COO-1493-10

CFSTI PRICES

H.C. \$ 3.00; MN 65

Department of Applied Mathematics and Computer Science
School of Engineering and Applied Science

Investigations in Integer Linear
Programming by Direct Search Methods

by

C. Drebes and L. Cooper

NOTICE

This report contains information of a preliminary nature and was prepared primarily for internal use at the originating installation. It is subject to revision or correction and therefore does not represent a final report. It is passed to the recipient in confidence and should not be abstracted or further disclosed without the approval of the originating installation or DTI Extension, Oak Ridge.

This research was performed under AEC
Research Contract No. AT(11-1)-1493.

Report No. COO-1493-10

RELEASED FOR ANNOUNCEMENT
IN NUCLEAR SCIENCE ABSTRACTS

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

I. Introduction

The present status of exact solution methods for integer linear programming problems is an unsatisfactory one. By the "integer linear programming problem" we mean the problem:

$$\begin{aligned} \text{Max } Z &= \sum_{j=1}^n c_j x_j \\ \text{subject to: } \sum_{j=1}^n a_{ij} x_j &\leq b_i \quad i=1, \dots, m \\ x_j &\geq 0 \\ x_j &\text{ are integers } j=1, \dots, n \end{aligned}$$

In matrix form this problem can be stated as:

$$\begin{aligned} \text{Max } Z &= \bar{c}'\bar{x} \\ \text{subject to: } A\bar{x} &\leq \bar{b} \\ \bar{x} &\geq \bar{0}, \text{ integer} \end{aligned}$$

where $\bar{c}, \bar{x}$ are n -vectors; $\bar{b}$ is an m -vector and A is an $m \times n$ matrix of real coefficients.

The unsatisfactory state of exact methods for the solution of the integer linear programming problem resides in the inability of the proposed exact methods to achieve the solution in a reasonable amount of time. A review of the current situation is to be found in [1]. Indeed, for certain problems, the principal algorithm, that of Gomory, seems to go on interminably.

To be able to provide for the needs of the considerable numbers of people in the industrial and military communities who have integer problems, some work was begun several years ago [2], [3] at Washington University under the principal investigator, to develop heuristic methods for the solution of the problem under discussion. The initial results were so encouraging that this work has been continued and is continuing. The results reported here are significant improvements in the development of heuristics, although much remains to be done.

II. General Description of the method.

The method is essentially a direct search technique, i.e., variables are manipulated one at a time, or a few at a time.

(See [4], [5])

A. Initialization - Phase 1.

A feasible solution is first obtained. In obtaining this feasible solution, three separate probes are attempted, using slightly different techniques, and the solution with the largest objective function value is chosen as the solution from which we will proceed. In each of the three probes just mentioned, each solution is maximized as much as possible through incrementing only one variable at a time until no further increase in Z can be obtained.

If this is the case, we have arrived at a solution which is sufficiently close to one or more constraints, such that increasing Z by changing the value of just one variable is no longer possible. We denote this optimization technique as Phase 1.

B. Stage I.

Here, we attempt to determine a better solution through incrementing all variables by one unit, and then decrementing certain variables, chosen deterministically. This amounts to violating the constraints and then attempting to return by a different path to an improved feasible solution. This technique is repeated until it fails to obtain a better solution. (Incrementing by two units is also tried in the event the former fails).

C. Stage II.

We now attempt to find an improved solution by reducing the size of the positive variables, and then incrementing certain variables chosen deterministically. Two passes are made through this stage. On the first pass, all positive variables are decreased by one unit. On the second pass, only some of the variables are decreased, where the variables are chosen deterministically and the reduction is one-half the maximum possible or else one unit if the maximum is one. Incrementation follows each set of reductions,

based also upon a deterministic technique which uses only a subset of the entire constraint set. This subset is chosen through a comparison of slack values to an adjusted average.

In this stage, we are attempting to find an improved solution by moving away from the constraints, remaining feasible, and then to proceed along a different path. (Stages I and II are alternately used until no better solutions are obtained).

D. Stage III.

This is equivalent to Echol's Phase II (see [2], [3]) technique with a slight extension. Two variables are allowed to change simultaneously, in the hope that the result will be an improved feasible solution. All combinations of the n variables, chosen two at a time, are analyzed until an improved solution occurs or until the set is exhausted. This stage is repeated until it fails to obtain an improved solution.

The extension mentioned above consists of retaining the solution corresponding to the minimum decrease determined in the event no increase was obtained. Stage III is then repeated using this solution as its starting point. If this also fails, we proceed to the next stage. (Stages II and III are alternately used until no better solutions are obtained.

E. Terminal Techniques.

Determine the variable with the maximum value. If this maximum is greater than one, several techniques are attempted, beginning with Stage IV. If the maximum value is one unit, we proceed to Stage VII, assuming a zero-one situation.

Stage IV.

This stage is similar to Stage II. Positive variables are decremented by 20%, if feasibility is retained. The "Phase 1" technique is then employed, followed by Stages II, III and I in that order, where improvement is now based on the value of Z corresponding to the decremented solution above. This stage is repeated until it fails to obtain an "improved" solution. Then we proceed to Stage V, with the best solution over all stages.

Stage V.

The variable with the maximum value is now decremented as much as possible. The "Phase 1" technique is now applied, but holding the variable just decremented, stationary. Stages I, III and II are then employed, in that order. If an improved solution results, we proceed to Stage VI, otherwise we terminate the method. If we were not able to decrement the variable with the maximum value, go to VI.

Stage VI.

This stage combines some of the techniques in Stages IV and V, however it uses an altered Phase 1 technique where the increment step size is reduced to one-third. First, all positive variables are reduced by 20%, where possible. Now we apply the Phase 1 technique where we again hold the variable which originally had the maximum value, fixed. Stages I, III and II are then applied in that order. This stage is repeated until no better solution is obtained, at which time the method is terminated.

Stage VII (zero-one assumption)

Each variable is reduced, one at a time, to zero, followed by a Phase 1 technique, where the variable just decremented is held at zero. Stage III is also employed after each Phase 1.

III. Detailed Description of the Method.

Throughout, the slacks, $s_i, i=1, \dots, m$ are determined by

$$s_i = \sum_{j=1}^n a_{ij}x_j - b_i .$$

Thus $s_i \leq 0$ implies the i^{th} constraint is satisfied, whereas $s_i > 0$ implies infeasibility.

A. Phase 1.

Two slightly different approaches have been incorporated into the method. Both require feasibility at all times.

Phase 1A. ("PHAZ1" subroutine)

(1). For each variable x_j such that $|c_j| > 0$, determine the maximum integer value, Δx_j , it can be incremented to obtain an increase, ΔZ_j , in the objective function Z :

$$(a). \text{ If } c_j > 0, \Delta x_j = \min_i \left\{ \left\lfloor \frac{-s_i}{a_{ij}} \right\rfloor \mid a_{ij} > 0 \right\}.$$

$$(b). \text{ If } c_j < 0, -\Delta x_j = \min_i \left\{ \left\lfloor \frac{s_i}{a_{ij}} \right\rfloor \mid a_{ij} < 0 \right\}.$$

Then $Z_j = c_j \Delta x_j$

(2). Determine $\Delta Z_k = \max_j \{ \Delta Z_j \}$

(3). If $\Delta Z_k = 0$, the phase terminates.

If $\Delta Z_k > 0$, let

$$\Delta_k = \begin{cases} \left\lfloor \frac{\Delta x_k}{3} \right\rfloor, & \text{if } \Delta x_k \leq -3 \\ -1, & \text{if } -3 < \Delta x_k < 0 \\ 1, & \text{if } 0 < \Delta x_k < 3 \\ \left\lfloor \frac{x_k}{3} \right\rfloor, & \text{if } x_k \geq 3 \end{cases}$$

(4). Increment x_k and Z :

$$\hat{x}_k = x_k + \Delta_k$$

$$\hat{Z} = Z + c_k \cdot \Delta_k$$

(5). Return to step (1), using the reverse order of inspection for the variables.

Phase 1B ("PHAZB" subroutine)

(1). For each variable x_j such that $|c_j| > 0$, determine the maximum integer value, Δx_j , it can be incremented to obtain an increase, ΔZ_j , in the objective function Z . (This is accomplished in identically the same manner as Phase 1A above.)

If $\Delta Z_j > 0$, also determine Δ_j by rule c or d, depending on the parameter IFFS:

(c). IFFS = 2

$$\Delta_j = \begin{cases} \Delta x_j & , \text{ if } |\Delta x_j| = 1 \\ \left\lfloor \frac{\Delta x_j}{2} \right\rfloor & , \text{ if } \Delta x_j > 1 \\ \left\lceil \frac{\Delta x_j}{2} \right\rceil & , \text{ if } \Delta x_j < 1 \end{cases}$$

(d). IFFS = 3

$$\Delta_j = \begin{cases} \left\lfloor \frac{\Delta x_j}{3} \right\rfloor & , \text{ if } \Delta x_j \leq -3 \\ -1 & , \text{ if } -3 < \Delta x_j < 0 \\ 1 & , \text{ if } 0 < \Delta x_j < 3 \\ \left\lceil \frac{\Delta x_j}{3} \right\rceil & , \text{ if } \Delta x_j \geq 3 \end{cases}$$

Also determine $d_{ij} = s_i + a_{ij}\Delta_j$ for all i , and determine $d_j = \max_i \{d_{ij}\}$; ($d_j \leq 0$)

(2). Determine $D_k = \min_j \{d_j | \Delta Z_j > 0\}$.

(3). Increment x_k and Z :

$$\hat{x}_k = x_k + \Delta_k$$

$$\hat{Z} = Z + c_k \cdot \Delta_k$$

(4). Return to step (1) using the reverse order of inspection for the variables. Termination of this phase occurs when all $\Delta Z_j = 0$.

B. Stage I. ("REVAMP" subroutine)

(1). Let $Q_i = \{j | a_{ij} < 0\}$, $P_i = \{j | a_{ij} > 0\}$ for $i = 1, \dots, m$.

Let $N[Q_i]$ and $N[P_i]$ represent the number of elements in Q_i and P_i respectively. Now determine

$$COL(i) = \begin{cases} \frac{\sum_{j \in P_i} a_{ij}}{N[P_i]}, & \text{if } N[P_i] > 0 \\ \frac{\sum_{j \in Q_i} |a_{ij}|}{N[Q_i]}, & \text{if } N[P_i] = 0 \end{cases}$$

(2). Increment all variables by one: $\hat{x}_j = x_j + 1$.

(3). Determine slacks s_i , $i = 1, \dots, m$. If all $s_i \leq 0$, go to step (8). If any $s_i > 0$, go to step (4).

(4). Determine: $R_I = \frac{s_I}{C\emptyset L(I)} = \max_i \left\{ \frac{s_i}{C\emptyset L(i)} \mid s_i > 0 \right\}$ where I is the number of the constraint producing the maximum ratio.

(5). For each variable x_j such that $a_{Ij} \neq 0$, determine the following:

(a). If $a_{Ij} > 0$ and $x_j > 0$, proceed to (i), (ii) or (iii), based on the value of c_j :

(i) If $c_j > 0$, determine

$$r_{1j} = \max \left\{ \left\lfloor |a_{Ij}/s_I| \right\rfloor, 1 \right\} \text{ and } r_{2j} = a_{Ij}/c_j.$$

Compute $r_j = r_{1j} + r_{2j}$, and return to step (5).

(ii) If $c_j = 0$, determine

$$t_j = a_{Ij}/s_I + |a_{Ij}|, \text{ and go to (iiii)}$$

(iii) If $c_j < 0$, determine

$$t_j = -a_{Ij}/c_j, \text{ and go to (iiii)}.$$

(iiii) If for all $s_i < 0$,

$$\left\{ \begin{array}{ll} s_i \leq a_{ij} & \text{if } a_{Ij} > 0 \\ \text{or} & \\ s_i \leq -a_{ij} & \text{if } a_{Ij} < 0 \end{array} \right.$$

and $|a_{ij} + a_{Ij}| = |a_{ij}| + |a_{Ij}|$ for all i such that $s_i \geq 0$,

x_j will be a candidate for the variable to be decremented.

Return to step (5).

(b). If $a_{Ij} < 0$, proceed to step (i), (ii) or (iii) based on the value of c_j :

(i) If $c_j < 0$, go to (i) in (a).

(ii) If $c_j = 0$, go to (ii) in (a).

(iii) If $c_j > 0$, go to (iii) in (a).

(6). If a variable passes the criteria in 5a (iiii), determine $t_k = \min \{t_j | x_j \text{ passes (iiii)}\}$. If no x_j passes (iiii), determine $r_k = \min \{r_j | x_j \text{ passes (i)}\}$. If both sets are empty, go to step (8). Otherwise, go to step (7).

