feasible, $g(\theta) > -\infty$. From the construction of $LB(\theta)$, (3.1) - (3.4), we know that there is some $k \in \bar{K}$ such that

$$\begin{aligned} g(\theta) &= LB(\theta) \\ &= LB^k(\theta) \\ &= \sum_{j=1}^{n} r_j x_j^k > -\infty . \end{aligned}$$

Furthermore, $LB^k(\theta) > -\infty$ means that $\theta_1^k \le \theta \le \theta_2^k$, or equivalently that

$$\sum_{j=1}^{n} A_j x_j^k \le b + \theta d.$$

Since x^k is feasible for (P_θ) and its return is equal to $g(\theta)$, it follows that x^k is optimal for (P_θ). To summarize, if $k \in \bar{K}$ and $\theta \in [0,1]$, then x^k is optimal for (P_θ) if and only if

$$\text{i)} \quad \sum_{j=1}^{n} A_j x_j^k \le b + \theta d$$

and $$\text{ii)} \quad \sum_{j=1}^{n} r_j x_j^k = g(\theta) .$$

At Step 6, in contrast to the prototype algorithm, x^q is not fathomed when the optimal primal solution of (CP_R^q) is all integer. This is because x^q may have other descendants which are optimal for $\theta>0$. The use of heuristics at Step 7, while in principle optional, is an important part of the algorithm since integer solutions of (CP_R^q) can only yield $LB(\theta) = LB(0)$ for $0\leq\theta\leq1$. The heuristics are needed to produce stronger values of $LB(\theta)$ for $\theta > 0$.

As with the prototype algorithm, the above procedure will admit considerable variation and refinement. If the dual simplex method is used, then suboptimal dual solutions can be used to perform additional bounding tests. Cutting planes can be generated for any candidate problem to give stronger upper bound functions. Parametric linear programming can be used to generate more than the first segment of $UB^q(\theta)$. If a candidate problem with an all-integer LP solution has to be separated at Step 8, then the same LP solution is optimal for one of the two new candidates and does not have to be recomputed. Extensive experimentation will be required to determine the most effective computational tactics.

6. Example

In this section the algorithm will be applied to a simple example. In order to illustrate all of the different cases that can arise, the parameterization will be done over a relatively large interval. The test problem is

$$\max \quad 10x_1 + 15x_2 + 10x_3 + 5x_4$$

subject to

$$2x_1 + 3x_2 + 5x_3 + 1x_4 \leq 4 + \theta 4$$

$$4x_1 + 2x_2 + 1x_3 + 1x_4 \leq 4 + \theta 4$$

$$x_j \in \{0,1\} \qquad 1 \leq j \leq 4$$

Thus $b=(4,4)$, $d=(4,4)$ and increasing θ from zero to one amounts to doubling the right-hand-side. A picture of the optimal return function $f(b')$ is given in Figure 3. The dashed line indicates the line segment of interest: $\{b+\theta d \mid 0 \leq \theta \leq 1\}$. It is clear from this picture that three optimal solutions must be found, with values of 20, 25, and 30. These solutions are (0,1,0,1), (1,1,0,0), and (1,1,0,1) respectively. The $g(\theta)$ function, shown in Figure 4, is

$$g(\theta) = \begin{cases} 20 \text{ for } 0 \leq \theta < 1/2 \\ 25 \text{ for } 1/2 \leq \theta < 3/4 \\ 30 \text{ for } 3/4 \leq \theta \leq 1. \end{cases}$$

The optimal LP solution of (P_0) is $x=(1/2,1,0,0)$, $u=(5,0)$, $v=(0)$ with value $UB^0=20$. The rounded down solution has value 15 and is feasible for $\theta \geq 0$; the rounded up solution has value 25 and is feasible for $\theta \geq 1/2$.

This provides an initial lower bound function:

$$LB(\theta) = \begin{cases} 15 \text{ for } 0 \le \theta < 1/2 \\ 25 \text{ for } 1/2 \le \theta \le 1. \end{cases}$$

The complete branch-and-bound tree is displayed in Figure 5. The nodes will be discussed in the order in which they were created.

