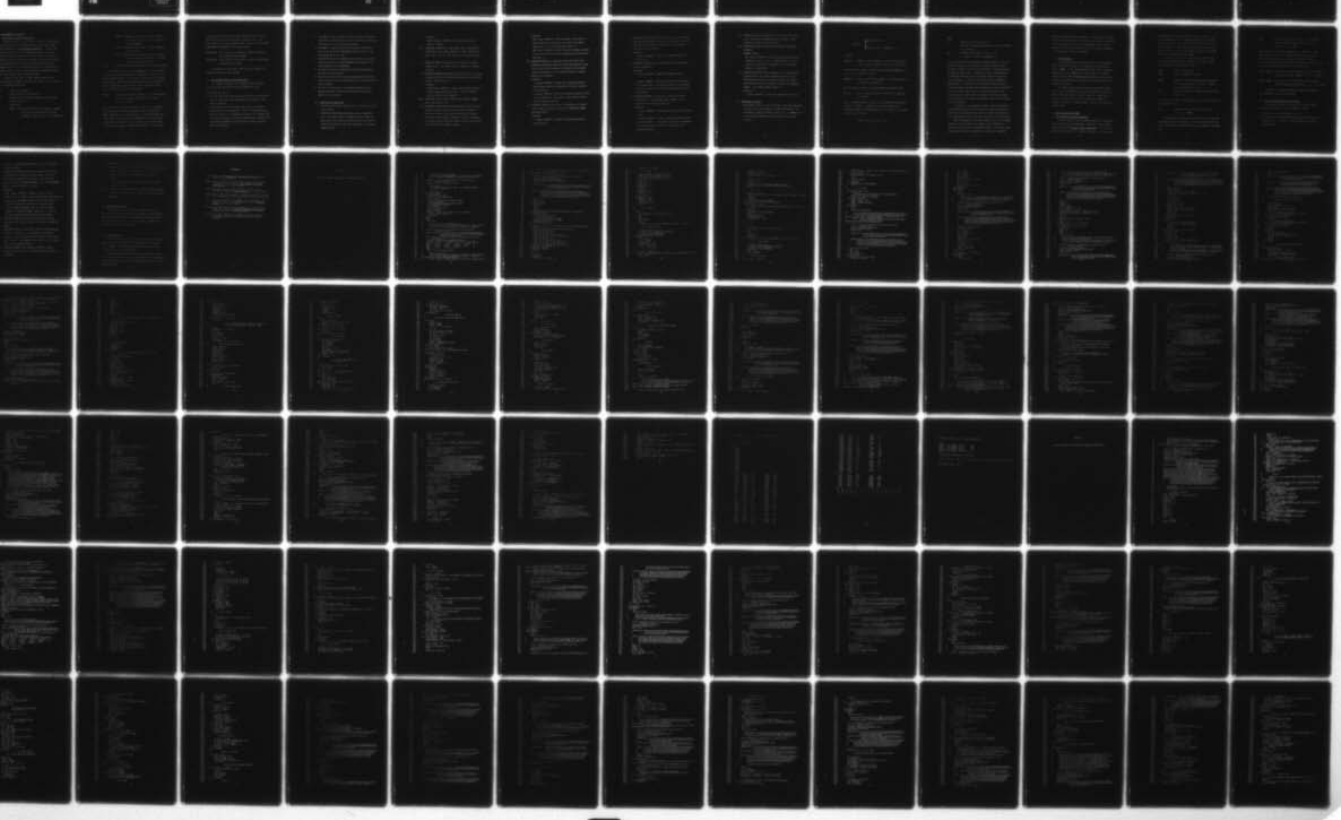
AD-A038 322 STANFORD UNIV CALIF SYSTEMS OPTIMIZATION LAB F/G 9/2
COMPUTER PROGRAMS FOR DECOMPOSITION IN INTEGER PROGRAMMING.(U)
SEP 76 G A KOCHMAN N00014-76-C-0418
UNCLASSIFIED SOL-76-20 ARO-12215.13-M NL

1 OF 2
AD A038 322



COMPUTER PROGRAMS FOR DECOMPOSITION IN INTEGER PROGRAMMING

BY

GARY A. KOCHMAN

TECHNICAL REPORT SOL 76-20
SEPTEMBER 1976

Systems Optimization Laboratory

Department of Operations Research

Stanford University

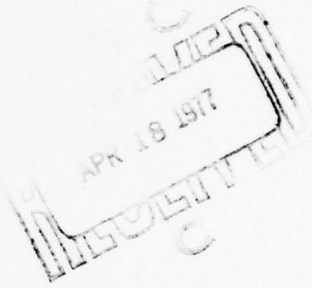
Stanford California 94305

COMPUTER PROGRAMS FOR DECOMPOSITION IN INTEGER PROGRAMMING

by

Gary A. Kochman

Technical Report SOL 76-20
September 1976



SYSTEMS OPTIMIZATION LABORATORY
DEPARTMENT OF OPERATIONS RESEARCH

Stanford University
Stanford, California

Research and reproduction of this report were partially supported by the Office of Naval Research under Contract N00014-76-C-0418; the Army Research Office Grant DAAC 29 74 C 0034; the Air Force Office of Scientific Research F44620-74-C-0079.

1. Introduction

The computer codes DSLC and DMLC, written to implement the decomposition algorithms developed by Kochman (1976), are documented and listed in this report. These programs are designed to solve the block angular integer programming problem,

$$(P) \quad \text{maximize} \quad \sum_{j=1}^{N} c^{(j)} x^{(j)}, \tag{1}$$

$$\text{subject to} \quad \sum_{j=1}^{N} A^{(j)} x^{(j)} \leq b \tag{2}$$

$$B^{(j)} x^{(j)} \leq \beta^{(j)} \qquad j = 1, 2, \cdots, N \tag{3}$$

$$L^{(j)} \leq x^{(j)} \leq U^{(j)}, \quad x^{(j)} \text{ integer}, \quad j = 1, 2, \cdots, N, \tag{4}$$

by decomposition into subproblems. The routine DSLC exploits certain special structures which arise when (P) has only a single linking constraint (2), i.e., when the constraint matrices $A^{(j)}$ $(1 \leq j \leq N)$ have only a single row. The routine DMLC treats the multiple linking constraints case. See Kochman (1976) for a detailed discussion of these decomposition algorithms; a step-by-step statement of the algorithms is presented in Section 5.2 there.

In addition to the programs DSLC and DMLC, the integer linear programming routine BB is documented and listed here. The computer code BB is designed to solve the general (unstructured) integer program,

$$(Q) \qquad \text{maximize} \quad cx$$

$$\text{subject to} \quad Ax \leq b$$

$$L \leq x \leq U, \quad x \text{ integer},$$

by branch-and-bound search. Specifically, the branch-and-bound algorithm BB follows the ideas of Dakin (1965). Tomlin's (1970) improved penalty calculations are used to guide the branching process, and the last-in-first-out (LIFO) rule is used to retrieve stored nodes in the search tree when backtracking occurs[1].

The code BB is embedded in both the routines DSLC and DMLC for the purpose of solving the integer programming subproblems as required by the decomposition procedures. It should be noted, however, that the routine BB itself uses only standard methods. This code does not represent a new contribution to the field of integer programming; its documentation is included here for the sake of completeness.

---

[1] For further references, see Beale (1968), Beale and Small (1965), and Little, Murty, Sweeney, and Karel (1963).

The linear programming portions of all three programs are based extensively on the Fortran code LPM-1, developed by Dr. John A. Tomlin of the Systems Optimization Laboratory, Department of Operations Research, Stanford University. LPM-1 is designed to implement the revised simplex method, using the product-form of the inverse. In adopting this code for use in the routines BB, DSLC, and DMLC, the present author has modified LPM-1 appropriately to incorporate the bounded-variable technique as well. LPM-1 has been documented by Tomlin (1975).

The three codes BB, DSLC, and DMLC are written in FORTRAN IV for Stanford's IBM 370/168, with all instructions and data in-core. Each code is WATFIV compatible as well. The codes may be machine-dependent, however, insofar as each code makes calls to the system subroutine PCLOCK. In each program, this subroutine is used to compute the total time the IBM 370/168 Central Processing Unit was available to the core partition containing the job. In addition, PCLOCK is used to determine the computation times required for various intermediate tasks performed by the algorithms.

The various subprograms (exclusive of the main programs) which comprise each of the three codes are listed in Table I. These subroutines are described in greater detail in the remaining sections of this report. More specifically, the branch-and-bound code BB is discussed in Section 2, while the decomposition codes DSLC and DMLC are discussed in Sections 3 and 4, respectively. Listings of the three programs, and the output results obtained in sample runs of each are given in Appendices I-III.

TABLE I

Subprograms for Computer Codes BB, DSLC, DMLC

BB	DSLC	DMLC
BLOCK DATA	BLOCK DATA	BLOCK DATA
INPUT(IFPROB, INITBD)	INPUT(IFPROB)	INPUT(IFPROB)
FTRAN(IPAR)	FTRAN(IPAR)	FTRAN(IPAR)
BTRAN	BTRAN	BTRAN
FORMC	FORMC	FORMC
PRICE	PRICE	PRICE
CHUZR	CHUZR	CHUZR
WRETA	WRETA	WRETA
SHIFTR(IOLD, INEW)	SHIFTR(IOLD, INEW)	SHIFTR(IOLD, INEW)
INVERT	INVERT	INVERT
UNPACK(IV)	UNPACK(IV)	UNPACK(IV)
SHFTE	SHFTE	SHFTE
UPBETA	UPBETA	UPBETA
NORMAL(ITSINV)	NORMAL	NORMAL
BANDB(INITBD, ITSINV)	BANDB(INITBD)	BANDB(INITBD)
DCHUZR	DCHUZR	DCHUZR
DCHUZC	DCHUZC	DCHUZC
TESTX	TESTX	TESTX
PENLTS	PENLTS	PENLTS
BRANCH	BRANCH	BRANCH
BKTRAK	BKTRAK	BKTRAK
WRAPUP		
	ALOCTE(IPAR, IRHS)	ALOCTE(IPAR)
		LDMSTR
		TESTZ
	DOMCHK(J, K)	DOMCHK(J, K)
	LDDATA(J)	LDDATA(J)
	GETLBD(J)	GETLBD(J)
		PISAVR(J)
	SAVER(J)	SAVER(J)
	DMBRAN(J, K)	DMBRAN(J, K)
	BBKTRK	BBKTRK
	OUTSOL(INC)	OUTSOL(INC)
	PHASE1(J)	

2. The Branch-and-Bound Code BB

2.1. Main Program and Input Requirements

The main program for the branch-and-bound code BB serves to coordinate the application of the input (INPUT), linear programming (NORMAL), branch-and-bound (BANDB), and output (WRAPUP) subroutines. The first card of input must contain values for the variables INPROB, IOBJ, INVFRQ, ITRFRQ, and INITBD in 4I4, I10 format. These variables have the following significance:

INPROB = problem identification number. (Must be nonzero; otherwise, no computations are carried out and BB terminates immediately.)

IOBJ = row number for objective row. Automatically set to one by default if IOBJ is omitted.

INVFRQ = Parameter determining frequency with which basis reinversion is carried out in the linear programming portions of the code; if INVFRQ = k, basis reinversion will be carried out after every k (primal and/or dual) simplex iterations. INVFRQ should not be set greater than the number of rows in the problem.

ITRFRQ = upper limit on total iterations; all computations are terminated if the total number of simplex iterations exceeds ITRFRQ.

INITBD = initial lower bound estimate for maximum objective function value. The value of INITBD is used in fathoming tests until the first integer-feasible incumbent solution is found with objective value greater than INITBD.

BB assumes all cost coefficients are integer-valued, and takes advantage of this by rounding all upper bounds down to the nearest integer. The formats used for input of row and column names, objective function coefficients, constraint matrix coefficients, and right-hand side coefficients are identical to those used by LPM-1. (This format is consistent with standard MPS format.) Following input of these data, initial lower and upper bounds XLB(J), XUB(J) for all decision variables must be input in 15F5.0 format.

The input for a sample run of BB is included in Appendix I, following the listing of the program.

2.2. Restrictions Relevant to the Use of BB

As currently configured, the following restrictions apply to the input data and problem size and complexity for BB:

1. The number of rows (including the objective function row) must be .LE. 60; the number of columns (including slack and artificial variables generated by BB) must be .LE. 122.
2. The number of nonzero objective and constraint coefficients must be .LE. 2000.