(7). Let $\Delta_k = \begin{cases} 1, & \text{if } a_{Ik} > 0 \\ -1, & \text{if } a_{Ik} < 0 \end{cases}$.

and decrement x_k : $\hat{x}_k = x_k - \Delta_k$. Return to step (3).

(8). Compute the value of Z for the current feasible solution, and apply Phase 1B. If an improved solution has been obtained, return to step (2) and repeat the procedure. (If twenty or more successive improvements are obtained, begin incrementing variables by ten units instead of one.) If no improvement is obtained, increment each positive variable by two units and return to step (3). If incrementing by two does not obtain an improved solution, we terminate this stage and proceed to the next stage with the best solution obtained thus far.

C. Stage II ("ADJUST" subroutine)

$$\text{Let AVC} = \frac{\sum_{j=1}^n c_j}{n} \quad (\text{average cost})$$

- (1). Determine cost ratios $CR(j)$

$$CR(j) = \begin{cases} \frac{0.01}{AVC} , & \text{if } c_j = 0 \\ \frac{c_j}{AVC} , & \text{if } c_j \neq 0 \end{cases}$$

- (2). Decrement positive variables by one if the reduction is feasible:

$$\hat{x}_j = x_j - 1 \quad \text{if } x_j > 0.$$

- (3) Determine the new slack values and the average slack:

$$AVS = \frac{\sum_{i=1}^m s_i}{n} .$$

Then compute:

$$AVSF = AVFAC \cdot AVS$$

where AVFAC is a factor of 0.5, 1 or 2.

- (4). Determine the set M of constraints

$$M = \{i | s_i > AVSF\}$$

and $DMIN = \min \{s_i | i \in M\}$. Let $AVSF = DMIN \cdot 1.05$ and redetermine

$$M = \{i | s_i > AVSF\} .$$

(5). If the number of constraints in M , $N[M] = 0$, determine

$$D_{MAX} = \max_i \{s_i\} \quad \text{and}$$

let $AVSF = D_{MAX} * 1.1$ and redetermine $M = \{i | s_i > AVSF\}$.

(6). For each variable x_j , determine

$$u_j = \sum_{i \in U_j} a_{ij} \quad \text{where } U_j = \{i | i \in M, a_{ij} > 0, s_i < 0\}$$

$$\text{and } v_j = - \sum_{i \in V_j} a_{ij} \quad \text{where } V_j = \{i | i \in M, a_{ij} > 0, s_i = 0\}$$

$$\text{Also determine } C\emptyset L(j) = \left| \frac{u_j + v_j}{CR(j)} \right|$$

(7). Determine $C\emptyset L(k) = \min_j \{C\emptyset L(j)\}$, and x_k is to be incremented.

$$(8). \text{ Determine: } \Delta x = \min_i \left\{ -\frac{s_i}{a_{ik}} \mid a_{ik} > 0 \right\},$$

$$\text{and } \Delta x_k = \lfloor \Delta x \rfloor \quad (\text{If } \Delta x_k = 0, \text{ go to (9)})$$

$$\text{and } \Delta_k = \begin{cases} \left\lfloor \frac{\Delta x_k}{2} \right\rfloor, & \text{if } \Delta x_k > 1 \\ 1, & \text{if } \Delta x_k = 1 \end{cases}.$$

Let $\hat{x}_k = x_k + \Delta_k$ and return to step (3).

(9). Compute Z for the current solution and apply Phase 1B. If an improved solution has been obtained, return to step (2) and repeat the procedure. If no improvement is obtained, go to step (10).

(10). Let N represent the number of positive variables in the best solution. Let $ICUT = \left\lfloor \frac{N}{3} \right\rfloor + 1$. The following technique is designed to reduce $ICUT$ of the positive variables.

(11). Repeat steps 3, 4 and 5, producing set M .

(12). For each variable $x_j > 0$, determine

$$SUM(j) = \sum_{i \in M} A_{ij}$$

$$\text{and } C\emptyset L(j) = SUM(j) + \frac{SUM(j)}{CR(j)}$$

(13). Determine $C\emptyset L(k) = \max_j \{C\emptyset L(j) | x_j > 0\}$

(14). Determine the maximum amount that variable x_k can be reduced:

$$\Delta x_k = \min_i \left\{ \left\lfloor \frac{s_i}{a_{ik}} \right\rfloor \mid a_{ik} < 0 \right\}$$

If no $a_{ik} < 0$, or $\Delta x_k > x_k$, set $\Delta_k = x_k$. Otherwise let $\Delta_k = \Delta x_k$.

If $\Delta_k = 0$, eliminate x_k from consideration and return to step (13).

Otherwise go to step (15).

(15). Let $\hat{x}_k = x_k - \Delta_k$ and repeat steps (11) through (15) until either $ICUT$ of the positive variables have been reduced or else no more variables can be reduced and retain feasibility.

Then employ steps (3) through (9), incrementing until no more variables can be changed without violating a constraint.

If an improved solution is obtained, return to step (2) and repeat the process.

D. Stage III (PHAZ2 subroutine)

(1). Let $x_M = \max_j \{x_j\}$ and determine

$$\Delta x = \begin{cases} x_M, & \text{if } x_M \leq 25 \\ \left(\left\lfloor \frac{x_M}{10} \right\rfloor + 1 \right) \cdot 10, & \text{if } 25 < x_M < 100 \\ 100, & \text{if } x_M = 100 \\ \left(\left\lfloor \frac{x_M}{100} \right\rfloor + 1 \right) \cdot 100, & \text{if } x_M > 100 \end{cases}$$

(2). The variables are examined, one at a time, in the order of decreasing absolute cost value. Only variables with non-zero costs are incremented.

(a). If $c_k > 0$, let $\hat{x}_k = x_k + \Delta x$. If $c_k < 0$, and $x_k > 0$, set $\hat{x}_k = x_k - \Delta x$, if $\Delta x \leq x_k$. If $x > x_k$, set $\Delta x = x_k$, and $\hat{x}_k = 0$.

(b). Determine the new slacks, s_i , and determine

$$\Delta Z_k = |c_k| \cdot \Delta x.$$

If $s_i \leq 0$, for all i , go to step (2).

If $s_i > 0$, for some i , go to (c).

(c). Examine all variables x_j , $j \neq k$, one at a time, for a change that obtains an improved feasible solution.

(i). Let $T_p = \{i \mid s_i > 0\}$ and

$$T_o = \{i \mid s_i = 0\}.$$

If $x_r > 0$, x_r will be decremented only if

$$\begin{cases} a_{ir} > 0, & \text{for all } i \in T_p, \text{ and} \\ a_{ir} \geq 0, & \text{for all } i \in T_o. \end{cases}$$

If the above conditions are met, compute

$$\Delta x_r = \max \left\{ \frac{s_i}{a_{ir}} \mid a_{ir} > 0 \text{ for } i \in (T_p \cup T_o) \right\}$$

and $\Delta_r = \lceil \Delta x_r \rceil$

If $\Delta_r > x_r$, x_r cannot be changed to regain feasibility and we continue with (c), next examining x_{r+1} . Otherwise, set $\hat{x}_r = x_r - \Delta_r$ and proceed to step (d).

If $\begin{cases} a_{ir} < 0 & \text{for } i \in T_p \\ a_{ir} \leq 0 & \text{for } i \in T_o \end{cases} \text{ and}$

then x_r can be incremented. Determine

$$\Delta x_r = \max \left\{ -\frac{s_i}{a_{ir}} \mid a_{ir} < 0 \text{ for } i \in (T_p \cup T_o) \right\}$$

$\Delta_r = \lceil \Delta x_r \rceil$. Then set $\hat{x}_r = x_r + \Delta_r$ and proceed to step (d)

(d). Determine

$$\Delta Z_r = \begin{cases} c_r * \Delta_r & \text{if } a_{ir} \leq 0 \\ -c_r * \Delta_r & \text{if } a_{ir} \geq 0 \end{cases}$$

and $\Delta Z = \Delta Z_k + \Delta Z_r$.

(e). If $\Delta Z > 0$, an improved solution has been obtained. Return to step (1) and repeat the process, using the new solution.

If $\Delta Z \leq 0$, retain the solution corresponding to the smallest decrease so obtained. Then reset x_r to its original value and return to step (c) for the next variable x_{r+1} . If $r = n$, proceed to step (f).

(f). Reset x_k to its original value. Decrease Δx by the following rule:

$$\hat{\Delta x} = \begin{cases} \Delta x - 50, & \text{if } \Delta x > 100 \\ \Delta x - 10, & \text{if } 30 < \Delta x \leq 100 \\ \Delta x - 2, & \text{if } 2 < \Delta x \leq 30 \\ \Delta x - 1, & \text{if } \Delta x \leq 2 \end{cases}$$

If $\hat{\Delta x} > 0$, return to step (2) with $\Delta x = \hat{\Delta x}$ and repeat the process for x_k . If $\hat{\Delta x} = 0$, set Δx back to the value obtained in step (1) and return to step (2) to examine the next variable. If no more variables are to be examined, go to step (3).

(3). Using the solution retained in step 2e, apply "Phase 1" and return to step (1). The entire process is repeated just once, unless an improved solution is obtained, in which event we continue to repeat this stage until we fail to obtain an improvement.

E. Initialization ("INTLZ" subroutine)

(1). Set $x_j = 1$, $j = 1, \dots, n$.

(2). Determine slacks s_i . If $s_i \leq 0$ for all i , retain this solution and proceed to step (4).

If $s_i > 0$ for some i , determine the constraint k with maximum slack s_k , and proceed to step (3).

(3). For each $x_j > 0$, determine

$$a_{kr} = \max_j \{a_{kj}\}, \text{ and}$$

set $x_r = 0$. Return to step (2) and repeat the process. If no further reductions can be made, go to step (5).

(4). Apply "Phase 1A" to the solution obtained from (2) or (3) and store this solution. Call it solution A. Proceed to step (5).

(5) If $\bar{x} = \bar{0}$ is a feasible solution, apply "Phase 1B" and store this as solution B. Proceed to step (6) in either event.

(6). For $\bar{x} = 0$, whether feasible or not, apply Stage I just once (to the first improvement only) and store this solution as C. Go to step (7).

(7). Select the solution obtaining the maximum of Z from A, B and C and proceed with it to Stage I.

F. Coordination of Stages

(1). Initialization.

(2). Stage I.

(3). Stage II, with AVFAC = 0.5. If improvement is obtained go to (6), otherwise to (4).

(4). Stage III. If improvement is obtained, go to (10), otherwise to (5).

(5). Stage II, with AVFAC = 2.0. If improvement is obtained go to (8), otherwise to (11).

(6). Stage I. If improvement is obtained go to step (2), otherwise to (7).

(7). Stage II, with AVFAC = 2.0. If improvement is obtained, go to step (8), otherwise to (9).

(8). Stage I. If improvement is obtained, go to step (7), otherwise to (9).

(9). Stage III. If improvement is obtained, go to step (10), otherwise to (11).

(10). Stage II, with AVFAC = 0.5, 1.0, and 2.0. If improvement is obtained, go to step (9), otherwise to (11).

(11). If $\max_j \{x_j\} < 1$, go to step (12), other to (15).

(12). Stage IV. If $\max_j \{x_j\} < 50$, go to step (13), otherwise terminate the method.

(13). Stage V. If $x_M = \max_j \{x_j\}$ cannot be decremented, go to step (14). If it can be decremented, and an improvement results, go to step (14); otherwise terminate the method.

(14). Stage VI, followed by termination of method.

(15). Stage VII, followed by termination of method.

IV. Test Problems

Initially, thirty-seven problems ranging in size from a (2×2) to a (21×21) were tested. Problems 1 through 21 are from Robert Echols' master's thesis [2] and 22 through 37 are from Dr. Robert Cook's doctoral dissertation, [6] for which the optimal solutions are known. Optimal solutions were obtained by the heuristic method described above in all but three problems, #10, 19 and 37, with errors ranging from 0.001% to 0.95%. Table I contains the results with additional information of computer time, stages needed to reach the optimal or best solution and the results of Echols' heuristic applied to these problems.