Node 1. LP solution: $x=(0,1,0,1)$, $u=(5,0)$, $v=(0)$, $UB^1=20$. $UB^1(\theta;*)=20\theta+20$. The LP solution is all integer and is feasible for $\theta \ge 0$. Therefore the lower bound function may be improved:

$$LB(\theta) = \begin{cases} 20 \text{ for } 0 \le \theta < 1/2 \\ 25 \text{ for } 1/2 \le \theta \le 1. \end{cases}$$

The bounding test for node 1 is shown in Figure 6. Node 1 is not fathomed.

Node 2. LP solution: $x=(1,0,0,0)$, $u=(0,10)$, $v=(0)$, $UB^2=10$. $UB^2(\theta;*)=40\theta+10$. The bounding test, shown in Figure 7, is not successful. Notice that if we were only interested in solving (P_0) we would be finished. Node 1 has an all integer solution with value 20 and node 2 has upper bound $UB^2=10<20=LB(0)$.

Node 3. LP solution: $x=(0,0,3/5,1)$, $u=(2,0)$, $v=(0,0,0,3)$, $UB^3=11$. $UB^3(\theta;*)=8\theta+11$. The bounding test, shown in Figure 8, is successful and node 3 is fathomed.

Node 4. Same as node 1, since optimal LP solution at node 1 has $x_2 = 1$.

Node 5. Same as node 2, since optimal LP solution at node 2 has $x_2 = 0$.

Node 6. LP is infeasible. The dual extreme point is u=(0,10), v=(0) and the extreme ray is y=(0,1), z=(0). The critical value of θ is $(-y(b-\beta^6)-ze)/yd=1/2$. Thus $UB^6(\theta)= -\infty$ for $0\le\theta<1/2$ and $UB^6(\theta;*)=40\theta+5$ for $1/2\le\theta\le1$. The bounding test is shown in Figure 9.

Node 7. Same as nodes 1 and 4, since optimal LP solution for node 4 has $x_3=0$.

Node 8. LP is infeasible. The dual extreme point is u=(5,0), v=(0) and the extreme ray is y=(1,0), z=(0). The critical value of θ is $(-y(b-\beta^8)-ze)/yd=1$, so $UB^8(\theta) = -\infty$ on $0\le\theta\le1$ and node 8 is fathomed.

Node 9. Same as nodes 2 and 5, since optimal LP solution for node 5 has $x_3=0$.

Node 10. LP is infeasible. The dual extreme point is u=(5,0), v=(0) and the extreme ray is y=(1,0), z=(0). The critical value of θ is $(-y(b-\beta^{10})-ze)/yd=3/4$. Thus $UB^{10}(\theta)= -\infty$ for $0\le\theta<3/4$ and $UB^{10}(\theta;*)=20\theta+5$ for $3/4\le\theta\le1$. Node 10 is therefore fathomed. See Figure 10.

Node 11. LP is infeasible. The dual extreme point is u=(0,5), v=(0) and the extreme ray is y=(0,1), z=(0). The critical value of θ is $(-y(b-\beta^{11})-ze)/yd=1/2$. Thus $UB^{11}(\theta) = -\infty$ for $0\le\theta<1/2$ and $UB^{11}(\theta;*)=20\theta+15$ for $1/2\le\theta\le1$. Node 11 is not fathomed. See Figure 11.

Node 12. LP is infeasible. The dual extreme point is u=(5,0), v=(0) and the extreme ray is y=(1,0), z=(0). The critical value of θ is $(-y(b-\beta^{12})-ze)/yd=1$ Therefore $UB^{12}(\theta) = -\infty$ for $0\le\theta\le1$ and node 12 is fathomed.

Nodes 13-18 are all at the bottom level of the search tree. The solution for node 18, (1,1,0,1), has value 30 and is feasible for $\theta \geq 3/4$. The lower bound function may be improved by redefining $LB(\theta)=30$ for $3/4 \leq \theta \leq 1$. $LB(\theta)$ now coincides with $g(\theta)$ on $0 \leq \theta \leq 1$. The algorithm terminates since the candidate list is empty.

The amount of extra computation required to solve (PIP), as compared to (P_0), depends on the length of the interval of parameterization. When this interval is small, the burden imposed by parameterization may be slight or even negligible. When it is large, however, as illustrated in this example, the burden can be quite substantial.