3. The list of stored nodes generated in the course of the branch-and-bound search must never exceed 500 nodes in length.

2.3. Description of Subprograms

1. BLOCK DATA--From LPM-1; sets initial values for a number of variables used throughout the program. (See Tomlin (1975).)
2. SUBROUTINE INPUT(IFPROB, INITBD)--From LPM-1; reads in all problem data. (See Tomlin (1975).) Modified to also read in lower and upper bounds on decision variables.

   Parameters:

   IFPROB (Output, INTEGER*4) Problem identification number (=INPROB)

   INITBD (Output, INTEGER*4) Initial lower bound estimate for maximum objective value.
3. SUBROUTINE FTRAN(IPAR)--From LPM-1; performs forward transformation of matrix columns by basis inverse. (See Tomlin (1975).)

   Parameters:

   IPAR (Input, INTEGER*4) Determines eta-vectors by which matrix column is updated.
4. SUBROUTINE BTRAN--From LPM-1; performs backward transformation. (See Tomlin (1975).)
5. SUBROUTINE FORMC--From LPM-1; forms objective function vector for Phase I of primal simplex method. Used in conjunction with SUBROUTINE PRICE. (See Tomlin (1975).) Modified for bounded-variables.

6. SUBROUTINE PRICE--From LPM-1; prices out columns for primal simplex method and chooses pivot column. (See Tomlin (1975).) Modified for bounded-variables.

7. SUBROUTINE CHUZR--From LPM-1; chooses pivot row for primal simplex method. (See Tomlin (1975).) Modified for bounded-variables.

8. SUBROUTINE WRETA--From LPM-1; forms new eta-vectors for product-form of basis inverse. (See Tomlin (1975).)

9. SUBROUTINE SHIFTR(IOLD, INEW)--From LPM-1; rearranges data storage. (See Tomlin (1975).)

   Parameters:

   IOLD, INEW (Both input, both INTEGER*4) IOLD indexes storage locations of data to be transferred by SHIFTR to locations indexed by INEW.

10. SUBROUTINE INVERT--From LPM-1; Determines basis inverse by LU decomposition. (See Tomlin (1975).) Modified to update RHS with bounded-variables.

11. SUBROUTINE UNPACK(IV)--From LPM-1; expands compressed matrix columns by inserting zeroes appropriately. (See Tomlin (1975).)

    Parameters:

    IV (Input, INTEGER*4) Indexes the matrix column to be expanded.

12. SUBROUTINE SHFTE--From LPM-1; subroutine for INVERT. (See Tomlin (1975).)

13. SUBROUTINE UPBETA--From LPM-1; updates RHS following a primal or dual simplex pivot. (See Tomlin (1975).) Modified for bounded-variables.

14. SUBROUTINE NORMAL(ITSINV)--From LPM-1; directs execution of primal simplex method. (See Tomlin (1975).) Modified for bounded-variables.

    Parameters:

    ITSINV (Input/Output, INTEGER*4) Counts number of simplex iterations since last basis inversion.

15. SUBROUTINE BANDB(INITBD, ITSINV)--directs execution of branch-and-bound search for optimal integer solution, after optimal LP solution has been computed in SUBROUTINE NORMAL. Also directs reoptimization via dual simplex method following a branch in the forward direction. Calls NORMAL for reoptimization after backtracking.

    Parameters:

    INITBD (Input, INTEGER*4) Initial lower bound estimate for maximal objective value.

    ITSINV (Input/Output, INTEGER*4) Counts number of simplex iterations since last basis inversion.

16. SUBROUTINE DCHUZR--Selects pivot row for dual simplex method (analogous to CHUZR).

17. SUBROUTINE DCHUZC--Selects pivot column for dual simplex method (Analogous to PRICE).

18. SUBROUTINE TESTX--Tests optimal LP solution at each node for fathoming. Installs new incumbent each time an improved integer-feasible solution is found.

19. SUBROUTINE PENLTS--Computes Tomlin's improved up- and down-penalties and the Gomory penalty at each node. Checks for forced branches on both basic and non-basic decision variables. If computed penalties are insufficient for fathoming, chooses branching variable, calls SUBROUTINE BRANCH.

20. SUBROUTINE BRANCH--Performs necessary bookkeeping for branching as indicated by SUBROUTINE PENLTS. Increments list of stored nodes; revises bounds on branching variable appropriately for branch in forward direction.

21. SUBROUTINE BKTRAK--Uses last-in-first-out (LIFO) strategy to retrieve unfathomed nodes from list of stored nodes.

22. SUBROUTINE WRAPUP--Outputs final solution information. (See output from sample run in Appendix I.)

2.4. Description of Output

The output from BB takes the following form. Whenever a new incumbent solution is found, the computation time required to reach the solution, and the corresponding objective value of the solution are printed out. When all computations are completed (i.e., when the optimal solution has been obtained, and its optimality verified), the total solution time is output. The optimal integer solution is then printed out in 2015 format. More specifically, the optimal

column values are output in the order (1) objective function value, (2) slack variables, and (3) decision variables. The values of the first twenty columns are printed on the first line, the second twenty on the second line, etc., until all column values have been printed. The output from a sample run of BB is given in Appendix I, following the listing of the program.

2.5. Further Comments

As currently written, the Fortran code BB contains approximately 1,450 source statements. All significant array storage is done in COMMON. In all, approximately 64K bytes of core storage are required by the three COMMON blocks. When compiled on Stanford's IBM 370/168 under IBM's FORTRAN-H-extended compiler, with level two optimization, approximately 113K bytes of core are required for all instruction and data storage.

The computer program BB was written by the present author in the course of completing his doctoral dissertation in the Department of Operations Research, Stanford University. Computational results (solution times) obtained on several test problems are presented in Chapter 6 of Kochman (1976). BB was last revised in September 1976.

3. The Decomposition Code DSLC

3.1. Main Program and Input Requirements

The main program for the code DSLC coordinates the application of the two phases of the decomposition algorithm for block angular integer programs with a single linking constraint. In Phase I, the problem data is input, and the computations required to construct the Master Problem are carried out. The branch-and-bound search for an optimal allocation of the linking resources is executed in Phase 2. DSLC has been written so that any of the four decision rules for subproblem selection may be specified by the user. (See Section 5.2 in Kochman (1976).) This is done by assigning the appropriate value to the input variable IPARM, as described below.

The first card of input for DSLC contains values for the variables NSUBS, IB, IPARM, and IZINC in 4I10 format. These variables have the following significance:

NSUBS = number of subproblems.

IB = linking constraint right-hand side.

IPARM = parameter determining which subproblem selection rule is used in Phase 2.

Specifically,

IPARM = 1 ⇒ Selection rule based on average gaps between LP bounds and corresponding exact integer programming objective values in the subproblems.

IPARM = 2 ⇒ Selection rule based on average slack in the linking constraint in the optimal subproblem solutions.

IPARM = 3 ⇒ Selection rule based on average subproblem solution times.

IPARM = 4 ⇒ First unfixed subproblem will be selected, where subproblems are numbered in the same order in which they were input.

IZINC = lower bound estimate for maximum overall objective value.

Following the first card, the problem data for the individual subproblems is read in sequentially. The format of this data is the same for each subproblem, and is similar to the format used to input problem data in the branch-and-bound code BB described in Section 2.1. For each subproblem, the first card of input contains the value for the variable INPROB in I4 format, where

INPROB = subproblem identification number. The subproblems must be numbered and input sequentially in the order 1, 2, 3, ..., NSUBS.

DSLC assumes that all cost coefficients in each of the subproblems are integer-valued, and takes advantage of this by rounding all upper bounds down to the nearest integer. For each subproblem, the formats used for input of row and column names, objective function coefficients, constraint matrix coefficients, and right-hand side coefficients are identical to those used by LPM-1. Following input

of these data, initial lower and upper bounds XLB(J), XUB(J) must be specified for the decision variables in 15F5.0 format. The final input card for each subproblem contains values for the variables IBLB(INPROB), IBUB(INPROB) in 2I10 format, where

IBLB(INPROB) = lower bound on the allocation variable associated with subproblem number INPROB.

IBUB(INPROB) = upper bound on the allocation variable associated with subproblem number INPROB.

The input for a sample run of DSLC is given in Appendix II, following the listing of the program.

3.2. Restrictions Relevant to the Use of DSLC

As currently configured, the following restrictions apply to the input data and problem size and complexity for DSLC:

1. NSUBS must be .LE. 10.
2. The first row in each subproblem must correspond to the objective row; the second row in each subproblem must correspond to the linking constraint.
3. No right-hand side value should be specified for the linking constraint row in each subproblem. Prior to computing the parametric LP solution for a subproblem in Phase 1, DSLC automatically initializes the linking constraint right-hand side to the specified lower bound IBLB(INPROB) on the allocation variable associated with the subproblem.

4. The number of rows (including objective and linking constraint rows) in each subproblem must be .LE. 22; the number of decision variables must be .LE. 50 in each subproblem.
5. The number of nonzero objective and constraint coefficients must be .LE. 500 in each subproblem; the total number in all subproblems must be .LE. 1000.
6. The modified concave and piecewise linear parametric objective function generated in SUBROUTINE PHASE1(J) must have .LE. 50 linear segments in each subproblem.
7. The number of distinct subproblem solutions generated must be .LE. 200 for each subproblem.
8. The list of stored nodes in the branch-and-bound search for an optimal allocation of the linking resource must never exceed 200 nodes in length.
9. The list of stored nodes in the branch-and-bound search for optimal subproblem solutions must never exceed 500 nodes in length for any subproblem.

3.3. Description of Subprograms

1. BLOCK DATA--From LPM-1; same as for BB. (See Section 2.3 and Tomlin (1975).)
2. SUBROUTINE INPUT(IFPROB)--From LPM-1; reads in subproblem data. (See Tomlin (1975).) Modified to read in lower and upper bounds on the decision variables and on the allocation variable associated with the given subproblem. Also stores subproblem data.

Parameters:

IFPROB (Output, INTEGER*4) Subproblem identification number (= INPROB).

3-13. SUBROUTINES FTRAN(IPAR), BTRAN, FORMC, PRICE, CHUZR, WRETA, SHIFTR(IOLD, INEW), INVERT, UNPACK(IV), SHFTE, UPBETA--From LPM-1; same as for BB. (See Section 2.3, and Tomlin (1975).)

14. SUBROUTINE NORMAL--From LPM-1; directs execution of primal simplex method. (See Tomlin (1975).) Modified for bounded-variables.

15. SUBROUTINE BANDB(INITBD)--From BB; directs execution of branch-and-bound search for an optimal integer solution to given subproblem. (See Section 2.3.)

Parameters:

INITBD (Input, INTEGER*4) Initial lower bound estimate for maximum subproblem objective value. Value of INITBD is specified by SUBROUTINE GETLBD(J).

16-21. SUBROUTINES DCHUZR, DCHUZC, TESTX, PENLTS, BRANCH, BKTRAK--From BB (See Section 2.3).

22. SUBROUTINE ALOCTE(IPAR, IRHS)--Solves Master Problem (concave-separable integer program; see Kochman (1976), Chapter 4). When current node in allocation-variables search tree was obtained by forward branch, the Master Problem is reoptimized by starting from the previously optimal solution and reallocating IRHS units of the linking resource.

Parameters:

IPAR (Input, INTEGER*4) Indicates whether current node in allocation-variables search tree was obtained by forward branch (IPAR = 1), or by backtracking (IPAR = 0).

IRHS (Input, INTEGER*4) Amount of slack in linking constraint in the optimal solution to last subproblem solved; used only when IPAR = 1.

23. SUBROUTINE DOMCHK(J, K)--Compares current allocation (from SUBROUTINE ALOCTE(IPAR, IRHS)) with previously examined allocations in each unfixed subproblem to determine if the optimal solution corresponding to the current allocation is already known for any subproblem.

Parameters:

J (Output, INTEGER*4) Indexes first subproblem found for which optimal solution corresponding to current allocation is already known.

K (Output, INTEGER*4) Indexes the previously examined allocation in subproblem J which leads to same optimal solution as would current allocation.