Table I

No.	Size (m×n)	Optimal Soln.	Test Result	% Error	Min. of 360 Time	Stages Needed	Results of Echols Heuristic	Echols % Error
1	2×2	33	33	-	0.05	A	33	-
2	3×3	20	20	-	0.05	A,I	20	-
3	4×6	25	25	-	0.05	A,I		-
4	2×5	11	11	-	0.05	B,II(0.5)	11	-
5	3×6	15	15	-	0.05	B	15	-
6	4×6	25,000	25,000	-	0.05	A,I	25,000	-
7	8×9	1072.44	1072.44	-	4	C,I,II	1072.44	-
8	3×3	19	19	-	0.05	A	19	-
9	3×2	1	1	-	0.05	C	1	-
10	2×5	105	104	0.95%	0.05	A,I,III,VI	100	5%
11	5×10	156	156	-	0.05	B	156	-
12	4×5	7	7	-	0.05	A	7	-
13	6×6	9	9	-	0.05	A	9	-
14	2×3	-15	-15	-	0.05	C	-15	-
15	4×3	11	11	-	0.05	A,I	10	9%
16	4×4	3	3	-	0.05	A	3	-
17	3×3	-12	-12	-	0.05	C	-12	-
18	3×3	-52	-52	-	0.05	C,III	-52	-
19	7×9	107,097	107,096	0.001%	4	C,I,II(.5),III, IV	106,722	0.35%
20	27×21	540	540	-	2	C,II(0.5),VII	538	0.37%

Table 1
Continued

No.	Size (m×n)	Optimal Soln.	Test Result	% Error	Min. of 360 Time	Stages Needed	Results of Echols Heuristic	Echols % Error
21	10×12	17	17	-	0.5	A,I,II(2.0)	17	-
22	2×5	99	99	-	0.05	A,I	99	-
23	5×10	820	820	-	0.05	A,I	820	-
24	5×20	2401	2401	-	1	A,I	2401	-
25	7×15	1093	1093	-	0.5	A,I	1068	2.3%
26	7×15	3909	3909	-	1	A,I,II(0.5),III	3909	-
27	10×15	4522	4522	-	1	A,I,II(0.5)	4522	-
28	8×20	1204	1204	-	1	A,I,II(0.5),III,V	1163	3.4%
29	9×20	1327	1327	-	1	C,III	1258	5.2%
30	10×20	1264	1264	-	1	C,II(0.5)	1210	4.3%
31	10×20	4616	4616	-	2	B,I,II(0.5),III	4387	5.0%
32	8×25	3221	3221	-	1	A,I	3221	-
33	8×25	5367	5367	-	3	B,I,III,V	4983	7.1%
34	9×25	1774	1774	-	2	B,I,II(0.5),III	1762	0.7%
35	10×29	23,544	23,544	-	4	B,I,III	23,544	-
36	15×24	5002	5002	-	3	B,I,II(0.5),III	4671	6.6%
37	21×21	5153	5110	0.83%	4	A,I,V	5096	1.1%

In addition to the above set of problems, a program was written which constructs a linear programming problem whose optimal solution is integer. Five such problems were constructed of size 25x50, and the heuristic applied. The results are in Table II.

Table II

No.	Size	Optimal Soln.	Test Result	% Error	Min. 360 Run Time	Stages Needed
1	25x50	3928	3832	2.5%	29	C,I,II(0.5,2.0),III
2	25x50	4794	4672	2.5%	44	A,I,II(0.5),V,VI
3	25x50	4257	4178	1.9%	45	B,I,III,II(1.0),III,IV
4	25x50	5721	5590	2.3%	60	B,I,II(0.5,2.0),IV
5	25x50	5105	5015	1.8%	44	B,I,II(0.5),III,II(1.0,2.0), III,II(1.0)

V. Description of the Integer L.P. Problem Constructor

(1). A coefficient matrix A , and costs c_j , $j = 1, \dots, n$ are randomly generated:

$$(0 \leq a_{ij} \leq 10) \quad , \quad (1 \leq c_j \leq 100).$$

(2). A vector $\bar{x}^*$ is generated randomly, containing integer components, with $N_p < M$ positive elements. $(0 \leq x_j \leq 10)$.

(3). The right hand side vector $\bar{b}$ is determined from $\bar{b} = A\bar{x}^*$.

(4). $(M - N_p)$ b_i elements are then increased by random amounts up to 50%, such that slack variables $x_{si} > 0$ for $(M - N_p)$ constraints.

(5). A modification of Phase 1 of revised simplex is then applied. The initial basis consists entirely of slack vectors. All vectors corresponding to $x_j^* > 0$ are then put in the basis, one at a time, replacing a slack vector. If all $Z_j - c_j \geq 0$, $j = 1, \dots, N$, then $\bar{x}^*$ is the optimal solution (all integer). If $Z_j - c_j < 0$ for some j , go to step (6).

(6). If $Z_j - c_j \geq 0$ for all slack vectors, then the costs of the non-slack vectors, not in the basis, can simply be reduced by $Z_j - c_j$ if $Z_j - c_j < 0$ and $\bar{x}^*$ will be the optimal solution. If $Z_j - c_j < 0$ for some slack vector, go to step (7).

(7). Consider the simplex tableau from step (5). Let P = total number of slacks not in the basis. Let Y^t represent the submatrix

of N_p rows and P columns, where the N_p rows correspond to the non-slack vectors in the basis and the P columns the slack vectors not in the basis. Consider the transpose of Y^t , Y , a $(P \times N_p)$ matrix, and let $\bar{d}$ represent the vector of z_j values corresponding to the slacks not in the basis taken in the order of increasing j . Let $\bar{h} = \bar{d} - (0.5)\bar{l}$. Now consider the problem:

$$\min Z = \bar{l} \cdot \bar{f} \quad \text{subject to}$$

$$Y\bar{f} \geq -\bar{h}$$

$$f_j \geq 0 \quad \text{all } j$$

where the f_j represent the amount the costs of variables in the basis of our original problem must be incremented to obtain $z_j \geq 0$ for slacks not in the basis. Solving the above problem for $\bar{f}$, we increment the costs accordingly and go to step (5).

Thus the following problem is obtained: $\max Z = \bar{c} \bar{x}$, subject to the constraints $A\bar{x} \leq \bar{b}$, $\bar{x} \geq 0$, whose optimal solution $\bar{x}^*$ has integer components only. (Slack variables may be non-integer).

The parameters needed for input to this program are

ICØN: number of constraints

IVAR: number of variables

INITL: initial value for the random number generator

DENS: matrix density desired

RNEGF: frequency of negative matrix elements

CNEGF: frequency of negative cost elements

VARF: determines the magnitude of values in the solution vector $\bar{x}^*$

FAC: determines the magnitude of values in the A matrix

CØSTF: determines the magnitude of values in the cost vector

RATIO: determines the number of positive elements in $\bar{x}^*$

VI. Discussion of Test Results

The results for the problems in Table I are very encouraging. Besides increased efficiency over Echols' method, the increment size in stage III is not required as input, but is determined through the method. Also, initial feasible solutions were not required for input in the event $\bar{x} = \bar{0}$ was not feasible, but were obtained in the initialization stage.

The computer times in Tables I and II are approximate. The errors could range up to 0.2 minutes, depending on the size of the problem. The computer time required is approximately the same as that for Echols' method; however in most of the problems, the optimal solution was obtained sooner than it was from Echols' method. This is evidenced by noting from Table I the stages needed to obtain optimality. In 10 of Echols' test problems (#1-21), our method achieved the optimal solution in the initialization stages A, B or C, which require very little time to implement. Using Echols' method, optimality was achieved without his Phase II technique in only 3 problems. Further, we obtained optimality in 17 of the 21 problems before Stage III (the equivalent to Echols' Phase II) was ever implemented. This seems to be worth noting, since stage III is very time consuming in comparison to the other stages. Generally, it seems that for those problems in which our method achieved optimality, the time required to achieve the optimal solution is at least less than half the time indicated in Table I.

The problems in Table II present a very severe test of our method, due to the optimal integer solution occurring as the extreme point solution to the problem without integer restrictions on the variables. This is apparent from the table in both the error and computer time required. In such problems, it appears that the closer a solution approaches optimality, the greater the number of variables that must be simultaneously changed in value in order to obtain a better solution. Table III illustrates the above.

VII. Discussion of the Method

The objective of the direct search method applied to the integer problem is to reduce the time required to obtain an optimal or near optimal solution. Deterministic algorithms already exist which will obtain an optimal solution; however the computer time required to achieve optimality is quite large. There is a need for an efficient heuristic which significantly reduces the run time and obtains a solution with a low degree of error.

It appears that the method presented is a step in the right direction. The results of Table II indicate that the method needs improvement, as both the error and run time are much too large. It would appear to be desirable to eliminate stage III, which is too time consuming. Stages I and II, which appear to be quite helpful, could stand further analysis.

Table III

No.	Size	Soln. from Init. Stage	Soln. After Stage III Used for First Time	Approximate Comp. Time	% Error	Final Result	Total Min Run Time	% Error	% Error Decrease	Additional Time
1	25×50	2671	3832	10	2.5	3832	29	2.5	0.0	19
2	25×50	3050	4630	10	3.4	4672	44	2.5	0.9	34
3	25×50	2772	4129	10	3.0	4178	45	1.9	1.1	35
4	25×50	4644	5498	15	3.9	5590	60	2.3	1.6	45
5	25×50	3149	4849	10	5.0	5015	44	1.8	3.2	34

In stage I, after each variable has been incremented, a single constraint is chosen which is used to help determine the variable to be changed in order to approach feasibility. Variables with non-zero coefficients in this constraint will be the only candidates for change at that point. The method of choosing this variable and the size of its change in value need closer scrutiny. As the stage now exists, the constraint is chosen from the set of constraints having an infeasibility. Intuitively, it is chosen from this set as the one in which the rate of change in the infeasibility per unit change in a variable is at a minimum. Variables with non-zero coefficients in this constraint form the set from which we select the variable to be changed in value. Intuitively, the variable is chosen as the one whose combined sum of rate of change of infeasibility and rate of change of the cost per unit change in the variable is at a minimum.

In stage II, after variables are decremented, we must determine which variables are to be incremented and the size of the increment. In this, we use a subset of the total constraint set, having the smallest infeasibilities. The parameter AVFAC, together with the arithmetic mean of slack values are used to determine this subset. Intuitively, the variable is chosen as the one whose ratio of the coefficient sum to its cost, is at a minimum. The size of the increment is chosen as half the maximum feasible or else one. Thus we increment the variables whose average rate of change of cost with respect to that variable is at a maximum, for the subset of constraints chosen.

It seems that the optimal solution to the problem without integer requirements on the solution could be valuable in making the choices in stages I and II. Hopefully, such an inclusion could result in increased efficiency and reduced running time, especially in the set of (25×50) problems. This LP solution could be used as a guide as to which variables are to be changed in value. This incorporation with a deletion of stage III could prove interesting.

Further, the initial solution choice seems critical. This was evidenced in the development of the method when at one time, the solution obtained from Phase 1A alone was used as a starting solution. In quite a few problems, optimality could not be achieved from it. Hence, this requires further investigation.