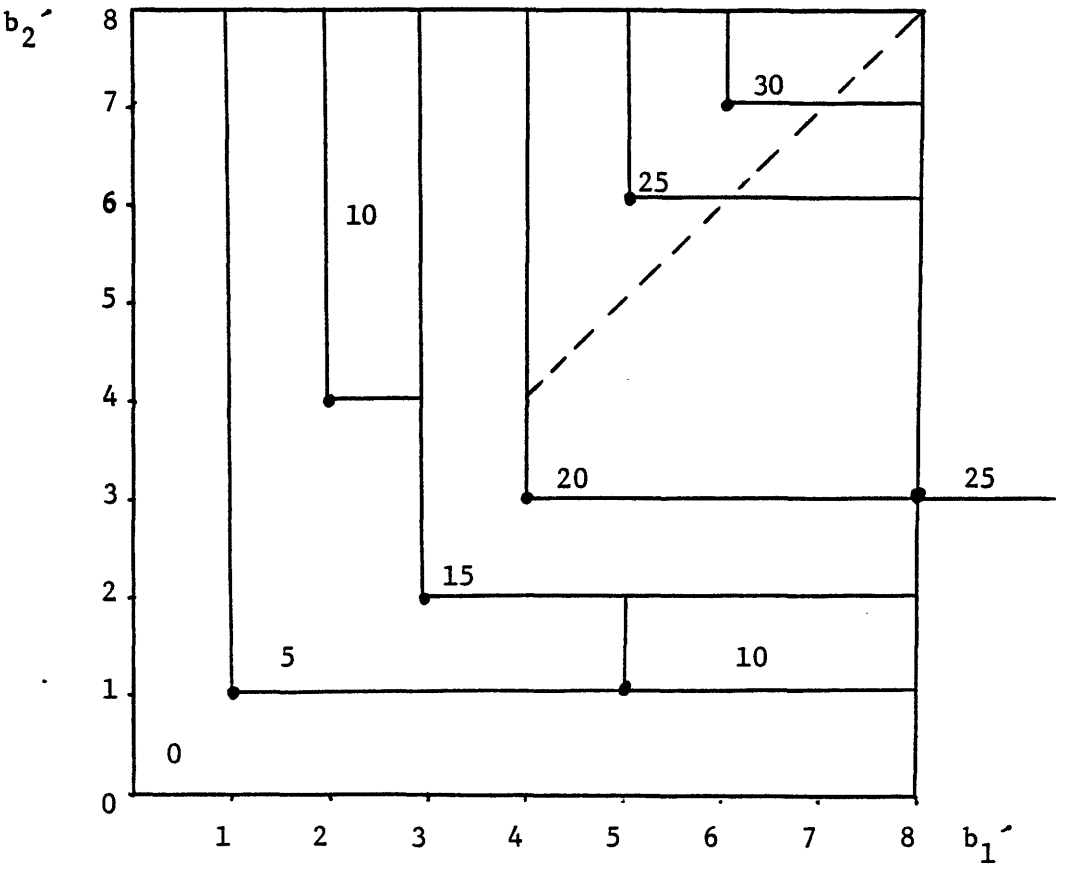


Figure 3. The optimal return function $f(b')$.

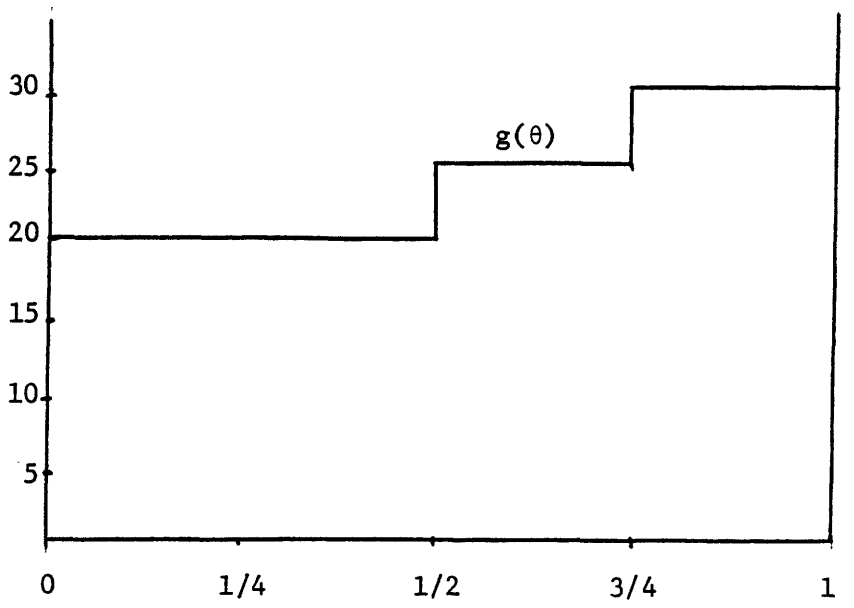


Figure 4. The parametric function $g(\theta)$.