24. SUBROUTINE LDDATA(J)--Loads data for subproblem J into appropriate storage locations for solution by SUBROUTINES NORMAL and BANDB(INITBD).

Parameters:

J (Input, INTEGER*4) Indexes the subproblem selected to be solved next.

25. SUBROUTINE GETLBD(J)--Compares current allocation in subproblem J with previously examined allocations to determine an initial lower bound on the maximal objective value in subproblem J under the current allocation.

Parameters:

J (Input, INTEGER*4) Indexes the subproblem selected to be solved next.

26. SUBROUTINE SAVER(J)--Stores allocation and solution information for subproblem J.

Parameters:

J (Input, INTEGER*4) Indexes last subproblem solved.

K (Input, INTEGER*4) Indexes a previously examined allocation in subproblem J. The current allocation in subproblem J is fixed at the level of the K^{th} allocation by the branching procedure.

27. SUBROUTINE DMBRAN(J, K)--Branches on allocation variables in subproblem J. (See Kochman (1976), Chapter 3 for a discussion of the branching procedure.)

Parameters:

J (Input, INTEGER*4) Indexes subproblem in which branching is done.

K (Input, INTEGER*4) Indexes a previously examined allocation in subproblem J. The current allocation in subproblem J is fixed at the level of the K allocation by the branching procedure.

28. SUBROUTINE BBKTRK--Backtracks in allocation-variables search tree to locate next unfathomed node to be examined. Last-in-first-out (LIFO) selection rule used.

29. SUBROUTINE OUTSOL(INC)--Outputs final solution information.

Parameters:

DIMENSION INC(10)

INC (Input, INTEGER*4) Array containing optimal solution information. For J = 1, 2, ..., NSUBS, INC(J) = k if the solution in subproblem J corresponds to the K^{th} solution obtained in subproblem J.

30. SUBROUTINE PHASE1(J)--Solves subproblem J as a linear program parametrically in the linking constraint right-hand side. Simultaneously modifies the resulting concave and piecewise linear parametric objective function to render all breakpoints integral. (See Kochman (1976), Chapter 4.)

Parameters:

J (Input, INTEGER*4) Indexes next subproblem to be solved.

3.4. Description of Output

The following output is generated by DSLC. After each subproblem J has been solved parametrically in SUBROUTINE PHASE1(J), the (all-integer) breakpoints IBP(K, J) (K = 1, 2, ..., NLAMDA(J) + 1) of the resulting parametric objective function are printed out, where

$$IBP(K, J) = \begin{cases} IBLB(J), & \text{when } K = 1 \\ IBUB(J), & \text{when } K = NLAMDA(J) + 1, \end{cases}$$

and where

NLAMDA(J) = number of linear segments in the concave and piecewise linear parametric objective function for subproblem J.

The second line of output for each subproblem displays RLAMADA(K, J) (K = 1, 2, ..., NLAMDA(J)), where

RLAMDA(K, J) = slope of the K^{th} linear segment of parametric objective functions for subproblem J.

The third line of output for each subproblem gives ZNAUT(J), where

ZNAUT(J) = value of parametric objective function for subproblem J at IBLB(J).

Next, the output generated in Phase 2 of the decomposition algorithm is printed. Whenever a subproblem is (re)solved explicitly by SUBROUTINES NORMAL and BANDB(INITBD), one line of output is generated; this line takes the form,

SUB J, SOL K: X_1 X_2, ..., X_n,

where

J = subproblem identification number

K = identification number associated with the given solutions for subproblem J.

X_ℓ = optimal column values, ℓ = 1, 2, ..., n,

n = number of columns in subproblem J.

As in the branch-and-bound code BB, the optimal solution to the subproblem is printed out in the order (1) objective function value, (2) slack variables, and (3) decision variables. It should be noted that when the lower bound on the maximal objective value generated in SUBROUTINE GETLBD(J) equals the optimal objective value for the given subproblem, no incumbent solutions are found in SUBROUTINE BANDB(INITBD). In this case, only the optimal objective function value and linking constraint slack are printed out; zeroes are printed for the remaining column values. The remaining optimal column values may be recovered through the solution identification number K, by referring to the output for subproblem J when the K^{th} solution was first generated.

In addition to the subproblem solution information, whenever a new incumbent solution to the overall block angular problem (P) is discovered, the computation time required to reach the solution and the corresponding incumbent objective value IZINC in (P) are printed.

When all computations have been completed, the overall optimal solution to (P) is printed. The solution in each subproblem is identified by the solution identification number. The optimal column values may be recovered by referring to the output of subproblem

solution information generated in Phase 2 of the algorithm. Finally, the maximal objective value to (P) is printed, followed by the output of various parameters related to the performance of DSLC on the problem. The output from a sample run of DSLC is given in Appendix II, following the listing of the program.

3.5. Further Comments

As currently written the Fortran code DSLC contains approximately 1,950 source statements. All significant array storage is done in COMMON. In all, approximately 65K bytes of core storage are required by the five COMMON blocks. When compiled on Stanford's IBM 370/168 under IBM's FORTRAN-H-Extended compiler, with level two optimization, approximately 126K bytes of core are required for all instructions and data.

The computer program DSLC was written by the present author in the course of completing his doctoral dissertation in the Department of Operations Research, Stanford University. Computational results (solution times) obtained on several test problems are presented in Chapter 6 of Kochman (1976). DSLC was last revised in September 1976.

4. The Decomposition Code DMLC

4.1. Main Program and Input Requirements

The main program for the code DMLC coordinates the application of the two phases of the decomposition algorithm for block angular integer programs with multiple linking constraints. As in DSLC, the problem data is input and the computations required to construct the

Relaxed Master Problem are executed in Phase 1. In Phase 2, the branch-and-bound search for an optimal allocation of the linking resources is carried out. The major differences between the codes DSLC and DMLC are in the methods used to construct and solve the Master Problem; see Chapter 4 and Chapter 5, Section 1 in Kochman (1976) for a discussion of these methods.

The first card of input for DMLC contains values for the variables NSUBS, NLINK, IPARM, and IZINC in 4I10 format. These variables have the following significance:

NSUBS = number of subproblems

NLINK = number of linking constraints

IPARM = parameter determining which subproblem selection rule is used in Phase 2. Identical to IPARM in the code DSLC; see Section 3.1.

IZINC = lower bound estimate for maximum overall objective value in (P).

The second card of input for DMLC contains values for the array IB(I) in 5I10 format, where

IB(I) = right-hand side for I^{th} linking constraint, I = 1, 2, ..., NLINK.

Following these two cards, the problem data for the individual subproblems is read in sequentially. For each subproblem, the first card of input contains the value for the variable INPROB in I4 format, where

INPROB = subproblem identification number. The subproblems must be numbered and input sequentially in the order 1, 2, 3, ..., NSUBS.

The format for the individual subproblem data is identical to that used in DSLC (see Section 3.1), with the exception of the input of the bounds on the allocation variables. The final two cards of input for each subproblem contains values for the arrays IBLB(I, INPROB) and IBUB(I, INPROB), respectively, in 5I10 format, where

IBLB(I, INPROB) = lower bound on the I^{th} allocation variable associated with subproblem number INPROB, I = 1, 2, ..., NLINK.

IBUB(I, INPROB) = upper bound on the I^{th} allocation variable associated with subproblem number INPROB, I = 1, 2, ..., NLINK.

The input for a sample run of DMLC is displayed in Appendix III, following the listing of the program.

4.2. Restrictions Relevant to the Use of DMLC

As currently configured, the following restrictions apply to the input data and problem size and complexity for DMLC:

1. NSUBS must be .LE. 10; NLINK must be .LE. 5.
2. The first row in each subproblem must correspond to the objective row; rows 2, 3, ..., NLINK + 1 must correspond to the linking constraint rows.

3. The initial right-hand side values specified for the linking constraints in each subproblem must be chosen so that, correspondingly, each subproblem has a feasible and finite optimal LP solution.

4. The number of rows (exclusive of objective and linking constraint rows) in each subproblem must be .LE. 10; the number of decision variables must be .LE. 20 in each subproblem.

5. The number of nonzero objective and constraint coefficients must be .LE. 500 in each subproblem; the total number in all subproblems must be .LE. 1000.

6. The number of distinct sets of optimal dual multipliers generated in each subproblem must be .LE. 20.

7. The number of distinct allocations explicitly examined in each subproblem must be .LE. 40.

8. The list of stored nodes in the branch-and-bound search for an optimal allocation of the linking resources must never exceed 200 nodes in length.

9. The list of stored nodes in the branch-and-bound search for optimal subproblem solutions must never exceed 500 nodes in length for any subproblem.

4.3. Description of Subprograms

1-21. BLOCK DATA and SUBROUTINES INPUT(IFPROB), FTRAN(IPAR), BTRAN, FORMC, PRICE, CHUZR, WRETA, SHIFTR(IOLD, INEW), INVERT, UNPACK(IV), SHFTE, UPBETA, NORMAI, BANDB(INITBD), DCHUZR, DCHUZC, TESTX, PENLTS, BRANCH, BKTRAK--Same as for DSLC; see

Section 3.3. SUBROUTINE INPUT(IFPROB) modified for multiple linking constraints.

22. SUBROUTINE ALOCTE(IPAR)--Obtains linear programming solution to Relaxed Master Problem; see Kochman (1976), Section 5.1. Calls SUBROUTINE LDMSTR (if necessary) to set up problem data for solution by SUBROUTINE NORMAL. Also calls SUBROUTINE TESTZ to round-off optimal LP solution.

    Parameters:

    IPAR (Input, INTEGER*4) Parameter indicating whether (IPAR = 0) or not (IPAR = 1) any subproblem has been explicitly solved since last Master Problem solution. When IPAR = 0, the data for the Master Problem must be reloaded from scratch (SUBROUTINE LDMSTR) into the storage locations required for SUBROUTINE NORMAL. When IPAR = 1, only the revised allocation-variables bounds need be reloaded.

23. SUBROUTINE LDMSTR--Loads data for Relaxed Master Problem into appropriate storage locations for solution by SUBROUTINE NORMAL.

24. SUBROUTINE TESTZ--Tests LP-optimal solution to Relaxed Master Problem for fathoming. If not fathomed, the solution is rounded to obtain an integer-feasible solution to the Relaxed Master Problem. See Kochman (1976), Section 5.1 for a discussion of the rounding procedure.

25-27. SUBROUTINES DOMCHK(J, K), LDDATA(J), GETLBD(J),--Same as for DSLC (see Section 3.3). Modified for multiple linking constraints.

28. SUBROUTINE PISAVR(J)--Tests current LP-optimal solution to subproblem J to determine if the set of optimal dual multipliers associated with the linking constraints is distinct from previous sets; stores these multipliers if distinct.

Parameters:

J (Input, INTEGER*4) Indexes subproblem solved under current allocation.

29-32. SUBROUTINES SAVER(J), DMBRAN(J, K), BBKTRK, OUTSOL(INC)--Same as for DSLC (see Section 3.3). Modified for multiple linking constraints.

4.4. Description of Output

The format for output from DMLC is identical to that for DSLC, with the obvious exception that no parametric linear programming solution information for the subproblems is generated in DMLC. The output from a sample run of DMLC is included in Appendix III, following the listing of the program.

4.5. Further Comments

As currently written, the Fortran code DMLC contains approximately 2,000 source statements. All significant array storage is done in COMMON. In all, approximately 50K bytes of core storage are required by the five COMMON blocks.

The computer program DMLC was written by the present author in the course of completing his doctoral dissertation in the Department of Operations Research, Stanford University. DMLC was last revised in September 1976.

RFFERENCES

Beale, E.M.L., 1968, Mathematical Programming in Practice, Wiley and Sons, Inc., New York.

Beale, E.M.L., and R.E. Small, 1965, "Mixed Integer Programming by a Branch-and-Bound Technique," Proceedings of the Third IFIP Congress, Vol. 2, pp. 450-451.

Dakin, R.J., 1965, "A Tree Search Algorithm for Mixed Integer Programming Problems," Computer Journal, Vol. 8, pp. 250-255.

Kochman, G.A., 1976, "Decomposition in Integer Programming," Technical Report, Department of Operations Research, Stanford University.

Little, J.D.C., K.C. Murty, B.W. Sweeney, and C. Karel, 1963, "An Algorithm for the Travelling Salesman Problem," Operations Research, Vol. 11, pp. 972-989.