APPENDIX I

INTEGER LINEAR PROGRAMMING HEURISTIC

```
/JOB 99 B626,DREBES,MULTIPLE,GO,MAP
/FTC LIST,REF,SIZE=256K
      DIMENSION X(50),XS(50),DT(50),C(50),B(50),A(50,50),
1COL(50),IMSG(13),ICOST(50),IOICOST(50),IPDT(50),IODT(50
X),XSS(50)
2,XL(50),XR(50),ISKIP(50)
      COMMON ICON,IVAR,Z,A,B,C,DT,X,XS,DELZ,COL,FRACT,DELX,
1IDELX,ZMAX,XDELX,XINCR,XXINC,ZS,ISW1,ISW2,JHAV,I1,I2,
2IFFS,IPROB,KPHASE,INP,INO,KTR,KTRO,ICOST,IOICOST,IPDT,I
XODT,ITRY
3,KHOLD,XSS,ZL,XL,ISKIP,JEND,AVFAC
      READ (5,1000) I1,I2
1000 FORMAT (2I2)
      8 READ (I1,9) ICON,IVAR,IPROB,XXINC,IFFS,ISTAX,ZS
9 FORMAT (3I5,F5.0,2I5,F10.0)
      IF(ICON-9999)2,80,2
2 IF(IFFS)5,5,3
3 READ (I1,4)(IMSG(I),I=1,13)
4 FORMAT(13A5)
5 READ (I1,20)(X(J),J=1,IVAR)
      WRITX=0.
      KHOLD=0
      ITRY=0
      IPROB=IPROB
      DO 10 I=1,ICON
10 READ (I1,20)(A(I,J),J=1,IVAR)
      READ (I1,20)(B(I),I=1,ICON)
      READ (I1,20)(C(J),J=1,IVAR)
20 FORMAT(16F5.0)
      WRITE (I2,12) IPROB
12 FORMAT(1H1,20X,14HPROBLEM NUMBER,I5)
      IF(IFFS)15,15,13
13 WRITE (I2,14)(IMSG(I),I=1,13)
14 FORMAT(1H013A5)
15 WRITE (I2,30) ICON,IVAR,XXINC
30 FORMAT(1H0,10X,2HI=I5,5X,2HJ=I5,5X,11HINCREMENT =F5.0)
21 IF(ISTAX)31,31,25
25 ISTAX=ISTAX-1
26 WRITE (I2,27)(X(J),J=1,IVAR)
27 FORMAT(1H0, 17HSTARTING X VECTOR(/ 7F9.1))
      IF(WRITX)31,31,45
31 WRITE (I2,34)
34 FORMAT(1H0 8HA MATRIX/1H 3HROW)
      DO 35 I=1,ICON
35 WRITE (I2,50) I,(A(I,J),J=1,IVAR)
50 FORMAT(1H I2,F7.2,6F9.2,/(1X7F9.2))
      WRITE (I2,36)
36 FORMAT(1H0, 8HB VECTOR)
```

```
WRITE (I2,40)(B(I),I=1,ICON)
40 FORMAT(1H 7F9.2)
WRITE (I2,37)
37 FORMAT(1H0, 8HC VECTOR)
WRITE (I2,40)(C(J),J=1,IVAR)
45 CGN=ICON
FRACT=0.
IFFS=2
DO 60 J=1,IVAR
60 FRACT=FRACT+C(J)
FRACT=FRACT/CON
ITERM=0
KTR=0
KTRO=0
DO 800 J=1,IVAR
ICOST(J)=0
800 COL(J)=0.0
805 SC=0.0
JEND=0
DO 820 J=1,IVAR
IF (COL(J)) 810,810,820
810 IF (C(J)) 812,811,812
811 COL(J)=1.
KTRO=KTRO+1
ICOST(KTRO)=J
GO TO 820
812 IF (ABS(C(J))-SC) 820,820,814
814 SC=ABS(C(J))
JEND=1
JSAV=J
820 CONTINUE
IF (JEND) 830,830,825
825 KTR=KTR+1
ICOST(KTR)=JSAV
COL(JSAV)=1.0
GO TO 805
830 IT=0
KPHASE=0
ZMAX=-1000000.
IF (ZS) 831,373,831
373 Z=0.0
DO 375 J=1,IVAR
375 Z=Z+C(J)*X(J)
ZMAX=Z
ZSAV=4630.
JSAV=12
GO TO 969
831 IT=IT+1
IF (KPHASE-1) 834,832,834
832 DO 833 J=1,IVAR
833 XR(J)=X(J)
ZSAV=Z
834 DO 580 J=1,IVAR
```

```
580 X(J)=0.0
    Z=0.0
    GO TO (120,130,150),IT
120 CALL INTLZ
    IF (KPHASE-1) 831,122,122
122 CALL PHAZ1
    GO TO 831
130 CALL PHAZB
    GO TO 831
150 IF (ZMAX+1000000.) 170,170,160
160 DO 165 J=1,IVAR
165 XS(J)=XR(J)
    ZL=ZSAV
    WRITE (I2,167) ZMAX
167 FORMAT (1H0,3X,7HZMAX = ,F12.2)
    GO TO 180
170 ISW1=100
    ZSAV=-0.1
180 JEND=0
    ISW1=100
840 CALL REVAMP
841 ZSAV=Z
    JEND=0
    AVFAC=0.5
    CALL ADJUST
    IF (Z-ZSAV-0.05) 643,643,842
643 CALL PHAZ2
    IF (Z-ZSAV-0.05) 846,846,851
846 AVFAC=2.0
    CALL ADJUST
    ZTEM=ZMAX
    IF (Z-ZSAV-0.05) 83,83,844
842 ZSAV=Z
    CALL REVAMP
    IF (Z-ZSAV-0.05) 843,843,841
843 AVFAC=2.0
    CALL ADJUST
    IF (Z-ZSAV-0.05) 850,850,844
844 ZSAV=Z
    CALL REVAMP
    IF (Z-ZSAV-0.05) 850,850,843
850 CALL PHAZ2
    IF (Z-ZSAV-0.05) 83,83,851
851 ZSAV=Z
    JEND=1
    AVFAC=0.5
    CALL ADJUST
    IF (Z-ZSAV-0.05) 83,83,852
852 ZSAV=Z
    GO TO 850
83 WRITE (I2,81) Z,(J,X(J),J=1,IVAR)
81 FORMAT (1H0,20X,9HMAX VALUE,/,21X,2HZ=,F10.3,/(21X,
11HX,I2,1H=,F10.3))
```

```
ZTEM=ZMAX
DO 905 J=1,IVAR
905 XR(J)=X(J)
ZSAV=Z
ZIP=Z
XBIG=0.0
DO 93 J=1,IVAR
IF (X(J)-XBIG) 93,93,91
91 XBIG=X(J)
JSAV=J
93 CONTINUE
IF (XBIG-1.0) 710,710,927
710 WRITE (I2,711)
711 FORMAT (1H0,8HZERO-ONE)
DO 750 J=1,IVAR
DO 715 K=1,IVAR
715 X(K)=XR(K)
IF (X(J)) 750,750,712
712 Z=ZSAV
KHOLD=J
Z=Z-X(J)*C(J)
X(J)=0.0
CALL PHAZB
CALL PHAZ2
IF (Z-ZIP-0.05) 750,750,720
720 DO 725 K=1,IVAR
725 XL(K)=X(K)
ZIP=Z
WRITE (I2,81) Z,(I,X(I),I=1,IVAR)
750 CONTINUE
KHOLD=0
GO TO 8
927 DO 930 J=1,IVAR
IX=(4.0*XR(J))/5.0
X(J)=IX
CALL POSDT
IF (INP) 930,930,928
928 X(J)=XR(J)
930 CONTINUE
WRITE (I2,926)
926 FORMAT (1H0,21HCUT POS VAR BY 20 PCT)
Z=0.0
DO 935 J=1,IVAR
935 Z=Z+X(J)*C(J)
ZMAX=Z
CALL PHAZ1
IF (ABS(ZMAX-ZTEM)-0.05) 960,960,936
936 JEND=0
AVFAC=0.5
CALL ADJUST
ZIP=ZMAX
IF (ABS(ZMAX-ZTEM)-0.05) 960,960,539
539 CALL PHAZ2
```

```
      CALL REVAMP
      IF (Z-ZSAV-0.05) 540,540,941
941 DO 943 J=1,IVAR
943 XR(J)=X(J)
      ZSAV=Z
      WRITE (I2,81) Z,(J,X(J),J=1,IVAR)
      GO TO 927
540 AVFAC=2.0
      CALL ADJUST
      IF (ZMAX-ZSAV-0.05) 960,960,941
960 IF (XBIG-50.0) 961,8,8
961 WRITE (I2,962)
962 FORMAT (18HODECR XBIG MAX AMT)
      DO 965 J=1,IVAR
965 X(J)=XR(J)
      Z=ZSAV
969 CALL SLACK
      INCR=X(JSAB)
      DO 520 I=1,ICON
      IF (A(I,JSAB)) 505,520,520
505 IDELX=DT(I)/A(I,JSAB)
      IF (IDELX-INCR) 508,520,520
508 INCR=IDELX
520 CONTINUE
      IF (INCR) 967,967,792
792 XINCR=INCR
      X(JSAB)=X(JSAB)-XINCR
      Z=Z-C(JSAB)*XINCR
      ZMAX=Z
      KHOLD=JSAB
      CALL PHAZ1
      CALL REVAMP
      KHOLD=0
      CALL PHAZ2
519 ZTEM=Z
      AVFAC=1.0
      CALL ADJUST
      IF (Z-ZTEM-0.05) 522,522,521
521 ZTEM=Z
      CALL PHAZ2
      IF (Z-ZTEM-0.05) 522,522,519
522 IF (Z-ZSAV-0.05) 8,8,975
967 Z=0.0
      WRITE (I2,963)
963 FORMAT (15HODECR BY 20 PCT)
      DO 970 J=1,IVAR
      IF (X(J)) 970,970,966
966 IX=(4.0*X(J))/5.0
      XINCR=IX
      X(J)=XINCR
      CALL POSDT
      IF (INP) 970,970,968
968 X(J)=XR(J)
```

```

970 Z=Z+C(J)*X(J)
    ZMAX=Z
    IFFS=3
    KHOLD=JSAV
    CALL PHAZB
    CALL REVAMP
    IF (ZMAX-ZSAV-0.05) 972,972,975
972 KHOLD=0
    CALL PHAZ2
    AVFAC=2.0
    CALL ADJUST
    IF (ZMAX-ZSAV-0.05) 990,990,975
975 DO 980 J=1,IVAR
980 XR(J)=X(J)
    ZSAV=Z
    WRITE (I2,81) Z,(J,X(J),J=1,IVAR)
    GO TO 967
990 WRITE (I2,81) ZSAV,(J,XR(J),J=1,IVAR)
    GO TO 8
80 STOP
END

```

```

/FTC      LIST,REF,SIZE=256K
          SUBROUTINE PHAZB
          DIMENSION X(50),XS(50),DT(50),C(50),B(50),A(50,50),
1COL(50),IMSG(13),ICOST(50),IOCOST(50),IPDT(50),IODT(50),
X),XSS(50)
2,XL(50),XR(50),ISKIP(50)
          COMMON ICON,IVAR,Z,A,B,C,DT,X,XS,DELZ,COL,FRACT,DELX,
1IDELX,ZMAX,XDELX,XINCR,XXINC,ZS,ISW1,ISW2,JHAV,I1,I2,
2IFFS,IPROB,KPHASE,INP,INO,KTR,KTRO,ICOST,IOCOST,IPDT,I
XODT,ITRY
3,KHOLD,XSS,ZL,XL,ISKIP,JEND,AVFAC
          KPHASE=0
          KCHG=0
          CALL POSDT
          IF (INP) 5,5,86
5 ZM=0.
          DTMIN=1.0
          CALL SLACK
          DO 50 JJ=1,IVAR
          IF (KCHG) 4,4,6
4 J=JJ
          GO TO 13
6 J=IVAR-JJ+1
13 IF (J-KHOLD) 7,50,7
7 IMIN=10000000
          IF (C(J)) 8,50,20
8 IF (X(J)) 50,50,9