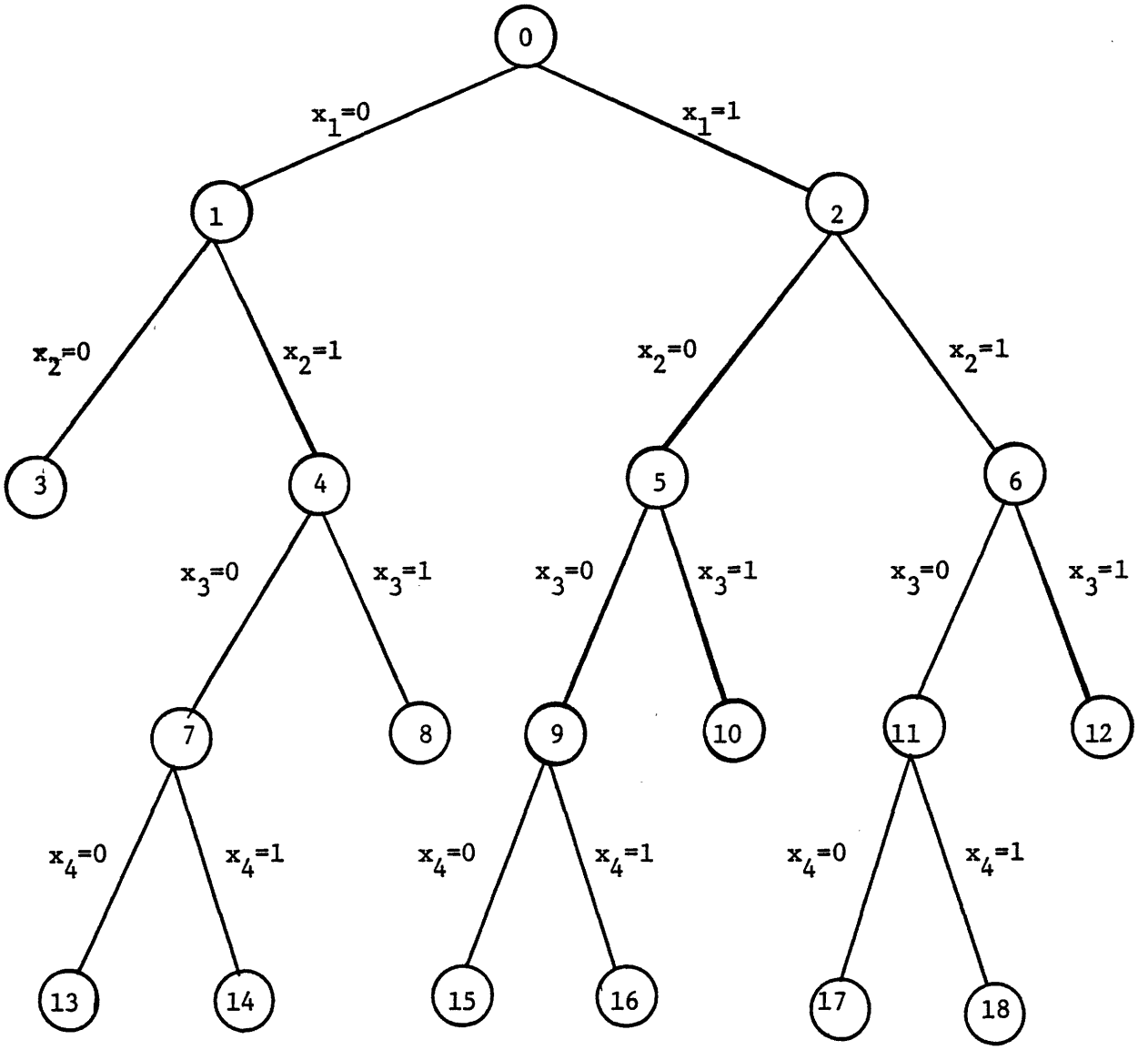


Figure 5. Branch-and-bound tree for the example.