Tomlin, J.A., 1970, "Branch-and-Bound Methods for Integer and Non-Convex Programming," in Integer and Nonlinear Programming, J. Abadie, ed., North Holland, Amsterdam.

Tomlin, J.A., 1975, "Users Guide for LPM-1," Systems Optimization Laboratory, Department of Operations Research, Stanford University.

APPENDIX I

LISTING AND SAMPLE INPUT/OUTPUT FOR COMPUTER PROGRAM BB

```
      C
      C         BRANCH-AND-BOUND ROUTINE, BB
      C         PURE-INTEGER LINEAR PROGRAMMING IN GENERAL INTEGER VARIABLES
      C         GARY A. KOCHMAN (OPERATIONS RESEARCH), SEPT. 1976, MOD 1
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            COMMON/TIMERS/ TFIRST,TOPT,IFVAL,ITOT
      C
      C         MAIN PROGRAM
      C         COORDINATES APPLICATION OF INPUT, LP, BRANCH-AND-BOUND,
      C         AND OUTPUT SUBROUTINES
      C
      C         START TIMER
         10 CALL PCLOCK(ITOT)
      C         INPUT PROBLEM DATA
            CALL INPUT(IFPROB,INITBD)
            IF (IFPROB .EQ. 0) GO TO 1000
            ITSINV = 99999
      C         SOLVE LP RELAXATION OF ORIGINAL PROBLEM
            CALL NORMAL(ITSINV)
      C         APPLY BRANCH-AND-BOUND SEARCH ROUTINE
            CALL BANDB(INITBD,ITSINV)
      C         STOP TIMER
            CALL PCLOCK(JTIME,ITOT)
            TTOT = JTIME/100.
            WRITE (6,1) TTOT
          1 FORMAT (' TOTAL SOLUTION TIME =',F7.2,' SECONDS')
      C         OUTPUT OPTIMAL SOLUTION
            CALL WRAPUP
            GO TO 10
       1000 STOP
            END
            BLOCK DATA
      C
      C         INITIALIZES GLOBAL PROGRAM CONSTANTS
      C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
            REAL ZTOLZE/1.E-5/,ZTOLPV/1.E-4/,ZTOST/1.E-3/,ZTOLRJ/1.E-4/,
           1     ZTOLSM/1.E-10/
            INTEGER*4 NRMAX/60/,NTMAX/1000/,NEMAX/4000/
            INTEGER*4 QRO/'ROW '/,QMA/'MATR'/,QBA/'BASI'/,QFI/'FIRS'/,
           1QEO/'EOF '/,QBL/'    '/,QA/'A   '/,QPL/' +  '/,QMI/' -  '/,
           2QZ/'  Z  '/,QI/'I   '/,QF/'F   '/,QN/'N   '/,QU/'U   '/,
           3QB/'B   '/,QC/'C   '/,QE/'E   '/,QH/'H   '/,QL/'L   '/,
           4QO/'O   '/,QR/'R   '/,QM/'M   '/,QG/'G   '/
            END
            SUBROUTINE INPUT(IFPROB,INITBD)
      C
      C         INPUTS PROBLEM DATA
      C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C********DESCRIPTION OF PARAMETERS********
      C         IFPROB = INPROB = NONZERO PROBLEM IDENTIFICATION NUMBER(OUTPUT)
```

```
      C         INITBD = INITIAL LOWER BOUND ESTIMATE FOR MAXIMAL OBJECTIVE
      C                  VALUE (OUTPUT)
      C*****************************************
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            DOUBLE PRECISION E(4000),ATEMP1,ATEMP2
            REAL A(2000)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEQ,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      C
            COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
           1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
           2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFREZ,JCOLP,NROW,NCOL,NELEM,
           3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
           4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
      C
            DO 10 I=1,60
         10 B(I) = 0.
            DO 20 J=1,122
         20 KINBAS(J) = 0
            NROW = 0
            ITCNT = 0
            ICS1=0
            ICS2=0
            READ (5,7000,END=9999) INPROB,IOBJ,INVFRQ,ITRFRQ,INITBD
       7000 FORMAT (4I4,I10)
            IFPROB = INPROB
            IF (IFPROB .EQ. 0) RETURN
            IF (IOBJ .EQ. 0)IOBJ = 1
            IF (INVFRQ .EQ. 0)INVFRQ = 99999
            IF (ITRFRQ .EQ. 0)ITRFRQ = 99999
            WRITE(6,8010) IFPROB
       8010 FORMAT(9H1PROBLEM ,I4)
      C
          5 READ(5,101) K1,K2,K3,K4,(NAME(I),I=1,4),ATEMP1,NAME(5),NAME(6),
           1ATEMP2
        101 FORMAT(4A1,2A4,2X,2A4,2X,F12.4,3X,2A4,2X,F12.4)
            IF(K1 .EQ. QE) GO TO 200
            IF(K1 .EQ. QBL) GO TO 50
            IF(K1 .EQ. QN) GO TO 100
            IF(K1 .EQ. QR .AND. K2 .EQ. QO) L=1
            IF(K1 .EQ. QR .AND. K2 .EQ. QO) GO TO 150
            IF(K1 .EQ. QC) L=2
            IF(K1 .EQ. QC) GO TO 150
            IF(K1 .EQ. QB .AND. K2 .EQ. QA) L=3
            IF(K1 .EQ. QB .AND. K2 .EQ. QA) GO TO 150
            IF(K1 .EQ. QF .AND. K2 .EQ. QH) L=4
            IF(K1 .EQ. QR .AND. K2 .EQ. QH) GO TO 150
         50 GO TO(210,320,410,500),L
        100 NTEMP(1) = NAME(3)
            NTEMP(2) = NAME(4)
        150 CONTINUE
            GO TO 5
      C
        210 NROW=NROW+1
            NCOL=NROW
            ICNAM(NROW,1) = NAME(1)
```

```
122.            ICNAM(NROW,2) = NAME(2)
123.      C
124.      C         TEST ROW TYPE
125.      C
126.            IF(K2.EQ.QL .OR. K3.EQ.QL)  GO TO 220
127.            IF(K2.EQ.QE .OR. K3.EQ.QE)  GO TO 230
128.            IF(K2.EQ.QG .OR. K3.EQ.QG)  GO TO 240
129.            IF(K2.EQ.QN .OR. K3.EQ.QN)  GO TO 250
130.            GO TO 230
131.          220 XLB(NROW) = 0.
132.            XUB(NROW) = 1.E4
133.            GO TO 250
134.          230 XLB(NROW) = 0.
135.            XUB(NROW) = 0.
136.            GO TO 250
137.          240 XLB(NROW) = 0.
138.            XUB(NROW) = 1.E4
139.            A(NROW) = -1.
140.            GO TO 260
141.          250 A(NROW) = 1.
142.          260 IA(NROW) = NROW
143.            LA(NROW) = NROW
144.            JH(NROW) = NROW
145.            KINBAS(NROW) = NROW
146.            NELEM=NROW
147.            GO TO 5
148.      C
149.      C         MATRIX ELEMENTS
150.      C
151.          320 J = 3
152.            K = 4
153.            IF (DABS(ATEMP1) .LE. ZTOLZE)  GO TO 321
154.            GO TO 324
155.          321 J=5
156.            K=6
157.            ATEMP1=ATEMP2
158.            GO TO 330
159.          324 CONTINUE
160.            IF (NAME(1) .EQ. ICS1 .AND. NAME(2) .EQ. ICS2)  GO TO 330
161.      C
162.      C         TEST FOR SPLIT VECTOR
163.      C
164.            DO 325 I = 1,NCOL
165.            IF (NAME(1) .NE. ICNAM(I,1))  GO TO 325
166.            IF (NAME(2) .NE. ICNAM(I,2))  GO TO 325
167.            WRITE(6,8250) NAME(1),NAME(2)
168.         8250 FORMAT(14HOSPLIT VECTOR ,2A4)
169.          325 CONTINUE
170.            NCOL = NCOL + 1
171.            ICS1 = NAME(1)
172.            ICS2 = NAME(2)
173.            ICNAM(NCOL,1) = ICS1
174.            ICNAM(NCOL,2) = ICS2
175.            LA(NCOL) = NELEM + 1
176.      C
177.      C         TEST FOR ROW MATCH
178.      C
179.          330 DO 340 I = 1,NROW
180.            IF(NAME(J) .NE. ICNAM(I,1) .OR. NAME(K) .NE. ICNAM(I,2))GO TO 340
181.            NELEM = NELEM + 1
182.            IA(NELEM) = I
```

```
183.            A(NELEM) = ATEMP1
184.            LA(NCOL+1)=NELEM+1
185.          335 IF(K .GT. 5)  GO TO 5
186.            IF(DABS(ATEMP2) .LE. ZTOLZE)  GO TO 5
187.            J = 5
188.            K = 6
189.            ATEMP1 = ATEMP2
190.            GO TO 330
191.          340 CONTINUE
192.            WRITE(6,8300) NAME(J),NAME(K),NAME(1),NAME(2)
193.         8300 FORMAT(18HONO MATCH FOR ROW ,2A4,10HAT COLUMN ,2A4)
194.            STOP
195.      C
196.      C        BASIS CARDS
197.      C
198.          410 DO 420 I = 1,NCOL
199.            IF(NAME(1) .NE. ICNAM(I,1) .OR. NAME(2) .NE. ICNAM(I,2)) GO TO 420
200.            IBVEC = I
201.            GO TO 425
202.          420 CONTINUE
203.            WRITE(6,8400) NAME(1),NAME(2)
204.         8400 FORMAT(21HONO MATCH FOR VECTOR ,2A4)
205.            GO TO 5
206.          425 DO 430 I = 1,NROW
207.            IF (NAME(3).NE.ICNAM(I,1).OR.NAME(4).NE.ICNAM(I,2))  GO TO 430
208.            IBROW=I
209.            GO TO 440
210.          430 CONTINUE
211.            WRITE(6,8300) NAME(3),NAME(4)
212.            GO TO 5
213.          440 JH(IBROW) = IBVEC
214.            KINBAS(IBROW) = 0
215.            KINBAS(IBVEC) = IBROW
216.            GO TO 5
217.      C
218.      C        RHS
219.      C
220.          500 J = 3
221.            K = 4
222.            IF (DABS(ATEMP1) .LE. ZTOLZE)  GO TO 521
223.            GO TO 524
224.          521 J=5
225.            K=6
226.            ATEMP1=ATEMP2
227.            GO TO 530
228.          524 CONTINUE
229.            IF (NAME(1) .EQ. ICS1 .AND. NAME(2) .EQ. ICS2)  GO TO 530
230.      C
231.      C        TEST FOR SPLIT VECTOR
232.      C
233.            DO 525 I = 1,NCOL
234.            IF (NAME(1) .NE. ICNAM(I,1))  GO TO 525
235.            IF (NAME(2) .NE. ICNAM(I,2))  GO TO 525
236.            WRITE(6,8250) NAME(1),NAME(2)
237.          525 CONTINUE
238.            ICS1 = NAME(1)
239.            ICS2 = NAME(2)
240.      C
241.      C        TEST FOR ROW MATCH
242.      C
243.          530 DO 540 I = 1,NROW
```

```
244.            IF(NAME(J) .NE. ICNAM(I,1) .OR. NAME(K) .NE. ICNAM(I,2))GO TO 540
245.            B(I) = ATEMP1
246.          535 IF(K .GT. 5)  GO TO 5
247.            IF(DABS(ATEMP2) .LE. ZTOLZE)  GO TO 5
248.            J = 5
249.            K = 6
250.            ATEMP1 = ATEMP2
251.            GO TO 530
252.          540 CONTINUE
253.            WRITE(6,8300) NAME(J),NAME(K)
254.            STOP
255.      C
256.      C             END OF INPUT
257.      C
258.          600 NSCOL = NCOL - NROW
259.            K = NROW + 1
260.      C        INPUT LOWER AND UPPER BOUNDS ON DECISION VARIABLES
261.            READ (5,650) (XLB(J), J=K,NCOL)
262.            READ (5,650) (XUB(J), J=K,NCOL)
263.          650 FORMAT (15F5.0)
264.            NELEM = NELEM - NROW
265.            RELEM = NELEM
266.            RDENS = RELEM/(NROW*NSCOL)
267.            RETURN
268.          9999 IFPROB = 0
269.            RETURN
270.            END
271.            SUBROUTINE FTRAN(IPAR)
272.      C
273.      C        PERFORMS FORWARD TRANSFORMATION ON COLUMN STORED IN VECTOR Y
274.      C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
275.      C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
276.      C********DESCRIPTION OF PARAMETERS********
277.      C        IPAR = PARAMETER INDICATING WHICH ETA-VECTORS MATRIX COLUMN
278.      C               IS TO BE UPDATED BY (INPUT)
279.      C*****************************************
280.      C