```

```
9 KALT=0
  DO 15 I=1,ICON
    IF (A(I,J)) 10,15,15
10 KALT=1
  IDELJ=DT(I)/A(I,J)
  IF (IDELJ) 50,50,11
11 IF (IDELJ-IMIN) 12,15,15
12 IMIN=IDELJ
15 CONTINUE
  IF (KALT) 16,16,17
16 IDELJ=-X(J)
  GO TO 30
17 JX=X(J)
  IF (IMIN-JX) 18,18,19
18 IDELJ=-IMIN
  GO TO 30
19 IDELJ=-X(J)
  GO TO 30
20 KAGT=0
  DO 25 I=1,ICON
    IF (A(I,J)) 25,25,21
21 KAGT=1
  IDELJ=-DT(I)/A(I,J)
  IF (IDELJ) 50,50,22
22 IF (IDELJ-IMIN) 23,25,25
23 IMIN=IDELJ
25 CONTINUE
  IF (KAGT) 26,26,27
26 WRITE (I2,900) J
900 FORMAT (1H0,10X,18HSOLUTION UNBOUNDED,10X,8HVARIBLE,
  X13,10X,
  17HPHASE 1)
  KPHASE=10
  RETURN
27 IDELJ=IMIN
30 DELJ=IDELJ
  DELZ=C(J)*DELJ
  IF (DELZ) 50,50,32
32 IF (IABS(IDELJ)-1) 50,110,120
110 DELX=1*ISIGN(1,IDELJ)
  GO TO 130
120 IF (IFFS-2) 127,127,122
122 IF (IABS(IDELJ)-2) 110,110,127
127 DELX=IDELJ/IFFS
130 DTM=-1.0E20
  DO 140 K=1,ICON
    DTS=DT(K)+A(K,J)*DELX
    IF (DTS-DTM) 140,140,135
135 DTM=DTS
140 CONTINUE
  IF (DTM-DTMIN) 150,50,50
150 DTMIN=DTM
  JSAV=J
```



```
SAVDX=DELX
50 CONTINUE
  IF (DTMIN-1.0) 51,82,82
51 X(JSAB)=X(JSAB)+SAVDX
  Z=Z+C(JSAB)*SAVDX
  IF (KCHG) 55,55,54
55 KCHG=1
  GO TO 5
54 KCHG=0
  GO TO 5
82 IF (Z-ZMAX) 86,86,83
86 WRITE (I2,98)
98 FORMAT (1H0,10X,20HNO HELP FROM PHASE B)
  GO TO 99
83 WRITE (I2,81) Z
81 FORMAT (1H0,10X,13HPHASE B, Z = ,F12.3)
  ZMAX=Z
  KPHASE=1
99 RETURN
END
```

```
/FTC      LIST,REF,SIZE=256K
SUBROUTINE PHAZ1
  DIMENSION X(50),XS(50),DT(50),C(50),B(50),A(50,50),
1COL(50),IMSG(13),ICOST(50),IQCOST(50),IPDT(50),IQDT(50),
X),XSS(50)
2,XL(50),XR(50),ISKIP(50)
  COMMON ICON,IVAR,Z,A,B,C,DT,X,XS,DELZ,COL,FRACT,DELX,
1IDELX,ZMAX,XDELX,XINCR,XXINC,ZS,ISW1,ISW2,JHAV,I1,I2,
2IFFS,IPROB,KPHASE,INP,INO,KTR,KTRO,ICOST,IQCOST,IPDT,I
XQDT,ITRY
3,KHOLD,XSS,ZL,XL,ISKIP,JEND,AVFAC
  KPHASE=0
  KCHG=0
  CALL POSDT
  IF (INP) 5,5,86
5 ZM=0.
  CALL SLACK
  DO 50 JJ=1,IVAR
  IF (KCHG) 4,4,6
4 J=JJ
  GO TO 13
6 J=IVAR-JJ+1
13 IF (J-KHOLD) 7,50,7
7 IMIN=10CC0000
  IF (C(J)) 8,50,20
8 IF (X(J)) 50,50,9
9 KALT=0
  DO 15 I=1,ICON
```

```
      IF (A(I,J)) 10,15,15
10  KALT=1
      IDELJ=DT(I)/A(I,J)
      IF (IDELJ) 50,50,11
11  IF (IDELJ-IMIN) 12,15,15
12  IMIN=IDELJ
15  CONTINUE
      IF (KALT) 16,16,17
16  IDELJ=-X(J)
      GO TO 30
17  JX=X(J)
      IF (IMIN-JX) 18,18,19
18  IDELJ=-IMIN
      GO TO 30
19  IDELJ=-X(J)
      GO TO 30
20  KAGT=0
      DO 25 I=1,ICON
      IF (A(I,J)) 25,25,21
21  KAGT=1
      IDELJ=-DT(I)/A(I,J)
      IF (IDELJ) 50,50,22
22  IF (IDELJ-IMIN) 23,25,25
23  IMIN=IDELJ
25  CONTINUE
      IF (KAGT) 26,26,27
26  WRITE (I2,900) J
900  FORMAT (1H0,10X,18HSOLUTION UNBOUNDED,10X,8HVARIABLE,
      XI3,10X,
      17HPHASE 1)
      KPHASE=10
      RETURN
27  IDELJ=IMIN
30  DELJ=IDELJ
      DELZ=C(J)*DELJ
      IF (DELZ-ZM) 50,50,32
32  ZM=DELZ
      JSAV=J
      JSVDJ=IDELJ
50  CONTINUE
      IF (ZM) 82,82,51
51  IF (IABS(JSVDJ)-2) 52,52,53
52  IDELX=1*ISIGN(1,JSVDJ)
      GO TO 60
53  IDELX=JSVDJ/3
60  DELX=IDELX
      X(JSAV)=X(JSAV)+DELX
      Z=Z+C(JSAV)*DELX
      IF (KCHG) 55,55,54
55  KCHG=1
      GO TO 5
54  KCHG=0
      GO TO 5
```

```

82 IF (Z-ZMAX) 86,86,83
86 WRITE (I2,98)
98 FORMAT (1H0,10X,20HNO HELP FROM PHASE 1)
GO TO 99
83 WRITE (I2,81) Z
81 FORMAT (1H0,10X,13HPPHASE 1, Z = ,F12.3)
ZMAX=Z
KPPHASE=1
99 RETURN
END

```

```

/FTC LIST,REF,SIZE=256K
SUBROUTINE PHAZZ
DIMENSION X(50),XS(50),DT(50),C(50),B(50),A(50,50),
1COL(50),IMSG(13),ICOST(50),IOCOST(50),IPDT(50),IODT(50
X),XSS(50)
2,XL(50),XR(50),ISKIP(50)
COMMON ICON,IVAR,Z,A,B,C,DT,X,XS,DELZ,COL,FRACT,DELX,
1IDELX,ZMAX,XDELX,XINCR,XXINC,ZS,ISW1,ISW2,JHAV,I1,I2,
2IFFS,IPROB,KPPHASE,INP,INO,KTR,KTRO,ICOST,IOCAST,IPDT,I
XODT,ITRY
3,KHOLD,XSS,ZL,XL,ISKIP,JEND,AVFAC
WRITE (I2,910) Z
910 FORMAT (1H0,9X,7HPPHASE 2,F10.2)
IF (KHOLD) 912,912,1000
912 IF (ABS(Z-ZSAV)-0.05) 155,155,1000
1000 KPPHASE=0
ITAB=0
ZSAV=Z
DO 10 J=1,IVAR
10 XS(J)=X(J)
1 XMAX=0.0
XXINC=0.
DZMAX=-1000000.
DO 3 J=1,IVAR
IF (X(J)-XMAX) 3,3,2
2 XMAX=X(J)
3 CONTINUE
IF (XMAX-25.) 4,4,6
4 JXX=XMAX
KXX=(JXX/2)*2
IF (JXX-KXX) 508,508,7
508 XMAX=XMAX+1.0
GO TO 7
6 IF (XMAX-100.) 8,7,9
8 IXM=XMAX
IXM=(IXM/10)+1
XMAX=IXM*10
GO TO 7

```

```

9 IXM=XMAX
  IXM=(IXM/100)+1
  XMAX=IXM*100
7 XINCR=XMAX
  DO 100 J=1,KTR
    XINCR=XMAX
    JJ=ICOST(J)
    IF (JJ-KHOLD) 21,100,21
21 IF (C(JJ)) 11,100,14
11 IF (X(JJ)) 100,100,12
12 IF (XINCR-X(JJ)) 14,14,13
13 XINCR=X(JJ)
14 X(JJ)=X(JJ)+XINCR*SIGN(1.0,C(JJ))
  DELZR=C(JJ)*XINCR*SIGN(1.0,C(JJ))
  DELZK=0.0
  CALL POSDT
  IF (INP) 400,400,35
400 Z=Z+DELZR
  IF (Z-ZMAX) 1,1,301
35 DO 90 KK=1,IVAR
  IF (KK-JJ) 16,90,16
16 IF (KK-KHOLD) 22,90,22
22 IF (INO) 30,30,18
18 JNO=INP
  KNO=0
  DO 25 K=1,INO
    II=IODT(K)
    IF (A(II,KK)) 20,25,20
20 JNO=JNO+1
  KNO=KNO+1
  IPDT(JNO)=II
25 CONTINUE
  INUM=INP+KNO
  GO TO 37
30 INUM=INP
37 IA=IPDT(1)
  IF (A(IA,KK)) 40,90,50
40 DO 45 IN=1,INUM
  IA=IPDT(IN)
  IF (A(IA,KK)) 45,90,90
45 CONTINUE
  JSGN=1
  GO TO 60
50 IF (X(KK)) 90,90,51
51 DO 55 IN=1,INUM
  IA=IPDT(IN)
  IF (A(IA,KK)) 90,90,55
55 CONTINUE
  JSGN=-1
60 DMAX=0.0
  DO 65 IN=1,INP
  IA=IPDT(IN)
  DELX=ABS(DT(IA)/A(IA,KK))

```

```
      IF (DELX-DMAX) 65,65,62
62  DMAX=DELX
65  CONTINUE
      IDELX=DMAX
      DELX=IDELX
      IF (DMAX-DELX-0.0001) 67,67,66
66  IDELX=IDELX+1
67  IDELX=IDELX*JSGN
      JX=X(KK)
      IF (JX+IDELX) 90,68,68
68  DELX=IDELX
      X(KK)=X(KK)+DELX
      DTMAX=0.0
      DO 75 I3=1,ICON
      DTS=0.0
      DO 70 J3=1,IVAR
70  DTS=DTS+A(I3,J3)*X(J3)
      DTS=DTS-B(I3)
71  IF (DTS-0.005) 75,75,80
75  CONTINUE
      DELZK=C(KK)*DELX
      DELZ=DELZR+DELZK
      IF (DELZ) 76,76,300
300 Z=Z+DELZ
      IF (Z-ZMAX-0.005) 1,1,301
76  IF (DELZ-DZMAX) 80,80,77
77  DZMAX=DELZ
      JJSAV=JJ
      KKSAV=KK
      XJSAV=X(JJ)
      XKSAV=X(KK)
      XXINC=XINCR
80  X(KK)=X(KK)-DELX
90  CONTINUE
      X(JJ)=X(JJ)-XINCR*SIGN(1.0,C(JJ))
      IF (XINCR-100.) 91,93,94
91  IF (XINCR-30.) 92,92,93
92  IF (XINCR-2.) 100,509,511
509 XINCR=1.0
      GO TO 14
511 XINCR=XINCR-2.0
      GO TO 14
93  XINCR=XINCR-10.
      GO TO 14
94  IF (XINCR-1000.) 95,95,96
95  XINCR=XINCR-50.
      GO TO 14
96  XINCR=XINCR-500.
      GO TO 14
99  IF (XINCR) 100,100,14
100 CONTINUE
      IF (DZMAX+1000000.) 120,120,110
110 X(JJSAV)=XJSAV
```

```
X(KKSAV)=XKSAV
Z=Z+DZMAX
ZTEMP=Z
IF (IFFS-2) 445,445,455
445 CALL PHAZ1
GO TO 450
455 CALL PHAZB
450 IF (KPHASE-1) 112,200,200
112 IF (ITAB) 114,114,120
114 ITAB=ITAB+1
GO TO 1
120 WRITE (I2,101)
101 FORMAT (1H0,10X,20HNO HELP FROM PHASE 2)
KPHASE=0
DO 150 K=1,IVAR
150 X(K)=XS(K)
Z=ZSAV
155 RETURN
200 IF (ITAB) 210,210,220
220 WRITE (I2,221)
221 FORMAT (1H0,10X,27HPHASE 1 AND 2 APPLIED TWICE)
KPHASE=2
GO TO 321
210 WRITE (I2,211)
211 FORMAT (1H0,10X,21HPHASE 2 USING PHASE 1)
KPHASE=2
GO TO 321
301 KPHASE=1
IF (ITAB) 321,321,305
305 IF (IFFS-2) 545,545,555
545 CALL PHAZ1
GO TO 220
555 CALL PHAZB
GO TO 220
321 ZMAX=Z
WRITE (I2,81) Z
81 FORMAT (1H0,10X,13HPHASE 2, Z = ,F12.3)
GO TO 1000
END
```

```
/FTC LIST,REF,SIZE=256K
SUBROUTINE POSDT
DIMENSION X(50),XS(50),DT(50),C(50),B(50),A(50,50),
1COL(50),IMSG(13),ICOST(50),IOCOST(50),IPDT(50),IODT(50
X),XSS(50)
2,XL(50),XR(50),ISKIP(50)
COMMON ICON,IVAR,Z,A,B,C,DT,X,XS,DELZ,COL,FRACT,DELX,
1IDELX,ZMAX,XDELX,XINCR,XXINC,ZS,ISW1,ISW2,JHAV,I1,I2,
2IFFS,IPROB,KPHASE,INP,INO,KTR,KTRO,ICOST,IOCAST,IPDT,I
```

```

XODT,ITRY
3,KHOLD,XSS,ZL,XL,ISKIP,JEND,AVFAC
  INP=0
  INO=0
  DTMAX=-1000000.
  DO 30 I=1,ICON
    DT(I)=0.0
    DO 10 J=1,IVAR
10  DT(I)=DT(I)+A(I,J)*X(J)
    DT(I)=DT(I)-B(I)
    IF (ABS(DT(I))-0.005) 12,12,14
12  DT(I)=0.0
14  IF (DT(I)) 23,15,20
15  INO=INO+1
    IODT(INO)=I
    GO TO 23
20  INP=INP+1
    IPDT(INP)=I
23  IF (DT(I)-DTMAX) 30,30,25
25  DTMAX=DT(I)
    JHAV=I
30  CONTINUE
    RETURN
  END

```

```

/FTC      LIST,REF,SIZE=256K
          SUBROUTINE SLACK
            DIMENSION X(50),XS(50),DT(50),C(50),B(50),A(50,50),
1COL(50),IMSG(13),ICOST(50),IOCAST(50),IPDT(50),IODT(50
X),XSS(50)
2,XL(50),XR(50),ISKIP(50)
          COMMON ICON,IVAR,Z,A,B,C,DT,X,XS,DELZ,COL,FRACT,DELX,
1IDELX,ZMAX,XDELX,XINCR,XXINC,ZS,ISW1,ISW2,JHAV,I1,I2,
2IFFS,IPROB,KPHASE,INP,INO,KTR,KTRO,ICOST,IOCAST,IPDT,I
XODT,ITRY
3,KHOLD,XSS,ZL,XL,ISKIP,JEND,AVFAC
  DO 15 I=1,ICON
    DT(I)=0.0
    DO 10 J=1,IVAR
10  DT(I)=DT(I)+A(I,J)*X(J)
15  DT(I)=DT(I)-B(I)
    RETURN
  END