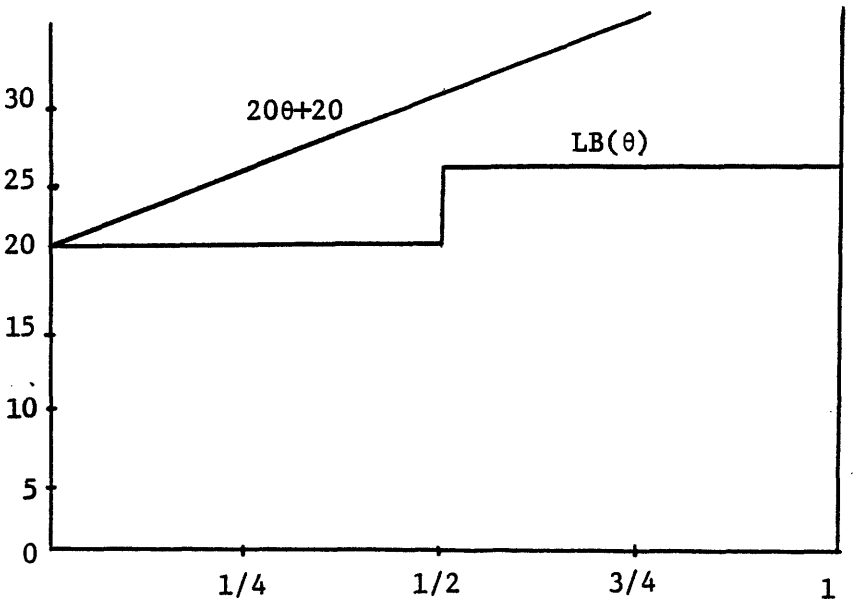


Figure 6. Bounding test for node 1.

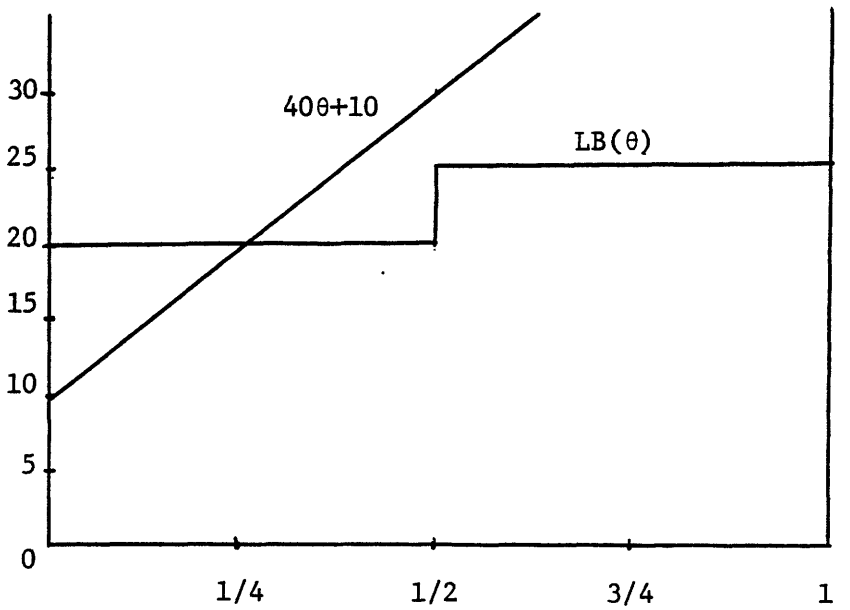


Figure 7. Bounding test for node 2.