281.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
282.           1         INTEGER*4 (I-N,Q)
283.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
284.            DOUBLE PRECISION E(4000)
285.            REAL A(2000)
286.      C
287.            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
288.           1              QED,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
289.           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
290.      C
291.            COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
292.           1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
293.           2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFREZ,JCOLP,NROW,NCOL,NELEM,
294.           3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
295.           4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
296.      C
297.            GO TO (100,110), IPAR
298.          100 NFE = 1
299.            NLE = NETA
300.            GO TO 200
301.          110 NFE = NLETA + 1
302.            NLE = NETA
303.          200 IF (NFE .GT. NLE) GO TO 9000
304.            DO 1000 IK = NFE,NLE
```

```
      LL = LE(IK)
      KK = LE(IK+1) - 1
      IPIV = IE(LL)
      DY = Y(IPIV)
      DY = DY/E(LL)
      Y(IPIV) = DY
      IF (KK .LE. LL) GO TO 1000
      LL = LL + 1
      DO 500 J = LL,KK
      IR = IE(J)
      Y(IR) = Y(IR) - E(J) * DY
  500 CONTINUE
 1000 CONTINUE
 9000 CONTINUE
      RETURN
      END
      SUBROUTINE BTRAN
C
C        PERFORMS BACKWARD TRANSFORMATION ON COLUMN STORED IN VECTOR Y
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(4000)
      REAL A(2000)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QD,QR,QM,QG,NTMAX
C
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFEEZ,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
C
      IF (NETA .LE. 0) GO TO 9000
      DO 1000 I = 1,NETA
      IK = NETA - I + 1
      LL = LE(IK)
      KK = LE(IK+1) - 1
      IPIV = IE(LL)
      DP = E(LL)
      DY = Y(IPIV)
      DSUM = 0.
      IF (KK .LE. LL) GO TO 600
      LL = LL + 1
      DO 500 J = LL,KK
      IR = IE(J)
      DE = E(J)
      DPROD = DE * Y(IR)
      DSUM = DSUM + DPROD
  500 CONTINUE
C
  600 Y(IPIV) = (DY - DSUM) / DP
 1000 CONTINUE
 9000 RETURN
      END
      SUBROUTINE FORMC
```

```
      C
      C        FORMS OBJ. FUNCTION VECTOR; USED IN SUBROUTINE PRICE
      C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            DOUBLE PRECISION E(4000)
            REAL A(2000)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QD,QR,QM,QG,NTMAX
      C
            COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
           1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
           2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFFEZ,JCOLP,NROW,NCOL,NELEM,
           3       NETA,NLELEM,NLETA,NJELEM,NUETA,JH(60),KINBAS(122),LA(122),
           4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
      C
            MSTAT = QI
            IFFEZ = 0
            SUMINF = 0.
            Y(IOBJ) = 0.
            DO 30 I=1,NROW
            IF (I .EQ. IOBJ) GO TO 30
            ICOL = JH(I)
            IF (X(I) .LE. (XLB(ICOL) - ZTOLZE)) GO TO 10
            IF (X(I) .GE. (XUB(ICOL) + ZTOLZE)) GO TO 20
            Y(I) = 0.
            GO TO 30
         10 Y(I) = 1.
            SUMINF = SUMINF + XLB(ICOL) - X(I)
            GO TO 30
         20 Y(I) = -1.
            SUMINF = SUMINF + X(I) - XUB(ICOL)
         30 CONTINUE
            IF (SUMINF .GT. ZTOLSM) RETURN
            Y(IOBJ) = 1.
            IFFEZ = 1
            MSTAT = QF
            RETURN
            END
            SUBROUTINE PRICE
      C
      C        PRICES OUT NONBASIC COLUMNS; CHOOSES PIVOT COLUMN JCOLP FOR
      C        CURRENT PRIMAL SIMPLEX ITERATION
      C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            INTEGER*2 IPART,INCUMB,IVBND,IVID,IOBND
            DOUBLE PRECISION E(4000)
            REAL A(2000)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QD,QR,QM,QG,NTMAX
```

```
      C
            COMMON/BLOCK2/ DPPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
           1               IPART(122),INCOMB(122),IVBND(500),IVID(500),
           2               ICBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
      C
            COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
           1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
           2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFREZ,JCOLP,NROW,NCOL,NELEM,
           3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
           4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
      C
            CMIN = 1.E10
            CMAX = -1.E10
            DO 1000 J=1,NCOL
            IF (KINBAS(J) .GT. 0) GO TO 1000
            IF ((XUB(J) - XLB(J)) .LT. ZTOLZE) GO TO 1000
            DSUM = 0.
            LL = LA(J)
            KK = LA(J+1) - 1
            DO 500 I = LL,KK
            IR = IA(I)
            DE = A(I)
            DPROD = DE * Y(IR)
            DSUM = DSUM + DPROD
        500 CONTINUE
            IF (KINBAS(J) .EQ. -1) GO TO 600
            IF (DSUM .GE. CMIN) GO TO 1000
            CMIN = DSUM
            JCOL1 = J
            GO TO 1000
        600 IF (DSUM .LE. CMAX) GO TO 1000
            CMAX = DSUM
            JCOL2 = J
       1000 CONTINUE
      C
            IF (CMIN .LE. -ZTCOST) GO TO 1500
            IF (CMAX .GE. ZTCOST) GO TO 2000
            JCOLP = 0
            RCOST = 0.
            RETURN
       1500 IF (CMAX .GE. ZTCOST) GO TO 2500
       1600 JCOLP = JCOL1
            RCOST = CMIN
            RETURN
       2000 JCOLP = JCOL2
            RCOST = CMAX
            RETURN
       2500 IF (ABS(CMIN) - CMAX) 2000,2000,1600
            END
            SUBROUTINE CHUZR
      C
      C        PERFORMS MIN-RATIO TEST FOR PIVOT COLUMN JCOLP DETERMINED IN
      C        SUBROUTINE PRICE.  SELECTS PIVOT ROW IROWP FOR CURRENT PRIMAL
      C        SIMPLEX ITERATION.
      C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1         INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
```

```
      DOUBLE PRECISION E(4000)
      REAL A(2000)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRC,QMA,QBA,QFI,
     1              QED,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QD,QR,QM,QG,NTMAX
C
      COMMON/BLOCK2/ DEPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               IPART(122),INCUMB(122),IVBND(500),IVID(500),
     2               IOBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
C
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVERQ,ITREQ,IFFEZ,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
C
      IF (KINBAS(JCOLP) .EQ. -1) GO TO 1000
C
C                INCOMING VARIABLE AT LOWER BOUND
C
      DP = 1.E10
      DO 500 I=1,NROW
      IF (I .EQ. IOBJ) GO TO 500
      ICOL = JH(I)
      IF (Y(I) .GT. ZTOLPV) GO TO 100
      IF (Y(I) .LT. -ZTOLPV) GO TO 200
      GO TO 500
C                POSITIVE COEFFICIENT
  100 IF (X(I) .LT. (XLB(ICOL) - ZTOLZE)) GO TO 500
      DE = (X(I) - XLB(ICOL))/Y(I)
      IF (DE .GE. DP) GO TO 500
      IPTYPE = 0
      GO TO 250
C                NEGATIVE COEFFICIENT
  200 IF (X(I) .GT. (XUB(ICOL) + ZTOLZE)) GO TO 500
      DE = (X(I) - XUB(ICOL))/Y(I)
      IF (DE .GE. DP) GO TO 500
      IPTYPE = -1
  250 DP = DE
      IROWP = I
  500 CONTINUE
      DE = DP + XLB(JCOLP)
      IF (DE .LT. XUB(JCOLP)) GO TO 600
      DP = XUB(JCOLP) - XLB(JCOLP)
      NPIVOT = 0
      RETURN
  600 NPIVOT = 1
      RETURN
C
C                INCOMING VARIABLE AT UPPER BOUND
C
 1000 DP = -1.E10
      DO 1500 I=1,NROW
      IF (I .EQ. IOBJ) GO TO 1500
      ICOL = JH(I)
      IF (Y(I) .GT. ZTOLPV) GO TO 1100
      IF (Y(I) .LT. -ZTOLPV) GO TO 1200
      GO TO 1500
C                POSITIVE COEFFICIENT
 1100 IF (X(I) .GT. (XUB(ICOL) + ZTOLZE)) GO TO 1500
```

```
            DE = (X(I) - XUB(ICOL))/Y(I)
            IF (DE .LE. DP) GO TO 1500
            IPTYPE = -1
            GO TO 1250
C                 NEGATIVE COEFFICIENT
 1200 IF (X(I) .LT. (XLB(ICOL) - ZTOLZE)) GO TO 1500
            DE = (X(I) - XLB(ICOL))/Y(I)
            IF (DE .LE. DP) GO TO 1500
            IPTYPE = 0
 1250 DP = DE
            IROWP = I
 1500 CONTINUE
            DE = DP + XUB(JCOLP)
            IF (DE .GT. XLB(JCOLP)) GO TO 1600
            DP = XLB(JCOLP) - XUB(JCOLP)
            NPIVOT = 0
            RETURN
 1600 NPIVOT = 1
            RETURN
            END
            SUBROUTINE WRETA
C
C           FORMS NEW ETA-VECTORS FOR PRODUCT FORM OF BASIS INVERSE
C           SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C           BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            DOUBLE PRECISION E(4000)
            REAL A(2000)
C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
C
            COMMON DSUM,DPROD,DY,JE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
           1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
           2       IVIN,IVOUT,ITCNT,INVFRQ,ITPERQ,IFFEZ,JCOLP,NROW,NCOL,NELEM,
           3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
           4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
C
            NELEM = NELEM + 1
            IE(NELEM) = IROWP
            E(NELEM) = Y(IROWP)
C
            DO 1000 I = 1,NROW
            IF (I .EQ. IROWP) GO TO 1000
            IF (DABS(Y(I)) .LE. ZTOLZE) GO TO 1000
            NELEM = NELEM + 1
            IE(NELEM) = I
            E(NELEM) = Y(I)
 1000 CONTINUE
C
            NETA = NETA + 1
            LE(NETA+1) = NELEM + 1
            RETURN
            END
            SUBROUTINE SHIFTR(IOLD,INEW)
C
C           REARRANGES DATA STORAGE
```

```
      C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C********DESCRIPTION OF PARAMETERS********
      C         IOLD,INEW = PARAMETERS INDEXING STORAGE LOCATIONS IN WHICH
      C                     DATA IS TO BE TRANSFERRED (INPUT)
      C*****************************************
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            DOUBLE PRECISION E(4000)
            REAL A(2000)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      C
            COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
           1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