```

```
/FTC      LIST,REF,SIZE=256K
          SUBROUTINE INTLZ
            DIMENSION X(50),XS(50),DT(50),C(50),B(50),A(50,50),
1COL(50),IMSG(13),ICOST(50),IOCOST(50),IPDT(50),IODT(50
X),XSS(50)
            2,XL(50),XR(50),ISKIP(50)
            COMMON ICON,IVAR,Z,A,B,C,DT,X,XS,DELZ,COL,FRACT,DELX,
1IDELX,ZMAX,XDELX,XINCR,XXINC,ZS,ISW1,ISW2,JHAV,I1,I2,
2IFFS,IPROB,KPHASE,INP,INO,KTR,KTRO,ICOST,IOCOST,IPDT,I
XODT,ITRY
            3,KHOLD,XSS,ZL,XL,ISKIP,JEND,AVFAC
            WRITE (I2,150)
150 FORMAT (1H0,9X,12HINITIAL SOLN)
            ZSAV=Z
            DO 5 J=1,IVAR
              5 XS(J)=X(J)
            DO 10 J=1,IVAR
              10 X(J)=1.0
              15 CALL POSDT
                IF (INP) 100,100,20
              20 AMAX=0.
                DO 30 J=1,IVAR
                  IF (X(J)) 30,30,23
                23 IF (A(JHAV,J)-AMAX) 30,30,24
                24 AMAX=A(JHAV,J)
                  JSAV=J
                30 CONTINUE
                  IF (AMAX) 40,40,31
                31 X(JSAV)=0.
                  GO TO 15
                40 KPHASE=0
                  Z=ZSAV
                  DO 50 J=1,IVAR
                    50 X(J)=XS(J)
                  WRITE (I2,60)
                60 FORMAT (1H0,10X,7HNO HELP)
                  RETURN
            100 KPHASE=1
                Z=0.
                DO 120 J=1,IVAR
                  120 Z=Z+C(J)*X(J)
                  WRITE (I2,160) Z,(J,X(J),J=1,IVAR)
            160 FORMAT (1H0,20X,2HZ=,F10.3,/(21X,1HX,I2,1H=,F10.3))
                  RETURN
            END
```



```

/FTC      LIST,REF,SIZE=256K
          SUBROUTINE REVAMP
          DIMENSION X(50),XS(50),DT(50),C(50),B(50),A(50,50),
1COL(50),IMSG(13),ICOST(50),IOCAST(50),IPDT(50),IODT(50
X),XSS(50)
          2,XL(50),XR(50),ISKIP(50)
          COMMON ICON,IVAR,Z,A,B,C,DT,X,XS,DELZ,COL,FRACT,DELX,
1IDELX,ZMAX,XDELX,XINCR,XXINC,ZS,ISW1,ISW2,JHAV,I1,I2,
2IFFS,IPROB,KPHASE,INP,INO,KTR,KTRO,ICOST,IOCAST,IPDT,I
XODT,ITRY
          3,KHGLD,XSS,ZL,XL,ISKIP,JEND,AVFAC
          KEND=0
          DO 750 I=1,ICON
          CCL(I)=0.
          KNTR=0
          DO 720 J=1,IVAR
          IF (A(I,J)) 720,720,710
710 KNTR=KNTR+1
          CCL(I)=CCL(I)+A(I,J)
720 CONTINUE
          AKTR=KNTR
          IF (AKTR) 730,730,740
730 DO 735 J=1,IVAR
          IF (A(I,J)) 732,735,735
732 COL(I)=COL(I)+ABS(A(I,J))
          KNTR=KNTR+1
735 CONTINUE
          AKTR=KNTR
740 COL(I)=COL(I)/AKTR
750 CONTINUE
          IUP=0
          1 WRITE (I2,150) Z
150 FORMAT (1H0,9X,11HREVAMP SOLN,F10.4)
          KEND=KEND+1
          DO 3 K=1,IVAR
          3 XL(K)=X(K)
          ZSAV=Z
          CALL SLACK
          IF (ISW2-100) 5,4,5
          5 IF (KEND-20) 8,8,4
          4 DO 7 K=1,IVAR
          IF (K-KHGLD) 6,7,6
          6 X(K)=X(K)+10.0
          7 CONTINUE
          ISW2=100
          WRITE (I2,882)
882 FORMAT (1H0,5X,11HINCR BY TEN)
          GO TO 15
          8 DO 880 K=1,IVAR
880 X(K)=X(K)+1.0
          15 CALL POSDT
          ISAV=0
          IF (INP) 100,100,20

```

```
20 AMAX=-1000000.
   DO 24 I=1,ICON
     IF (DT(I)) 24,24,22
22 RAT=DT(I)/COL(I)
   IF (RAT-AMAX) 24,24,23
23 AMAX=RAT
   ISAV=I
24 CONTINUE
   IF (ISAV) 59,59,25
25 AMAX=0.
   RMAX=0.
   RMIN=1000000.
   JSAV=0
   KSAV=0
   LSAV=0
   DO 50 J=1,IVAR
     IF (A(ISAV,J)) 31,50,26
26 IF (X(J)) 50,50,27
27 IF (C(J)) 32,34,28
28 RAT1=ABS(A(ISAV,J)/DT(ISAV))
   IRAT=RAT1
   IF (IRAT) 29,29,210
29 RAT1=1.0
210 RAT=RAT1+A(ISAV,J)/C(J)
   IF (RAT-RMAX) 50,50,30
30 RMAX=RAT
   JSAV=J
   GO TO 50
31 IF (C(J)) 28,34,32
32 RAT=-A(ISAV,J)/C(J)
767 IF (RAT-RMIN) 33,50,50
33 DO 270 I=1,ICON
   IF (DT(I)) 252,255,255
252 RSTR=A(I,J)*SIGN(1.0,A(ISAV,J))
   IF (DT(I)-RSTR) 270,270,50
255 IF (I-ISAV) 257,270,257
257 IF (SIGN(1.0,A(I,J))-SIGN(1.0,A(ISAV,J))) 50,270,50
270 CONTINUE
   RMIN=RAT
   KSAV=J
   GO TO 50
34 RAT=ABS(A(ISAV,J))+A(ISAV,J)/DT(ISAV)
   GO TO 767
50 CONTINUE
   IF (KSAV) 51,51,54
51 IF (LSAV) 52,52,55
52 JCHG=JSAV
   IF (JCHG) 59,59,56
54 JCHG=KSAV
   GO TO 56
55 JCHG=LSAV
56 X(JCHG)=X(JCHG)-SIGN(1.0,A(ISAV,JCHG))
   GO TO 15
```

```
59 WRITE (I2,60)
60 FORMAT (1H0,10X,7HNO HELP)
GO TO 610
100 KPHASE=1
Z=0.
DO 120 J=1,IVAR
120 Z=Z+C(J)*X(J)
CALL PHAZB
IF (KPHASE-1) 610,608,610
608 ISW1=0
IF (IUP) 1,1,510
510 DO 520 J=1,IVAR
520 XL(J)=X(J)
ZSAV=Z
GO TO 630
610 DO 620 J=1,IVAR
620 X(J)=XL(J)
Z=ZSAV
IF (ISW1-100) 625,621,625
621 DO 623 K=1,IVAR
623 X(K)=XS(K)
Z=ZL
ISW1=0
IF (Z) 1,910,1
910 DO 930 I=1,ICON
IF (B(I)) 915,930,930
915 DO 920 J=1,IVAR
IF (A(I,J)) 917,920,920
917 INCR=B(I)/A(I,J)+1.0
XINCR=INCR
X(J)=X(J)+XINCR
GO TO 930
920 CONTINUE
930 CONTINUE
ZMAX=-1000000.
GO TO 15
625 IF (IUP) 630,630,680
630 IUP=IUP+1
DO 650 K=1,IVAR
IF (X(K)-1.0) 650,650,640
640 X(K)=X(K)+2.0
650 CONTINUE
WRITE (I2,655)
655 FORMAT (1H0,5X,11HINCR BY TWO)
IF (IUP-5) 15,15,4
680 ISW2=0
RETURN
END
```

```
/FTC      LIST,REF,SIZE=256K
          SUBROUTINE ADJUST
            DIMENSION X(50),XS(50),DT(50),C(50),B(50),A(50,50),
            1CCL(50),IMSG(13),ICOST(50),IOCOST(50),IPDT(50),IUDT(50
            X),XSS(50)
            2,XL(50),XR(50),ISKIP(50),XT(50)
            COMMON ICON,IVAR,Z,A,B,C,DT,X,XS,DELZ,COL,FRACT,DELX,
            1IDELX,ZMAX,XDELX,XINCR,XXINC,ZS,ISW1,ISW2,JHAV,I1,I2,
            2IFFS,IPROB,KPHASE,INP,INO,KTR,KTRO,ICOST,IOCOST,IPDT,I
            XODT,ITRY
            3,KHOLD,XSS,ZL,XL,ISKIP,JEND,AVFAC
            WRITE (I2,300) Z
300  FORMAT (1H0,10X,6HADJUST,F10.3)
            DO 25 J=1,IVAR
            IF (C(J)) 24,23,24
            23 XSS(J)=0.01
            GO TO 25
            24 XSS(J)=ABS(C(J)/FRACT)
            25 CONTINUE
1000 KPH=0
            WRITE (I2,833) AVFAC
            833 FORMAT (1H0,5X,17HDECR POS VAR BY 1,5X,8HAVFAC = ,
            XF5.1)
            IJOB=0
            1 DO 5 J=1,IVAR
            5 XT(J)=X(J)
            ZSAV=Z
            ISW2=0
            DO 847 J=1,IVAR
            IF (X(J)) 847,847,844
            844 X(J)=X(J)-1.0
            CALL POSDT
            IF (INP) 847,847,845
            845 X(J)=X(J)+1.0
            847 CCONTINUE
            GO TO 410
            6 NVAR=0
            WRITE (I2,927) AVFAC
            927 FORMAT (1H0,5X,12HDECR BY ICUT,5X,8HAVFAC = ,F5.1)
            IHAV=ICON/2
            ICUT=0
            DO 878 J=1,IVAR
            IF (X(J)) 878,878,877
            877 ICUT=ICUT+1
            878 CONTINUE
            ICUT=ICUT/3+1
            DO 400 I=1,ICUT
            CALL POSDT
            SUM=0.0
            DO 10 K=1,ICON
            10 SUM=SUM+DT(K)
            CGN=ICON
            AVDT=SUM/CGN
```

```
AVDT=AVFAC*AVDT
NUM=0
DMIN=0.0
DO 20 K=1,ICON
IF (DT(K)-AVDT) 20,15,15
15 NUM=NUM+1
XS(NUM)=K
IF (DT(K)-DMIN) 16,20,20
16 DMIN=DT(K)
20 CONTINUE
AVDT=DMIN*1.05
21 NUM=0
DO 820 K=1,ICON
IF (DT(K)-AVDT) 820,815,815
815 NUM=NUM+1
XS(NUM)=K
820 CONTINUE
IF (NUM) 822,822,828
822 DMAX=-100000.
DO 824 K=1,ICON
IF (DT(K)-DMAX) 824,824,823
823 DMAX=DT(K)
824 CONTINUE
AVDT=1.1*DMAX
GO TO 21
828 DO 350 J=1,IVAR
COL(J)=0.
IF (X(J)) 350,350,330
330 DO 335 K=1,NUM
ID=XS(K)
335 COL(J)=COL(J)+A(ID,J)
350 CONTINUE
DO 390 J=1,IVAR
390 COL(J)=COL(J)+COL(J)/XSS(J)
RMIN=100000.
346 JSAV=0
RMAX=0.
DO 360 J=1,IVAR
IF (X(J)) 360,360,351
351 IF (COL(J)-RMIN) 352,360,360
352 IF (COL(J)-RMAX) 360,354,353
354 IF (JSAV) 353,353,355
355 IF (C(J)-C(JSAV)) 353,360,360
353 RMAX=COL(J)
JSAV=J
360 CONTINUE
IF (JSAV) 410,410,365
365 X(JSAV)=0.0
369 CALL POSDT
IF (-INP) 400,400,370
370 X(JSAV)=X(JSAV)+1.
IF (X(JSAV)-XT(JSAV)) 369,371,371
371 RMIN=RMAX
```

```
GO TO 346
400 CONTINUE
CALL POSDT
IF (INP) 410,410,405
405 CALL REVAMP
410 DO 500 I=1,ICON
CALL POSDT
SUM=0.0
DO 610 K=1,ICON
610 SUM=SUM+DT(K)
CON=ICON
AVDT=SUM/CON
AVDT=AVFAC*AVDT
NUM=0
DMIN=0.0
DO 620 K=1,ICON
IF (DT(K)-AVDT) 620,615,615
615 NUM=NUM+1
XS(NUM)=K
IF (DT(K)-DMIN) 616,620,620
616 DMIN=DT(K)
620 CONTINUE
AVDT=DMIN*1.05
521 NUM=0
DO 720 K=1,ICON
IF (DT(K)-AVDT) 720,715,715
715 NUM=NUM+1
XS(NUM)=K
720 CONTINUE
IF (NUM) 522,522,528
522 DMAX=-100000.
DO 524 K=1,ICON
IF (DT(K)-DMAX) 524,524,523
523 DMAX=DT(K)
524 CONTINUE
AVDT=1.1*DMAX
GO TO 521
528 DO 450 J=1,IVAR
COL(J)=0.
DO 435 K=1,NUM
ID=XS(K)
IF (A(ID,J)) 435,435,432
432 IF (DT(ID)) 434,433,433
433 COL(J)=COL(J)+A(ID,J)
GO TO 435
434 COL(J)=COL(J)-A(ID,J)
435 CONTINUE
450 CONTINUE
DO 460 J=1,IVAR
COL(J)=ABS(COL(J)/XSS(J))
460 CONTINUE
RMAX=-0.1
451 JSAV=0
```

```
RMIN=1000000.
DC 650 J=1,IVAR
IF (COL(J)-RMAX) 650,650,452
452 IF (COL(J)-RMIN) 453,454,650
454 IF (JSAV) 453,453,455
455 IF (C(J)-C(JSAV)) 650,650,453
453 RMIN=COL(J)
JSAV=J
650 CONTINUE
IF (JSAV) 501,501,461
461 VAL=1000000.
DO 470 K=1,ICON
IF (A(K,JSAV)) 470,470,465
465 XINCR=-DT(K)/A(K,JSAV)
IF (XINCR-VAL) 467,470,470
467 VAL=XINCR
470 CONTINUE
IF (VAL-1000000.) 471,987,987
987 WRITE (I2,988) JSAV
988 FORMAT (1H0,11HCOEFFS OF X,I3,7HALL NEG)
GO TO 501
471 IF (VAL-1.0) 482,472,474
472 XINCR=1.0
GO TO 480
474 IF (VAL-2.0) 472,472,476
476 INCR=VAL/2.
XINCR=INCR
480 X(JSAV)=X(JSAV)+XINCR
CALL POSDT
IF (INP) 500,500,481
481 X(JSAV)=X(JSAV)-XINCR
482 RMAX=RMIN
CALL POSDT
GO TO 451
500 CONTINUE
501 Z=0.
DO 210 J=1,IVAR
210 Z=Z+X(J)*C(J)
WRITE (I2,320) Z
320 FORMAT (1H0,10X,13HADJUSTED Z = ,F10.2)
IF (ABS(Z-ZSAV)-0.05) 240,240,211
211 CALL PHAZB
IF (KPHASE-1) 240,212,240
212 ZMAX=Z
WRITE (I2,81) Z
81 FORMAT (1H0,10X,13HPHASE A, Z = ,F12.3)
GO TO 1000
240 IF (IJOB) 241,241,243
241 IJOB=1
DO 246 J=1,IVAR
246 X(J)=XT(J)
GO TO 6
243 DO 245 J=1,IVAR
```

```
245 X(J)=XT(J)
    Z=ZSAV
    IF (JEND) 250,250,769
769 IF (AVFAC-1.0) 770,775,250
770 AVFAC=1.0
    GO TO 1000
775 AVFAC=2.0
    GO TO 1000
250 RETURN
    END
```