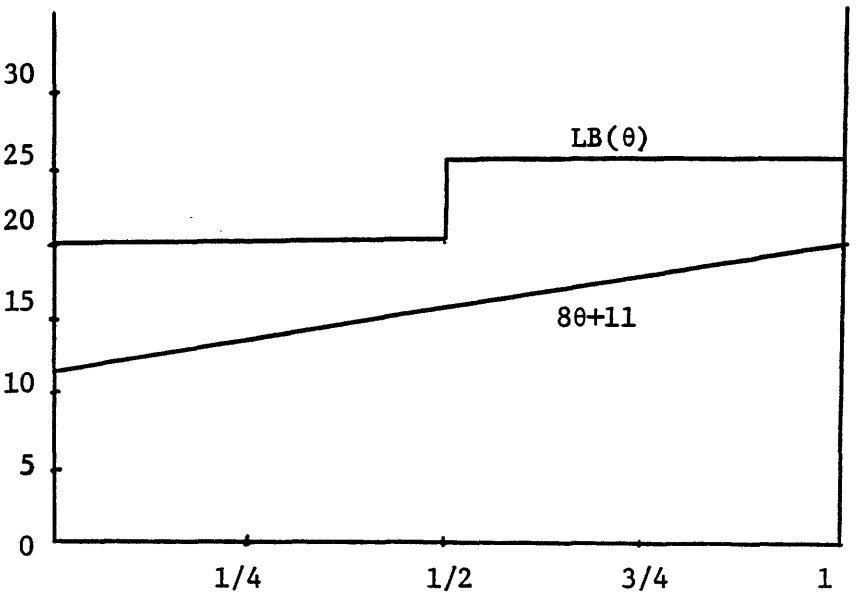


Figure 8. Bounding test for node 3.

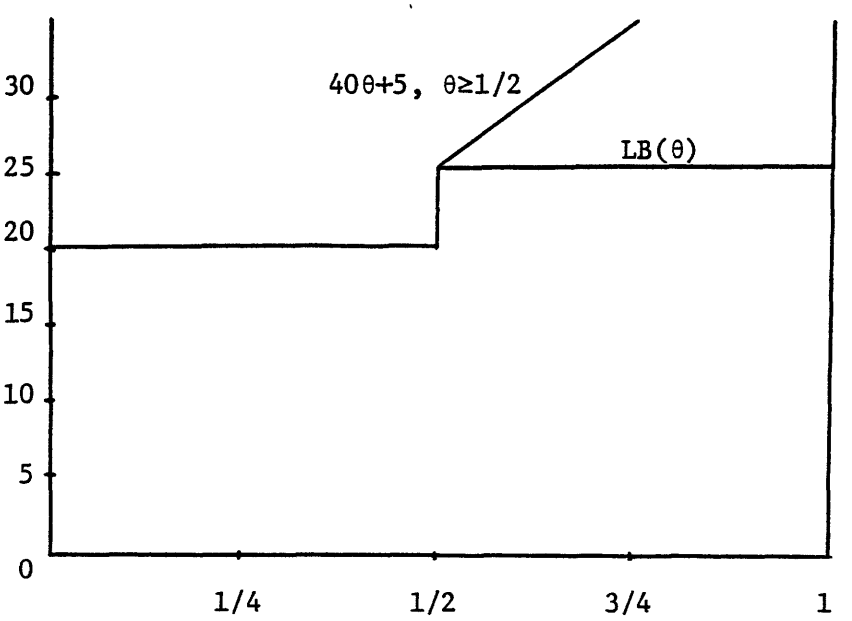


Figure 9. Bounding test for node 6.

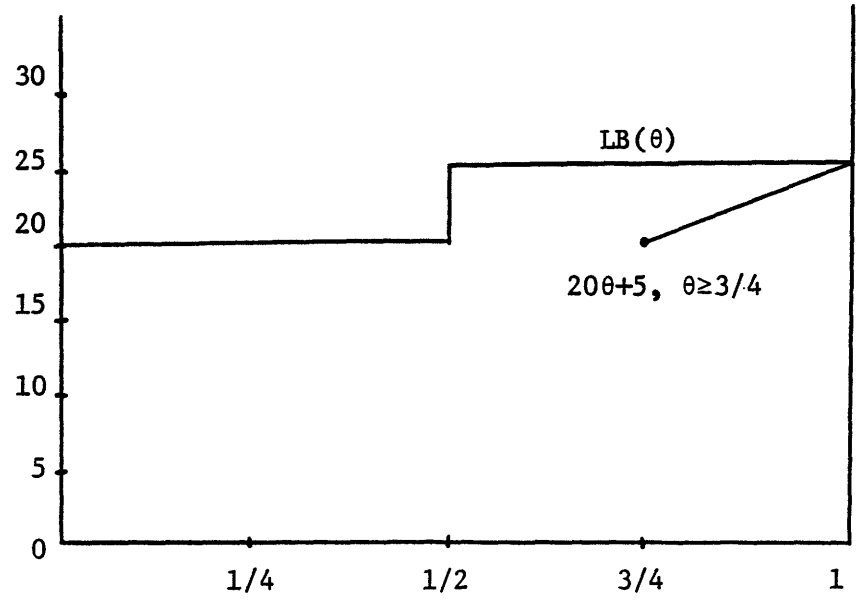


Figure 10. Bounding test for node 10.