           2       IVIN,IVOUT,ITCNT,INVERQ,ITREQ,IFEZ,JCOLP,NROW,NCOL,NELEM,
           3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
           4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
      C
            DIMENSION BARRAY(240)
            EQUIVALENCE (BARRAY(1),B(1))
            IFO = (IOLD - 1) * NRMAX
            IFN = (INEW - 1) * NRMAX
      C
            DO 1000 I = 1,NROW
            BARRAY(IFN + I) = BARRAY(IFO + I)
       1000 CONTINUE
            RETURN
            END
            SUBROUTINE INVERT
      C
      C        COMPUTES INVERSE OF CURRENT BASIS BY LU DECOMPOSITION
      C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            DOUBLE PRECISION E(4000)
            REAL A(2000)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      C
            COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
           1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
           2       IVIN,IVOUT,ITCNT,INVERQ,ITREQ,IFEZ,JCOLP,NROW,NCOL,NELEM,
           3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
           4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
      C
            INTEGER*2 MREG,HREG,VREG
            DIMENSION MREG(60),HREG(60),VREG(60)
            EQUIVALENCE (MREG(1),YTEMP(1)),(HREG(1),YTEMP(31)),(VREG(1),X(1))
      C
      C           SET PARAMETERS
      C
```

```
671.            NETA = 0
672.            NLETA = 0
673.            NUETA = 0
674.            NELEM = 0
675.            NLELEM = 0
676.            NUELEM = 0
677.            NABOVE = 0
678.            LE(1) = 1
679.            LR1 = 1
680.            KR1 = 0
681.            LR4 = NROW + 1
682.            KR4 = NROW
683.      C
684.      C         PUT SLACKS AND ARTIFICIALS IN PART 4 AND REST IN PART 1
685.      C
686.            DO 100 I = 1,NROW
687.            IF (JH(I) .GT. NROW) GO TO 50
688.            LR4 = LR4 - 1
689.            MREG(LR4) = JH(I)
690.            VREG(LR4) = JH(I)
691.            GO TO 90
692.         50 KR1 = KR1 + 1
693.            VREG(KR1) = JH(I)
694.         90 HREG(I) = -1
695.            JH(I) = 0
696.        100 CONTINUE
697.      C
698.            KR3 = LR4 -1
699.            LR3 = LR4
700.      C
701.            DO 200 I = LR4,KR4
702.            IR = MREG(I)
703.            HREG(IR)= 0
704.            JH(IR) = IR
705.            KINBAS(IR) = IR
706.        200 CONTINUE
707.      C
708.      C              PULL OUT VECTORS BELOW BUMP AND GET ROW COUNTS
709.      C
710.            NBNONZ = KR4 - LR4 + 1
711.            IF (KR1 .EQ. 0) GO TO 1190
712.            J = LR1
713.        210 IV = VREG(J)
714.            LL = LA(IV)
715.            KK = LA(IV+1) -1
716.            IRCNT = 0
717.            DO 220 I = LL,KK
718.            NBNONZ = NBNONZ + 1
719.            IR = IA(I)
720.            IF (HREG(IR) .GE. 0) GO TO 220
721.            IRCNT = IRCNT + 1
722.            HREG(IR) = HREG(IR) - 1
723.            IRP = IR
724.        220 CONTINUE
725.            IF (IRCNT - 1) 230,250,300
726.        230 WRITE(6,8000)
727.       8000 FORMAT(18HOMATRIX SINGULAR )
728.            KINBAS(IV) = 0
729.            VREG(J) = VREG(KR1)
730.            KR1 = KR1 - 1
731.            IF (J .GT. KR1) GO TO 310
```

```
732.            GO TO 213
733.      C
734.        250 VREG(J) = VREG(KR1)
735.            KR1 = KR1 - 1
736.            LR3 = LR3 - 1
737.            VREG(LR3) = IV
738.            MREG(LR3) = IRP
739.            HREG(IRP) = 0
740.            JH(IRP) = IV
741.            KINBAS(IV) = IRP
742.            IF (J .GT. KR1) GO TO 310
743.            GO TO 210
744.        300 IF (J .GE. KR1) GO TO 310
745.            J = J+1
746.            GO TO 210
747.      C
748.      C                     PULL OUT REMAINING VECTORS ABOVE AND BELOW THE
749.      C                     BUMP AND ESTABLISH MERIT COUNTS OF COLUMNS
750.      C
751.        310 NVREM = 0
752.            IF(KR1 .EQ. 0) GO TO 1190
753.            J = LR1
754.        320 IV = VREG(J)
755.            LL = LA(IV)
756.            KK = LA(IV+1) - 1
757.            IRCNT = 0
758.            DO 800 I = LL,KK
759.            IR = IA(I)
760.            IF(HREG(IR) .NE. -2) GO TO 400
761.      C
762.      C             PIVOT ABOVE BUMP (PART OF L)
763.      C
764.            NABOVE = NABOVE + 1
765.            IROWP = IR
766.            CALL UNPACK(IV)
767.            CALL WPETA
768.            NLETA = NETA
769.            JH(IR) = IV
770.            KINBAS(IV) = IR
771.            VREG(J) = VREG(KR1)
772.            KR1 = KR1 - 1
773.            NVREM = NVREM + 1
774.            HREG(IR) = IV
775.            GO TO 540
776.      C
777.        400 IF (HREG(IR) .GE. 0) GO TO 800
778.            IRCNT = IRCNT + 1
779.            IRP = IR
780.        800 CONTINUE
781.      C
782.            IF (IRCNT - 1) 810,900,1000
783.        810 WRITE(6,8000)
784.            KINBAS(IV) = 0
785.            VREG(J) = VREG(KR1)
786.            NVREM = NVREM + 1
787.            KR1 = KR1 - 1
788.            IF (J .GT. KR1) GO TO 1010
789.            GO TO 320
790.      C
791.      C                   PUT VECTOR BELOW BUMP
792.      C
```

```
  900 VREG(J) = VREG(KR1)
      NVREM = NVREM + 1
      KR1 = KR1 - 1
      LR3 = LR3 - 1
      VREG(LR3) = IV
      MREG(LR3) = IRP
      MREG(IRP) = 0
      JH(IRP) = IV
      KINPAS(IV) = IRP
C
C           CHANGE ROW COUNTS
C
  940 DO 950 II = LL,KK
      IIR = IA(II)
      IF (MREG(IIR) .GE. 0) GO TO 950
      MREG(IIR) = MREG(IIR) + 1
  950 CONTINUE
      IF (J .GT. KR1) GO TO 1010
      GO TO 320
 1000 IF (J .GE. KR1) GO TO 1010
      J = J+1
      GO TO 320
 1010 IF(NVREM .GT. 0) GO TO 310
C
C               GET MERIT COUNTS
C
 1020 IF (KR1 .EQ. 0) GO TO 1190
      DO 1100 J = LR1,KR1
      IV = VREG(J)
      LL = LA(IV)
      KK = LA(IV+1) - 1
      IMCNT = 0
      DO 1050 I = LL,KK
      IR = IA(I)
      IF (MREG(IR) .GE. 0) GO TO 1050
      IMCNT = IMCNT - (MREG(IR) +1)
 1050 CONTINUE
      MREG(J) = IMCNT
 1100 CONTINUE
C
C                   SORT COLUMNS INTO MERIT ORDER
C                   USING SHELL SORT
C
      ISD = 1
 1106 IF (KR1 .LT. 2*ISD) GO TO 1108
      ISD = 2*ISD
      GO TO 1106
 1108 ISD = ISD - 1
C                         END OF INITIALIZATION
 1101 IF (ISD .LE. 0) GO TO 1107
      ISK = 1
 1102 ISJ = ISK
      ISL = ISK + ISD
      ISY = MREG(ISL)
      ISZ = VREG(ISL)
 1103 IF (ISY .LT. MREG(ISJ)) GO TO 1104
 1105 ISL = ISJ + ISD
      MREG(ISL) = ISY
      VREG(ISL) = ISZ
      ISK = ISK + 1
      IF ((ISK + ISD) .LE. KR1) GO TO 1102
```

```
            ISD = (ISD - 1) / 2
            GO TO 1101
 1104 ISL = ISJ + ISD
            MREG(ISL) = MREG(ISJ)
            VREG(ISL) = VREG(ISJ)
            ISJ = ISJ - ISD
            IF (ISJ .GT. 0) GO TO 1103
            GO TO 1105
 1107 CONTINUE
C                              END OF SORT ROUTINE
C
C           PUT OUT BELOW BUMP ETAS (PART OF U)
C
 1190 NSLCK = 0
            NBELOW = 0
            NELAST = NEMAX
            NTLAST = NTMAX
            LE(NTLAST + 1) = NELAST + 1
C
            LR = LR3
            IF (LR3 .GE. LR4) LR = LR4
            IF (LR .GT. KR4) GO TO 2050
            JK = KR4 + 1
            DO 2000 JJ= LR,KR4
            JK = JK - 1
            IV = VREG(JK)
            I = MREG(JK)
            NBELOW = NBELOW + 1
            IF (IV .GT. NROW) GO TO 1200
            NSLCK = NSLCK + 1
 1200 LL = LA(IV)
            KK = LA(IV+1) -1
            IF (KK .GT. LL) GO TO 1300
 1250 IF (ABS(A(LL) - 1.) .LE. ZTOLZE) GO TO 2000
C
 1300 NUETA = NUETA + 1
            DO 1400 J = LL,KK
            IR = IA(J)
            IF (IR .EQ. I)GO TO 1390
            IE(NELAST) = IR
            E(NELAST) = A(J)
            NELAST = NELAST - 1
            NUELEM = NUELEM + 1
            GO TO 1400
 1390 EP = A(J)
 1400 CONTINUE
            IE(NELAST) = I
            E(NELAST) = EP
            LE(NTLAST) = NELAST
            NELAST = NELAST - 1
            NTLAST = NTLAST - 1
            NUELEM = NUELEM + 1
 2000 CONTINUE
 2050 IF(KR1 .EQ. 0) GO TO 3500
C
C           DO L-U DECOMPOSITION OF BUMP
C
            DO 3000 J = LR1,KR1
            IV = VREG(J)
            CALL UNPACK(IV)
            CALL FTRAN(2)
```

```
915.            IROWP = 0
916.            IRCMIN = -999999
917.            DO 2100 I = 1,NROW
918.            IF (DABS(Y(I)) .LE. ZTOLPV) GO TO 2100
919.            IF (HREG(I) .GE.0) GO TO 2100
920.            IF (HREG(I) .LE. IRCMIN) GO TO 2100
921.            IRCMIN = HREG(I)
922.            IROWP = I
923.       2100 CONTINUE
924.            IF (IROWP .GT. 0) GO TO 2150
925.            WRITE(6,8000)
926.            KINBAS(IV) = 0
927.            GO TO 3500
928.      C
929.       2150 INCR = HREG(IROWP) + 3
930.      C
931.      C               WRITE L AND U ETAS
932.      C
933.            IF (J .EQ. KR1) GO TO 2160
934.            NELEM = NELEM + 1
935.            IE(NELEM) = IROWP
936.            E(NELEM) = Y(IROWP)
937.       2160 DO 2300 I = 1,NROW
938.            IF (I .EQ. IROWP) GO TO 2300
939.            IF(DABS(Y(I)) .LE. ZTOLZE) GO TO 2300
940.            IF (HREG(I) .GE. 0) GO TO 2200
941.      C
942.      C                  L ETA ELEMENTS
943.      C
944.            NELEM = NELEM + 1
945.            IE(NELEM) = I
946.            E(NELEM) = Y(I)
947.            GO TO 2300
948.      C
949.      C                  U ETA ELEMENTS
950.      C
951.       2200 IE(NELAST) = I
952.            E(NELAST) = Y(I)
953.            NELAST = NELAST - 1
954.            NUELEM = NUELEM + 1
955.       2300 CONTINUE
956.      C
957.            JH(IROWP) = IV
958.            KINBAS(IV) = IROWP
959.            NUETA = NUETA + 1
960.            IE(NELAST) = IROWP
961.            IF (J .NE. KR1) GO TO 2330
962.            E(NELAST) = Y(IROWP)
963.            GO TO 2340
964.       2330 E(NELAST) = 1.
965.            NETA = NETA + 1
966.            LE(NETA+1) = NELEM + 1
967.       2340 NUELEM = NUELEM + 1
968.            LE(NTLAST) = NELAST
969.            NELAST = NELAST - 1
970.            NTLAST = NTLAST - 1
971.      C
972.      C               UPDATE ROW COUNTS
973.      C
974.            DO 2350 I = 1,NROW
975.            IF (DABS(Y(I)) .LE. ZTOLZE) GO TO 2350
```

```
      IF (HREG(I) .GE. 0) GO TO 2350