APPENDIX II

PROBLEM GENERATOR FOR INTEGER LINEAR PROGRAMMING
PROBLEMS WHOSE OPTIMAL INTEGER SOLUTION COINCIDES
WITH THE LINEAR PROGRAMMING SOLUTION

```

/JOB 05 B626,DREBES,MULTIPLE,GO,MAP
/FTC LIST
  DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
  IB(30),X(100),IPDT(100),ICOST(30),COL(30),CS(100),Y(32,
X100)
  COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
X,I,J,SUMU,
  ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE
XGF,CNEGF,
  2VARF,FAC,COSTF,RATIO,IBAD,NS,IPROB,I1,I2,NPOS,CS,N1
  READ (5,3) I1,I2
  3 FORMAT (2I2)
  1 READ (I1,50) IPROB,ICON,IVAR,INITL,DENS,RNEGF,CNEGF,VA
XRF,FAC,
  ICOSTF,RATIO
  50 FORMAT (3I5,I10,7F5.0)
  WRITE (I2,950) IPROB,ICON,IVAR,INITL,DENS,RNEGF,CNEGF
950 FORMAT (1H1,4HPROB,I3,I7,4H CON,I7,4H VAR,3X,6HINITL=,
XI10,3X,
  15HDENS=,F6.4,3X,5HNEGF=,F6.4,3X,6HCNEGF=,F6.4)
  WRITE (I2,951) VARF,FAC,COSTF,RATIO
951 FORMAT (1H0,10X,5HVARF=,F7.1,3X,4HFAC=,F4.1,3X,
X6HCOSTF=,F8.1,5X,
  16HRATIO=,F4.1)
  IBAD=0
  INSAV=INITL
  5 CALL GEN
  CALL WRIT
  IF (IBAD-1) 10,15,15
  15 WRITE (I2,66)
  66 FORMAT (1H0,5X,11HGEN FAILURE)
  GO TO 1
  10 CALL OPT
  WRITE (I2,130)
  130 FORMAT (1H0,20X,15HBASIC FEAS SOLN)
  WRITE (I2,135) (ICOST(I),YU(I,N1),I=1,ICON)
  135 FORMAT (5X,1HX,I2,3H = ,F10.1)
  WRITE (I2,137) YU(M1,N1)
  137 FORMAT (1H0,10X,7HZMAX = ,F10.1)
  WRITE (I2,140)
  140 FORMAT (1H0,20X,5HZJ-CJ)
  WRITE (I2,145) (YU(M1,J),J=1,NREAL)
  145 FORMAT (1H0,12F8.1)
  IF (IBAD-1) 80,65,70
  65 WRITE (I2,760)

```

```
760 FORMAT (1H0,5X,11HOPT' FAILURE)
    GO TO 1
70  CALL CBR
    CALL SIMPL(1)
    WRITE (I2,710)
710 FORMAT (1H0,5X,12HCOST CHANGES)
    WRITE (I2,720) (IV(I),XBU(I),I=1,M1)
720 FORMAT (1H0,5X,1HX,I2,3H = ,F10.3)
    IF (IBAD) 740,740,71
71  WRITE (I2,730)
730 FORMAT (1H0,5X,15HSIMPLEX FAILURE)
    GO TO 1
740 IDUM=0
    DO 75 I=1,ICON
    IF (ICOST(I)-IVAR) 72,72,75
72  IDUM=IDUM+1
    DO 74 K=1,NPOS
    IF (IV(K)-IDUM) 74,73,74
73  JX=ICOST(I)
    CS(JX)=CS(JX)+XBU(K)
    KVAL=CS(JX)+1.0
    CS(JX)=KVAL
    GO TO 75
74  CONTINUE
75  CONTINUE
    WRITE (I2,705)
705 FORMAT (1H0,5X,9HNEW COSTS)
    WRITE (I2,706) (CS(J),J=1,IVAR)
706 FORMAT (1H0,5X,10F10.3)
    INITL=INSAV
    CALL GEN
    DO 78 J=1,IVAR
78  C(J)=CS(J)
    CALL OPT
    ZMAX=YU(M1,N1)
    WRITE (I2,130)
    WRITE (I2,135) (ICOST(I),YU(I,N1),I=1,ICON)
    WRITE (I2,137) YU(M1,N1)
    WRITE (I2,140)
    WRITE (I2,145) (YU(M1,J),J=1,NREAL)
80  DO 95 J=1,IVAR
    IF (YU(M1,J)) 93,95,95
93  C(J)=C(J)+YU(M1,J)-1.0
    KVAL=C(J)
    C(J)=KVAL
95  CONTINUE
    WRITE (I2,705)
    WRITE (I2,706) (C(J),J=1,IVAR)
    DO 100 I=1,ICON
    TYPE(I)=1.0
    XBU(I)=B(I)
    DO 100 J=1,IVAR
100  YU(I,J)=A(I,J)
```

```
M=ICON
NREAL=IVAR
CALL SIMPL(1)
WRITE (12,800)
800 FORMAT (1H0,5X,8HOPT SOLN)
WRITE (12,720) (IV(I),XBU(I),I=1,M1)
WRITE (3,610) ICON,IVAR,IPROB,ZMAX
610 FORMAT (3I5,15X,F10.0)
WRITE (3,615) (X(J),J=1,IVAR)
615 FORMAT (16F5.0)
DO 630 I=1,ICON
630 WRITE (3,615) (A(I,J),J=1,IVAR)
WRITE (3,615) (B(I),I=1,ICON)
WRITE (3,615) (C(J),J=1,IVAR)
GO TO 1
END
```

/FTC LIST,SIZE=256K

```
SUBROUTINE GEN
DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
1B(30),X(100),IPDT(100),ICOST(30),COL(30),CS(100),Y(32,
X100)
COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
X,I,J,SUMU,
ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE
XGF,CNEGF,
2VARF,FAC,COSTF,RATIO,IBAD,NS,IPROB,I1,I2,NPOS,CS,N1
IBAD=0
RN=RANF(INITL)
DO 15 I=1,ICON
DO 15 J=1,IVAR
RN=RANF(0)
IF (RN-DENS) 10,14,14
10 JA=RANF(0)*FAC+1.0
RNEG=RANF(0)
IF (RNEG-RNEGF) 12,13,13
12 A(I,J)=-JA
GO TO 15
13 A(I,J)=JA
GO TO 15
14 A(I,J)=0.0
15 CONTINUE
DO 18 I=1,ICON
DO 19 J=1,IVAR
IF (A(I,J)) 18,19,18
19 CONTINUE
A(I,I)=1.0
18 CONTINUE
```

```
DO 16 J=1,IVAR
X(J)=0.0
DO 17 I=1,ICON
IF (A(I,J)) 16,17,16
17 CONTINUE
A(I,J)=1.0
16 CONTINUE
DO 40 J=1,IVAR
JC=RANF(0)*COSTF+1.0
CNEG=RANF(0)
IF (CNEG-CNEGF) 20,25,25
20 C(J)=-JC
GO TO 40
25 C(J)=JC
40 CONTINUE
TOON=ICON
NPOS=TCN*RATIO
MPOS=0
DO 60 I=1,IVAR
IF (RANF(0)-RATIO) 50,60,60
50 IX=RANF(0)*VARF+1.0
X(I)=IX
MPOS=MPOS+1
IF (MPOS-NPOS) 60,80,80
60 CONTINUE
63 DO 70 I=1,IVAR
IF (X(I)) 65,65,70
65 IF (RANF(0)-RATIO) 67,70,70
67 IX=RANF(0)*VARF+1.0
X(I)=IX
MPOS=MPOS+1
IF (MPOS-NPOS) 70,80,80
70 CONTINUE
GO TO 63
80 DO 90 I=1,ICON
B(I)=0.
DO 90 J=1,IVAR
90 B(I)=B(I)+A(I,J)*X(J)
NSLAK=ICCN-NPOS
NS=NSLAK+1
DO 120 I=1,NSLAK
92 FPCT=RANF(0)
IF (FPCT-0.5) 93,92,92
93 B(I)=B(I)+B(I)*FPCT
IB=B(I)+1.0
B(I)=IB
120 CONTINUE
IVP=0
DO 140 J=1,IVAR
IF (X(J)) 140,140,130
130 IVP=IVP+1
IPDT(IVP)=J
140 CONTINUE
```

```
      DO 150 J=1,NSLAK
      JJ=IVAR+J
      IVP=IVP+1
      IPDT(IVP)=JJ
150  CONTINUE
      IF (IVP-ICON) 149,152,149
149  IBAC=1
      RETURN
152  DO 160 I=1,ICON
      DO 160 J=1,ICON
      JJ=IVAR+J
      IF (I-J) 157,153,157
153  A(I,JJ)=1.0
      GO TO 159
157  A(I,JJ)=0.0
159  C(JJ)=0.0
160  CONTINUE
      NREAL=IVAR+ICON
      IBAD=0
      RETURN
      END
```

```
/FTC      LIST,SIZE=256K
          SUBROUTINE WRIT
          DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
          IB(30),X(100),IPDT(100),ICOST(30),COL(30),CS(100),Y(32,
X100)
          COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
          X,I,J,SUMU,
          ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE
          XGF,CNEGF,
          2VARF,FAC,COSTF,RATIO,IBAD,NS,IPROB,I1,I2,NPOS,CS,N1
          WRITE (I2,205)
205  EFORMAT (1H0,20X,8HA MATRIX)
          DO 220 I=1,ICON
          WRITE (I2,340) I
340  FORMAT (1H0,4HROW ,I3)
220  WRITE (I2,210) (A(I,J),J=1,NREAL)
210  FORMAT (1H0,12F8.1)
          WRITE (I2,215)
215  FORMAT (1H0,20X,10HRIGHT SIDE)
          WRITE (I2,210) (B(I),I=1,ICON)
          WRITE (I2,230)
230  FORMAT (1H0,20X,13HINITIAL COSTS)
          WRITE (I2,210) (C(J),J=1,IVAR)
          WRITE (I2,240)
240  FORMAT (1H0,20X,12HOPTIMAL SOLN)
          WRITE (I2,210) (X(J),J=1,IVAR)
```