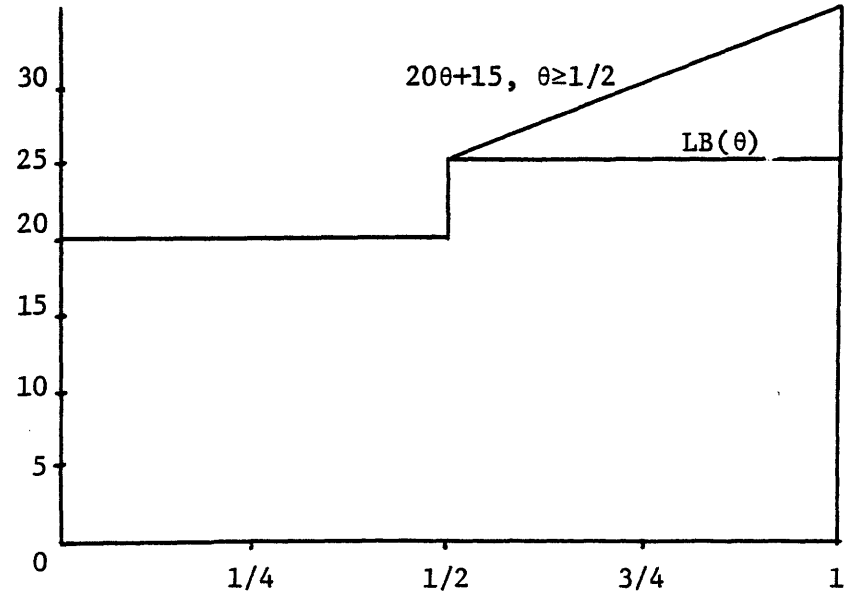


Figure 11. Bounding test for node 11.

7. Computational Results

The ideas presented above were tested by incorporating them into a branch-and-bound computer code [3]. The results for four test problems are presented in Table 1. In each run the direction vector d was taken as some percentage of the right-hand-side b. For example, if d=5%b, then (PIP) has right-hand-sides $b+\theta(.05)b$ for $0\leq\theta\leq 1$. The column headed "solutions" gives the number of optimal solutions found, or equivalently the number of steps of the $g(\theta)$ function. "Heuristic" is the number of (evenly spaced) θ values for which the heuristic is applied at Step 7. The problems are of the capital budgeting type and the heuristic employed is that of Toyoda [12]. "Pivots" is the total number of linear programming pivots and "time" is the total solution time in seconds on an IBM 370/168.

These results illustrate quite clearly how the computational burden increases as the interval of parameterization is lengthened. In order to facilitate comparison with our results by other researchers we have included the data for the 5×30 problem as Table 2 and the corresponding $g(\theta)$ function for a 15% increase in b as Table 3.

Table 1. Computation Results for four test problems.

m	n	d	solutions	heuristic	pivots	time
5	15	0	1	1	39	.239
		.05b	4	10	62	.541
		.10b	5	10	91	.815
		.15b	7	10	124	1.044
		.20b	8	10	130	1.170
		.25b	10	10	171	1.534
		.50b	16	20	315	3.162
5	30	0	1	1	153	1.605
		.05b	11	10	529	8.114
		.10b	28	20	1173	18.005
		.15b	37	20	2606	43.304
10	28	0	1	1	66	1.155
		.05b	16	10	173	4.469
		.10b	29	20	645	13.129
		.15b	42	20	1621	30.888
20	30	0	1	1	180	3.242
		.025b	6	5	400	9.486
		.05b	12	10	1350	32.185

Table 2. The 5×30 test problem.