      HREG(I) = HREG(I) - INCR
      IF (HREG(I) .GE. 0) HREG(I) = -1
 2350 CONTINUE
      HREG(IROWP) = 0
 3000 CONTINUE
C
C               MERGE L AND U ETAS
C
 3500 NLETA = NETA
      NETA = NLETA + NUETA
      NLELEM = NELEM
      NELEM = NLELEM + NUELEM
      IF (NUELEM .EQ. 0) GO TO 3550
      CALL SHFTE
C
C                   INSERT SLACKS FOR DELETED COLUMNS
C
 3550 DO 3600 I = 1,NROW
      IF (JF(I) .NE. 0) GO TO 3600
      JH(I) = 1
      IROWP = I
      CALL UNPACK(I)
      CALL FTRAN(1)
      CALL WRETA
 3600 CONTINUE
C
C                   UPDATE X
C
      CALL SHIFTR(1,3)
      DO 9000 J=1,NCOL
      IF (KINBAS(J)) 8600,8700,9000
 8600 DE = XUB(J)
      GO TO 8750
 8700 DE = XLB(J)
 8750 LL = LA(J)
      KK = LA(J+1) - 1
      DO 8800 I=LL,KK
      IR = IA(I)
 8800 Y(IR) = Y(IR) - A(I)*DE
 9000 CONTINUE
      CALL FTRAN(1)
      CALL SHIFTR(3,2)
C
C                   PRINT STATISTICS
C
      NOFD = NELEM - NETA
      NSTR = NROW - NSLCK
C
      RETURN
      END
      SUBROUTINE UNPACK(IV)
C
C        EXPANDS COMPRESSED MATRIX COLUMNS
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C********DESCRIPTION OF PARAMETERS********
C        IV = PARAMETER INDEXING COLUMN TO BE EXPANDED (INPUT)
C*************************************
C
      IMPLICIT REAL*4 (A,C,E-H,J,P,R-W,Z), REAL*8 (B,D,X,Y),
```

```
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(4000)
      REAL A(2000)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,ORG,OMA,OBA,OFI,
     1              OEQ,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
     2              OE,OH,OL,OO,OR,OM,OG,NTMAX
C
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVERQ,ITREPQ,IFREEZ,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
C
      DO 100 I = 1,NROW
      Y(I) = 0.
  100 CONTINUE
C
      LL = LA(IV)
      KK = LA(IV+1) - 1
      DO 200 I = LL,KK
      IR = IA(I)
      Y(IR) = A(I)
  200 CONTINUE
C
      RETURN
      END
      SUBROUTINE SHFTE
C
C        SUBROUTINE FOR INVERT
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM=1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(4000)
      REAL A(2000)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,ORG,OMA,OBA,OFI,
     1              OEQ,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
     2              OE,OH,OL,OO,OR,OM,OG,NTMAX
C
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVERQ,ITREPQ,IFREEZ,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
C
C           SHIFT IE AND E OF U ELEMENTS
C
      NF = NEMAX - NUELEM + 1
      INCR = 0
      DO 1000 I = NF,NEMAX
      INCR = INCR + 1
      IE(NLELEM + INCR) = IE(I)
      E(NLELEM + INCR) = E(I)
 1000 CONTINUE
C
      IDIF = NEMAX - NLELEM - NUELEM
```

```
      NF = NTMAX - NBETA + 1
      INCR = 0
      DO 2000 I = NF,NTMAX
      INCR = INCR + 1
      LE(NBETA + INCR) = LE(I) - IDIF
 2000 CONTINUE
      LE(NBETA+1) = NELEM + 1
      RETURN
      END
      SUBROUTINE UPBETA
C
C        UPDATES RIGHT-HAND SIDES TO REFLECT NEW BASIS RESULTING FROM
C        CURRENT SIMPLEX PIVOT
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(4000)
      REAL A(2000)
C
      COMMON/BLOCK1/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRC,QMA,QBA,QFI,
     1               QEC,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2               QE,QP,QL,QO,QR,QM,QG,NTMAX
C
      COMMON/BLOCK2/ DEPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               IPART(122),INCOMB(122),IVBND(500),IVID(500),
     2               ICBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
C
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFREZ,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
C
      DO 1000 I=1,NROW
 1000 X(I) = X(I) - Y(I)*DP
      IF (NPIVOT .EQ. 1) GO TO 2000
      KINBAS(JCOLP) = -(KINBAS(JCOLP) + 1)
      IVOUT = JCOLP
      RETURN
 2000 X(IROWP) = DE
      IVOUT = JH(IROWP)
      KINBAS(JCOLP) = IROWP
      KINBAS(IVOUT) = IPTYPE
      JH(IROWP) = JCOLP
      RETURN
      END
      SUBROUTINE NORMAL(ITSINV)
C
C        SERVES AS MASTER PROGRAM FOR LINEAR PROGRAMMING COMPONENT
C        (REVISED, PRIMAL-SIMPLEX METHOD) OF BRANCH-AND-BOUND ROUTINE BB
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C********DESCRIPTION OF PARAMETERS********
C        ITSINV = NUMBER OF SIMPLEX ITERATIONS SINCE LAST BASIS
C                 INVERSION (INPUT/OUTPUT)
C*************************************************
C
```

```
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(4000)
      REAL A(2000)
C
      COMMON/BLOCK1/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRD,QMA,QBA,QFI,
     1               QEO,QBL,QA,QPL,QMI,QZ,QT,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2               QE,QH,QL,QD,QR,QM,QG,NTMAX
C
      COMMON/BLOCK2/ DEPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               IPART(122),INCUMB(122),IVBND(500),IVID(500),
     2               ICBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
C
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFEAS,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NJELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
C
      IF (ITSINV .LT. INVFRQ) GO TO 1500
 1000 CALL INVERT
      ITSINV = 0
C
C               SIMPLEX CYCLE
C
 1500 CALL FORMC
      CALL PTRAN
      CALL PRICE
      IF (JCOLP .GT. 0) GO TO 3000
      IF (MSTAT .EQ. 01    ) GO TO 2000
      MSTAT = QBL
      GO TO 6000
 2000 MSTAT = QN
      GO TO 6000
 3000 IVIN = JCOLP
      CALL UNPACK(JCOLP)
      CALL FTRAN(1)
      CALL CHUZR
      CALL UPBETA
      ITCNT = ITCNT + 1
      ITSINV = ITSINV + 1
      IF (NPIVOT .EQ. 0) GO TO 4010
      IF (NELEM .GT. 5600) GO TO 1000
      CALL WRETA
 4010 IF (ITSINV .GE. INVFRQ) GO TO 1000
      IF (ITCNT .GE. ITRFRQ) GO TO 6000
      GO TO 1500
C
 6000 RETURN
      END
      SUBROUTINE BANDB(INITBD,ITSINV)
C
C        MASTER PROGRAM FOR BRANCH-AND-BOUND INTEGER PROGRAMMING
C        ROUTINE.  ALSO SERVES AS MASTER PROGRAM FOR REOPTIMIZATION
C        VIA REVISED DUAL-SIMPLEX METHOD AFTER A FORWARD BRANCH.
C********DESCRIPTION OF PARAMETERS********
C        INITBD = INITIAL LOWER BOUND ON MAXIMAL OBJECTIVE VALUE (INPUT)
C        ITSINV = NUMBER OF SIMPLEX ITERATIONS SINCE LAST BASIS
C                 INVERSION (INPUT/OUTPUT)
```

```
      C*************************************************
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
            DOUBLE PRECISION E(4000)
            REAL A(2000)
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
            COMMON/BLOCK2/ DEPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
           1               IPART(122),INCUMB(122),IVBND(500),IVID(500),
           2               ICBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
            COMMON DSUM,DPRCD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
           1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
           2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFEAZ,JCOLP,NROW,NCOL,NELEM,
           3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
           4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
            IF (ITCNT .GE. ITRFRQ) RETURN
            IFEAS = 0
            LISTL = 0
            INCVAL = INITBD
      C        TEST FOR FATHOMING
        100 CALL TESTX
            IF (MSTAT .EQ. QBL) GO TO 200
      C        CURRENT NODE FATHOMED; BACKTRACK TO LAST PROMISING NODE ON LIST
        150 CALL BKTRAK
      C        IF LIST IS EMPTY, RETURN TO MAIN(COMPUTATIONS COMPLETED)
            IF (LISTL .EQ. 0) RETURN
      C        USE PRIMAL SIMPLEX METHOD FOR REOPTIMIZATION AT NEW NODE
            CALL NORMAL(ITSINV)
            IF (ITCNT .GE. ITRFRQ) RETURN
            GO TO 100
      C        CURRENT NODE NOT FATHOMED; COMPUTE PENALTIES
      C        BRANCHING AT CURRENT NODE IS DONE FROM SUBROUTINE PENLTS
        200 CALL PENLTS
            IF (IDIR) 400,150,400
      C
      C        REINVERT CURRENT BASIS
       1000 CALL INVERT
            ITSINV = 0
      C
      C              DUAL SIMPLEX CYCLE
      C
      C        CHOOSE PIVOT ROW IROWP
        300 CALL DCHUZR
            IF (IROWP .GT. 0) GO TO 400
            MSTAT = QBL
            GO TO 100
      C        CHOOSE PIVOT COLUMN JCOLP
        400 CALL DCHUZC
            IF (JCOLP .EQ. 0) GO TO 150
            IVIN = JCOLP
      C        UPDATE RIGHT-HAND SIDES TO REFLECT NEW BASIS RESULTING FROM
      C        CURRENT SIMPLEX PIVOT
            CALL UPBETA
            ITCNT = ITCNT + 1
            IF (ITCNT .GE. ITRFRQ) RETURN
            ITSINV = ITSINV + 1
      C        DECIDE WHETHER TO REINVERT CURRENT BASIS
```

```
      IF ((NELEM .GT. 5000) .OR. (ITSINV .GE. INVFRQ)) GO TO 1000
C        REINVERSION NOT NECESSARY YET; WRITE OUT NEW ETA-VECTOR FOR
C        CURRENT SIMPLEX PIVOT
      CALL WRETA
      GO TO 300
      END
      SUBROUTINE DCHUZR
C
C        SELECTS PIVOT ROW IROWP FOR CURRENT DUAL-SIMPLEX ITERATION.
C        SETS IROWP=0 IF CURRENT BASIS IS OPTIMAL.  OTHERWISE, IROWP IS
C        CHOSEN TO BE THE ROW WITH GREATEST PRIMAL INFEASIBILITY.
C
      IMPLICIT REAL*4 (A,C-E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,IOBND
      DOUBLE PRECISION E(4000)
      REAL A(2000)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRG,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DEPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               IPART(122),INCUMB(122),IVBND(500),IVID(500),
     2               IOBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFEAS,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
      IROWP = 0
      DP = -1.E10
      DO 1000 I=1,NROW
      IF (I .EQ. IOBJ) GO TO 1000
      ICOL = JH(I)
      IF (X(I) .LT. (XLB(ICOL) - ZTOLZE)) GO TO 100
      IF (X(I) .GT. (XUB(ICOL) + ZTOLZE)) GO TO 200
      GO TO 1000
C
C        BASIC VARIABLE ON ROW I FALLS BELOW ITS LOWER BOUND
  100 DE = XLB(ICOL) - X(I)
      IF (DE .LE. DP) GO TO 1000
      IPTYPE = 0
      GO TO 250
C
C        BASIC VARIABLE ON ROW I EXCEEDS ITS UPPER BOUND
  200 DE = X(I) - XUB(ICOL)
      IF (DE .LE. DP) GO TO 1000
      IPTYPE = -1
C
  250 IROWP = I
 1000 CONTINUE
      RETURN
      END
      SUBROUTINE DCHUZC
C
C        SELECTS PIVOT COLUMN JCOLP FOR CURRENT DUAL-SIMPLEX ITERATION.