RETURN
END

```
/FTC      LIST,SIZE=256K
          SUBROUTINE OPT
          DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
          IB(30),X(100),IPDT(100),ICOST(30),COL(30),CS(100),Y(32,
X100)
          COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
          X,I,J,SUMU,
          ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE
          XGF,CNEGF,
          ZVARF,FAC,COSTF,RATIO,IBAD,NS,IPROB,I1,I2,NPOS,CS,N1
          KTR=0
          DO 10 I=1,ICON
10  ICOST(I)=IVAR+I
          N1=NREAL+1
          M1=ICON+1
          DO 20 I=1,ICON
          DO 20 J=1,NREAL
20  YU(I,J)=A(I,J)
          DO 30 I=1,ICON
30  YU(I,N1)=B(I)
          DO 40 J=1,NREAL
40  YU(M1,J)=-C(J)
          YU(M1,N1)=0.0
41  DO 100 J=1,NPOS
          KK=IPDT(J)
          IF (KK) 100,100,42
42  DO 50 I=NS,ICON
          IF (YU(I,KK)) 45,50,45
45  IF (ICOST(I)-IVAR) 50,50,55
50  CONTINUE
          GO TO 100
55  IR=I
          ICOST(IR)=KK
          TEMU=YU(IR,KK)
          YU(IR,KK)=1.0
          DO 70 I=1,M1
          IF (I-IR) 65,60,65
60  CGL(I)=1./TEMU-1.0
          GO TO 70
65  CGL(I)=-YU(I,KK)/TEMU
          YU(I,KK)=0.
70  CONTINUE
          DO 80 I=1,N1
          IF (I-KK) 75,80,75
75  SUMU=YU(IR,I)
```

```

      DO 80 K=1,M1
      YU(K,I)=YU(K,I)+SUMU*COL(K)
      IF (ABS(YU(K,I))-0.001) 78,78,80
78  YU(K,I)=0.0
      80 CONTINUE
      IPDT(J)=0
100  CONTINUE
      KTR=KTR+1
      IF (KTR-5) 110,125,125
110  DO 120 J=1,NPOS
      IF (IPDT(J)) 120,120,41
120  CONTINUE
125  IF (KTR-5) 129,126,126
126  WRITE (I2,127)
127  FORMAT (1H0,10X,17HBASIS NOT REACHED)
      IBAD=1
129  DO 200 J=1,ICON
      JJ=IVAR+J
      IF (YU(M1,JJ)) 250,200,200
200  CONTINUE
      RETURN
250  IBAD=2
      RETURN
      END

```

```

/FTC      LIST,SIZE=256K
          SUBROUTINE CBR
          DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
          IB(30),X(100),IPDT(100),ICOST(30),COL(30),CS(100),Y(32,
X100)
          COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
          X,I,J,SUMU,
          ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE
          XGF,CNEGF,
          ZVARF,FAC,COSTF,RATIO,IBAD,NS,IPROB,I1,I2,NPOS,CS,N1
          TOL=0.001
          IBAD=0
          NVAR=IVAR+ICON
          DO 10 J=1,NVAR
          CS(J)=C(J)
          C(J)=0.0
          DO 10 I=1,ICON
          Y(I,J)=YU(I,J)
10  YU(I,J)=0.0
          JCON=0
          DO 50 J=1,ICON
          JJ=IVAR+J
          DO 35 K=1,ICON

```

```

      IF (JJ-ICOST(K)) 35,50,35
35  CONTINUE
      JCON=JCON+1
      XBU(JCON)=YU(M1,JJ)-0.5
      JVAR=0
      DO 40 K=1,ICON
      IF (ICOST(K)-IVAR) 37,37,40
37  JVAR=JVAR+1
      YU(JCON,JVAR)=Y(K,JJ)
40  CONTINUE
50  CONTINUE
      DO 60 I=1,NPOS
      IF (XBU(I)) 56,57,58
56  XBU(I)=-XBU(I)
57  TYPE(I)=-1.0
      GO TO 60
58  DO 59 J=1,NPOS
59  YU(I,J)=-YU(I,J)
      TYPE(I)=1.0
60  CONTINUE
      DO 65 J=1,NPOS
65  G(J)=-1.0
      M=NPOS
      NREAL=NPOS
      RETURN
      END

```

```

/FTC      LIST,SIZE=256K
          SUBROUTINE SIMPL(NPHAS)
          DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
          IB(30),X(100),IPDT(100),ICOST(30),CQL(30),CS(100),Y(32,
X100)
          COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
          X,I,J,SUMU,
          ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE
          XGF,CNEGF,
          ZVARF,FAC,COSTF,RATIO,IBAD,NS,IPROB,I1,I2,NPOS,CS,N1
          BIGGG=1.E+8
          5  M1=M+1
              IF (NPHAS-1) 250,10,250
          10  N=NREAL
          60  DO 100 I=1,M
              IV(I)=0
              IF (TYPE(I)) 80,100,70
          70  IV(I)=N+1
          80  N=N+1
              DO 90 J=1,M
          90  YU(J,N)=0.

```



```
      YU(I,N)=TYPE(I)
100  CONTINUE
      JZ=0
      DO 130 J=JZ,N
      YU(M1,J)=0.
      DO 120 I=1,M
      IF (IV(I)) 120,110,120
110  YU(M1,J)=YU(M1,J)-YU(I,J)
120  CONTINUE
130  CALL ZERO(YU(M1,J))
150  IF (XBU(M1)) 160,200,200
160  CALL ENTER
      IF (KK)170,500,170
170  CALL LEAVE(0)
      CALL TRANS
      JCTR=JCTR+1
      GO TO 150
200  CONTINUE
250  JZ=0
      DO 280 J=JZ,N
      SUMU=0.
      DO 270 I=1,M
      IF (IV(I)) 260,270,260
260  KK=IV(I)
      SUMU=SUMU+YU(I,J)*C(KK)
270  CONTINUE
      YU(M1,J)=SUMU-C(J)
      CALL ZERO(YU(M1,J))
280  CONTINUE
300  CALL ENTER
      IF (KK) 310,400,310
310  CALL LEAVE(1)
      IF (IR) 320,330,320
320  CALL TRANS
      JCTR=JCTR+1
      GO TO 300
330  XBU(M1)=BIGGG
400  RETURN
500  DO 520 I=1,M
      IF (IV(I)) 520,510,520
510  IF (XBU(I)) 530,520,530
520  CONTINUE
      GO TO 200
530  WRITE (I2,540)
540  FORMAT (1H0,5X,20HNO FEAS SOLN PHASE 2)
      IBAD=1
      RETURN
      END
```

```

/FTC      LIST,SIZE=256K
          SUBROUTINE ENTER
          DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
          1B(30),X(100),IPDT(100),ICOST(30),COL(30),CS(100),Y(32,
X100)
          COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
X,I,J,SUMU,
          ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE
XGF,CNEGF,
          2VARF,FAC,COSTF,RATIO,IBAD,NS,IPOB,I1,I2,NPOS,CS,N1
          5 DO 10 J=1,N
            IF (YU(M1,J)) 20,10,10
          10 CONTINUE
            KK=0
            RETURN
          20 KK=J
            DO 40 J=KK,N
              IF (YU(M1,KK)-YU(M1,J)) 40,40,30
            30 KK=J
          40 CONTINUE
            RETURN
          END

```

```

/FTC      LIST,SIZE=256K
          SUBROUTINE LEAVE(PHASE)
          DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
          1B(30),X(100),IPDT(100),ICOST(30),COL(30),CS(100),Y(32,
X100)
          COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
X,I,J,SUMU,
          ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE
XGF,CNEGF,
          2VARF,FAC,COSTF,RATIO,IBAD,NS,IPOB,I1,I2,NPOS,CS,N1
          BIGGG=1.E8
          5 IR=0
            TEMU=BIGGG
            DO 70 I=1,M
              IF (YU(I,KK)) 10,70,40
            10 IF (PHASE) 20,70,20
            20 IF (IV(I)) 70,30,70
            30 IR=I
              RETURN
            40 SUMU=XBU(I)/YU(I,KK)
              IF (TEMU-SUMU) 70,70,60
            60 IR=I
              TEMU=SUMU
            70 CONTINUE

```

RETURN
END

```

/FTC      LIST,SIZE=256K
          SUBROUTINE TRANS
          DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
          IB(30),X(100),IPDT(100),ICOST(30),COL(30),CS(100),Y(32,
X100)
          COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
          X,I,J,SUMU,
          ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE
          XGF,CNEGF,
          ZVARF,FAC,COSTF,RATIO,IBAD,NS,IPROB,I1,I2,NPOS,CS,N1
          DIMENSION PHIU(32)
5          IV(IR)=KK
          TEMU=YU(IR,KK)
          YU(IR,KK)=1.
          DO 30 I=1,M1
            IF (I-IR) 20,10,20
10         PHIU(I)=1./TEMU-1.
            GO TO 30
20         PHIU(I)=-YU(I,KK)/TEMU
            YU(I,KK)=0.
30         CONTINUE
            JZ=0
            DO 50 J=JZ,N
              IF (J-KK) 40,50,40
40             SUMU=YU(IR,J)
              DO 50 I=1,M1
                YU(I,J)=YU(I,J)+SUMU*PHIU(I)
                CALL ZERO(YU(I,J))
50             CONTINUE
            RETURN
          END

```

```

/FTC      LIST,SIZE=256K
          FUNCTION ZERO(UP)
          DIMENSION YU(32,100),XBU(32),IV(32),C(100),TYPE(32),A(
X30,90),
          IB(30),X(100),IPDT(100),ICOST(30),COL(30),CS(100),Y(32,
X100)
          COMMON DUMMY,C,XBU,YU,IV,TYPE,M,NREAL,NFREE,N,M1,KK,IR
          X,I,J,SUMU,
          ITEMU,TOL,A,B,X,IPDT,ICOST,COL,ICON,IVAR,INITL,DENS,RNE

```

```
XGF,CNEGF,  
2VARF,FAC,COSTF,RATIO,IBAD,NS,IPROB,I1,I2,NPOS,CS,N1  
5 IF (ABS(UP)-TOL) 10,10,20  
10 UP=0.  
20 ZERO=UP  
RETURN  
END
```

```
/FTC LIST,SIZE=256K  
FUNCTION RANF(N)  
IF (N) 30,20,30  
30 IF (N/2*2-N) 50,90,50  
90 NN=N+1  
GO TO 50  
20 IF (NN) 50,40,50  
40 NN=162526457  
50 NN=NN*65539  
IF (NN) 60,70,70  
60 NN=NN+2147483647+1  
70 RANF=NN  
RANF=RANF*.4656613E-9  
RETURN  
END
```

References

- [1] M. L. Balinski - "Integer Programming: Methods, Uses, Computation", Management Science 12 No. 3 (1965).
- [2] R. E. Echols - "The Solution of Integer Linear Programming Problems by Direct Search", M. S. thesis, Washington University (1966).
- [3] R. E. Echols and L. Cooper - "The Solution of Integer Linear Programming Problems by Direct Search", Report No. AM66-1, Department of Applied Mathematics and Computer Science, Washington University (1966).
- [4] R. Hooke and T. A. Jeeves - "Direct Search Solution of Numerical and Statistical Problems", Journal of Ass'n. for Computing Machinery 8 No. 2 (1961).
- [5] H. Glass and L. Cooper - "Sequential Search: A Method for Solving Constrained Optimization Problems", Journal of Ass'n. for Computing Machinery 12 No. 1 (1965).
- [6] R. A. Cook - "An Algorithm for Integer Linear Programming", D.Sc. dissertation, Washington University (1966).