a_{1j}	a_{2j}	a_{3j}	a_{4j}	a_{5j}	c_j
188	91	20	86	164	936
92	179	99	97	98	695
6	146	95	42	2	390
80	155	95	90	165	1152
91	102	84	101	140	980
44	112	136	3	106	1000
108	126	166	101	88	815
166	21	13	34	68	109
171	39	20	25	84	807
64	67	124	72	131	156
97	29	42	96	55	548
35	55	58	36	11	335
51	72	43	3	17	316
98	17	43	88	4	528
36	0	44	97	47	36
70	42	2	77	45	573
27	15	88	50	11	38
94	64	55	14	77	3
68	53	68	77	36	800
13	30	22	88	49	392
13.2	2.8	6.8	11.3	2.9	92
15.1	15.0	8.3	13.8	11.7	4
3.3	2.6	8.9	4.5	19.2	29
7.4	3.5	3.1	17.1	18.1	81
7.0	17.0	16.5	11.8	3.8	2
1.2	3.5	2.2	17.1	18.0	40
7.0	5.1	9.7	19.1	8.8	17
17.0	16.2	4.7	5.0	3.9	16
13.8	13.2	1.8	10.2	16.9	30
9.4	13.9	11.0	3.6	13.8	118
$b =$ 800	800	700	700	800	

Table 3. The g (θ) function for a 15% increase in b; 5 x 30 problem.

θ	g(θ)	θ	g(θ)	θ	g(θ)
0.0	7515	.41523	7846	.60166	8112
.01833	7578	.42000	7869	.62333	8141
.05524	7607	.43714	7891	.71083	8161
.10286	7612	.45667	7913	.73333	8171
.11250	7633	.45809	7931	.75809	8181
.11416	7696	.47833	7942	.77500	8204
.13583	7725	.49428	7947	.79333	8224
.20952	7777	.49809	7957	.82083	8253
.25238	7806	.51809	7994	.87916	8270
.30666	7807	.54190	8009	.93583	8283
.32750	7822	.56095	8023	.99416	8300
.34952	7836	.56285	8049		
.39333	7839	.59416	8060		

REFERENCES

1. Bowman, V.J., "The Structure of Integer Programs under the Hermitian Normal Form," Operations Research, Vol. 22 No. 5 (Sept-Oct), 1974, pp. 1067-1080.

2. Geoffrion, A.M. and Marsten, R.E., "Integer Programming Algorithms: A Framework and State-of-the-Art-Survey," Management Science, Vol. 18 (1972), pp. 465-491.

3. Marsten, R.E. and Morin, T.L., "A Hybrid Approach to Discrete Mathematical Programming," Sloan School of Management, MIT, Cambridge, Mass. July, 1975.

4. Morin, T.L. and Marsten, R.E., "An Algorithm for Nonlinear Knapsack Problems," Technical Report No. 95, Operations Research Center, Massachusetts Institute of Technology, Cambridge, Mass., May, 1974.

5. Morin, T.L. and Marsten, R.E., "Branch and Bound Strategies for Dynamic Programming," WP750-74, Sloan School of Management, MIT, Cambridge, Mass., November, 1974.

6. Nauss, R.M., "Parametric Integer Programming," Ph.D Dissertation. University of California, Los Angeles, January, 1975.

7. Noltemeier, H., "Sensitivitalsanalyse bei disketen linearen Optimierungsproblemen," in M. Beckman and H.P. Kunzi (eds), Lecture Notes in Operations Research and Mathematical Systems, #30, Sprinter-Verlag, New York, 1970.

8. Piper, C.J. and Zoltners, A.A., "Implicit Enumeration Based Algorithms for Postoptimizing Zero-One Problems," Management Sciences Research Report, No. 313, Graduate School of Industrial Administration, Carnegie-Mellon University, March, 1973.

9. Piper, C.J. and Zoltners, A.A., "Some Easy Postoptimality Analysis for Zero-One Programming", Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pa., 1975 (forthcoming in Management Science).

10. Roodman, G.M., "Postoptimality Analysis in Zero-One Programming by Implicit Enumeration," Naval Research Logistics Quarterly, Vol. 19, 1972, pp. 435-447.

11. Roodman, G.M., "Postoptimality Analysis in Integer Programming by Implicit Enumeration: The Mixed Integer Case," The Amos Tuck School of Business Administration, Dartmouth College, October, 1973.

12. Toyoda, Y., "A Simplified Algorithm for Obtaining Approximate Solutions to Zero-One Programming Problems," Management Science, Vol. 21, 1975 pp. 1417-1427.