C        SETS JCOLP=0 IF LP-PROBLEM AT CURRENT NODE IS INFEASIBLE.
C        OTHERWISE CHOOSES JCOLP TO MAINTAIN PRIMAL-OPTIMALITY.
C
      IMPLICIT REAL*4 (A,C-E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
```

```
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(4000)
      REAL A(2000)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DEPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               IPART(122),INCOMB(122),IVBND(500),IVID(500),
     2               ICBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFREZ,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
C
      JCOLP = 0
      IF (IPTYPE .EQ. -1) GO TO 1000
C
C               LEAVING VARIABLE FALLS BELOW ITS LOWER BOUND
C
      DP = -1.E10
      DO 500 J=1,NCOL
      IF (KINBAS(J) .GT. 0) GO TO 500
      IF ((XUB(J) - XLB(J)) .LE. ZTOLZE) GO TO 500
      CALL UNPACK(J)
      CALL FTRAN(1)
      IF (KINBAS(J) .EQ. -1) GO TO 200
      IF (Y(IROWP) + ZTOLPV) 225,225,500
  200 IF (Y(IROWP) - ZTOLPV) 500,225,225
C
  225 DE = Y(IOBJ)/Y(IROWP)
      IF (DE - DP) 500,500,250
  250 JCOLP = J
      DP = DE
  500 CONTINUE
C
      IF (JCOLP .EQ. 0) RETURN
      CALL UNPACK(JCOLP)
      CALL FTRAN(1)
      ICOL = JH(IROWP)
      DP = (X(IROWP) - XLB(ICOL))/Y(IROWP)
      GO TO 2000
C
C               LEAVING VARIABLE EXCEEDS ITS UPPER BOUND
C
 1000 DP = 1.E10
      DO 1500 J=1,NCOL
      IF (KINBAS(J) .GT. 0) GO TO 1500
      IF ((XUB(J) - XLB(J)) .LE. ZTOLZE) GO TO 1500
      CALL UNPACK(J)
      CALL FTRAN(1)
      IF (KINBAS(J) .EQ. -1) GO TO 1200
      IF (Y(IROWP) - ZTOLPV) 1500,1225,1225
 1200 IF (Y(IROWP) + ZTOLPV) 1225,1225,1500
C
 1225 DE = Y(IOBJ)/Y(IROWP)
      IF (DE - DP) 1250,1500,1500
 1250 JCOLP = J
      DP = DE
 1500 CONTINUE
```

```
      C
            IF (JCOLP .EQ. 0) RETURN
            CALL UNPACK(JCOLP)
            CALL FTRAN(1)
            ICOL = JH(IROWP)
            DP = (X(IROWP) - XUB(ICOL))/Y(IROWP)
      C
       2000 IF (KINBAS(JCOLP) .EQ. 0) DE = DP + XLB(JCOLP)
            IF (KINBAS(JCOLP) .EQ. -1) DE = DP + XUB(JCOLP)
            NPIVOT = 1
            RETURN
            END
            SUBROUTINE TESTX
      C
      C        TESTS LP-OPTIMAL SOLUTION AT CURRENT NODE FOR FATHOMING.
      C        FATHOMING OCCURS IF
      C        (1) LP PROBLEM AT CURRENT NODE IS INFEASIBLE (MSTAT = ON); OR
      C        (2) LP-OPTIMAL OBJ. VALUE (IVAL) .LE. OBJ. VALUE OF CURRENT
      C            INCUMBENT SOLUTION(INCVAL); OR
      C        (3) LP-OPTIMAL SOLUTION SATISFIES INTEGER RESTRICTIONS.
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            INTEGER*2 IPART,INCUMB,IVBND,IVID,IOBND
            DOUBLE PRECISION E(4000)
            REAL A(2000)
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
            COMMON/BLOCK2/ DFPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
           1               IPART(122),INCUMB(122),IVBND(500),IVID(500),
           2               IOBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
            COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
           1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
           2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFREZ,JCOLP,NROW,NCOL,NELEM,
           3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
           4       LL(1002),IA(2000),IE(4000),XLB(122),XUB(122)
            COMMON/TIMERS/ TFIRST,TOPT,IFVAL,ITCT
            DP = X(IOBJ) + ZTOLZE
            IVAL = IDINT(DP)
            IF (DP .LT. 0.) IVAL = IVAL - 1
            IF ((MSTAT .EQ. ON) .OR. (IVAL .LE. INCVAL)) GO TO 2000
      C
      C              COMPUTE INTEGER AND FRACTIONAL PARTS OF EACH BASIC VAR.
      C
            DO 100 I=1,NROW
            IPART(I) = IDINT(X(I) + ZTOLZE)
            NTEMP(I) = IPART(I)
        100 DFPART(I) = X(I) - FLOAT(NTEMP(I))
      C
      C              CHECK FOR ALL-INTEGER SOLUTION
      C
            DO 200 I=1,NROW
            IF (JH(I) .LE. NROW) GO TO 200
            IF (DFPART(I) .GE. ZTOLZE) RETURN
        200 CONTINUE
      C
      C              SOLUTION ALL-INTEGER: INSTALL AS NEW INCUMBENT
      C
      C        COMPUTE AND OUTPUT COMPUTATION TIME REQUIRED TO REACH NEW
```

```
C         INCUMBENT SOLUTION.  ALSO OUTPUT OBJ. VAL. FOR THIS SOLUTION.
      CALL PCLOCK(JTIME,ITOT)
      TOPT = JTIME/100.
      WRITE (6,1) TOPT,IVAL
    1 FORMAT (' TIME =',F7.2,' SECONDS:  INCVAL =',I10)
      IF (IFEAS .EQ. 1) GO TO 500
      TFIRST = TOPT
      IFVAL = IVAL
      IFEAS = 1
  500 INCVAL = IVAL
      DO 1000 J=1,NCOL
      IF (KINBAS(J))600,700,1000
  600 INCUMB(J) = IDINT(XUB(J))
      GO TO 1000
  700 INCUMB(J) = IDINT(XLB(J))
 1000 CONTINUE
C
      DO 1100 I=1,NROW
      ICOL = JH(I)
      INCUMB(ICOL) = IPART(I)
 1100 CONTINUE
C
C              CURRENT PROBLEM NO LONGER OF INTEREST
C
 2000 MSTAT = QI
      RETURN
      END
      SUBROUTINE PENLTS
C
C         COMPUTES TOMLIN'S IMPROVED UP- AND DOWN- PENALTIES AND THE
C         GOMORY PENALTY FOR EACH NONINTEGER BASIC VARIABLE.  ALSO CHECKS
C         FOR FORCED BRANCHES ON BOTH BASIC AND NONBASIC VARIABLES.  IN
C         THE ABSENCE OF FORCED BRANCHES ON BASIC VARIABLES, THE
C         BRANCHING VARIABLE IS CHOSEN TO BE THE ONE WITH LARGEST
C         ASSOCIATED UP- OR DOWN-PENALTY.  THE FORWARD BRANCH IS TAKEN IN
C         THE DIRECTION OPPOSITE TO THE MAXIMUM PENALTY.  THE NODE
C         CORRESPONDING TO THE BRANCH IN THE SAME DIRECTION AS THE
C         MAXIMUM PENALTY IS ADDED TO THE LIST, TO BE EXAMINED LATER.
C         THE BRANCHING PROCESS ITSELF IS CARRIED OUT IN SUBROUTINE
C         BRANCH, WHICH IS CALLED FROM SUBROUTINE PENLTS.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(4000)
      REAL A(2000)
      REAL PU(60),PD(60),PG(60)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QM,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DFPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               IPART(122),INCUMB(122),IVBND(500),IVID(500),
     2               ICBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVERQ,ITRERQ,IFEAZ,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
      DO 10 I=1,NROW
      IF (DFPART(I) .LT. ZTOLZE) GO TO 5
```

```
1525.            PU(I) = 1.E6
1526.            PD(I) = 1.E6
1527.            PG(I) = 1.E6
1528.            GO TO 10
1529.          5 PU(I) = 0.
1530.            PD(I) = 0.
1531.            PG(I) = 0.
1532.         10 CONTINUE
1533.      C
1534.            DO 1000 J=1,NCOL
1535.            IF (KINBAS(J) .GT. 0) GO TO 1000
1536.            IF ((XUB(J) - XLB(J)) .LE. ZTOLZE) GO TO 1000
1537.            CALL UNPACK(J)
1538.            CALL FTRAN(1)
1539.            IF (KINBAS(J) .EQ. 0) GO TO 30
1540.            DO 20 I=1,NROW
1541.         20 Y(I) = -Y(I)
1542.      C
1543.      C                 CHECK FOR FORCED BRANCH ON XJ
1544.         30 IF (J .LE. NROW) GO TO 50
1545.            DP = X(IOBJ) - Y(IOBJ) + ZTOLZE
1546.            IVAL = IDINT(DP)
1547.            IF (DP .LT. 0.) IVAL = IVAL - 1
1548.            IF (IVAL .GT. INCVAL) GO TO 50
1549.            IDIR = 2*KINBAS(J) + 1
1550.            IF (IDIR .EQ. -1) REVBND = SNGL(XUB(J))
1551.            IF (IDIR .EQ. 1) REVBND = SNGL(XLB(J))
1552.            ICOL = J
1553.            CALL BRANCH
1554.            GO TO 1000
1555.      C
1556.         50 DO 500 I=1,NROW
1557.            IF (JH(I) .LE. NROW) GO TO 500
1558.            IF (DFPART(I) .LT. ZTOLZE) GO TO 500
1559.      C
1560.      C                 COMPUTE UP PENALTY FOR X(I), XJ
1561.        100 IF (Y(I) .GT. -ZTOLPV) GO TO 200
1562.            DE = Y(IOBJ)*(DFPART(I) - 1.)/Y(I)
1563.            IF (DE .LT. Y(IOBJ)) DE = Y(IOBJ)
1564.            IF (DE .LT. PU(I)) PU(I) = DE
1565.            GO TO 300
1566.      C
1567.      C                 COMPUTE DOWN PENALTY FOR X(I), XJ
1568.        200 IF (Y(I) .LT. ZTOLPV) GO TO 300
1569.            DE = Y(IOBJ)*DFPART(I)/Y(I)
1570.            IF (DE .LT. Y(IOBJ)) DE = Y(IOBJ)
1571.            IF (DE .LT. PD(I)) PD(I) = DE
1572.      C
1573.      C                 COMPUTE GOMORY PENALTY FOR X(I), XJ
1574.        300 DP = DABS(Y(I))
1575.            NTEMP(I) = IDINT(DP)
1576.            DP = DP - FLOAT(NTEMP(I))
1577.            IF ((DP .LE. ZTOLZE) .OR. (DP .GE. 1.-ZTOLZE)) GO TO 500
1578.            IF (Y(I) .LT. 0.) DP = 1. - DP
1579.            IF (DP .GT. DFPART(I)) GO TO 310
1580.            DE = Y(IOBJ)*DFPART(I)/DP
1581.            GO TO 320
1582.        310 DE = Y(IOBJ)*(1. - DFPART(I))/(1. - DP)
1583.        320 IF (DE .LT. PG(I)) PG(I) = DE
1584.      C
1585.        500 CONTINUE
```

```
 1586.       1000 CONTINUE
 1587.      C
 1588.      C              COMPUTE LARGEST GOMORY PENALTY AND TEST FOR FATHOMING
 1589.            PEN = 0.
 1590.            DO 2000 I=1,NROW
 1591.            IF (JF(I) .LE. NROW) GO TO 2000
 1592.            IF (PG(I) .GT. PEN) PEN = PG(I)
 1593.       2000 CONTINUE
 1594.            DP = X(IOBJ) + PEN + ZTOLZE
 1595.            IVAL = IDINT(DP)
 1596.            IF (DP .LT. 0.) IVAL = IVAL - 1
 1597.            IF (IVAL .GT. INCVAL) GO TO 3000
 1598.            IDIR = 0
 1599.            RETURN
 1600.      C
 1601.      C              PROBLEM NOT FATHOMED: CHECK FOR FORCED BRANCHES ON X(I)
 1602.       3000 NTEMP(2) = 0
 1603.            DO 3900 I=1,NROW
 1604.            IF (JF(I) .LE. NROW) GO TO 3900
 1605.            IF (PG(I) .GT. PD(I)) GO TO 3600
 1606.            DP = X(IOBJ) + PD(I) + ZTOLZE
 1607.            NTEMP(1) = IDINT(DP)
 1608.            IF (DP .LT. 0.) NTEMP(1) = NTEMP(1) - 1
 1609.            IF (NTEMP(1) .GT. INCVAL) GO TO 3900
 1610.      C              FORCED BRANCH UP ON X(I)
 1611.            IVAL = NTEMP(1)
 1612.            IDIR = -1
 1613.            NTEMP(1) = IPART(I) + 1
 1614.            GO TO 3700
 1615.      C
 1616.       3600 DP = X(IOBJ) + PD(I) + ZTOLZE
 1617.            NTEMP(1) = IDINT(DP)
 1618.            IF (DP .LT. 0.) NTEMP(1) = NTEMP(1) - 1
 1619.            IF (NTEMP(1) .GT. INCVAL) GO TO 3900
 1620.      C              FORCED BRANCH DOWN ON X(I)
 1621.            IVAL = NTEMP(1)
 1622.            IDIR = 1
 1623.            NTEMP(1) = IPART(I)
 1624.       3700 IROWP = I
 1625.            ICOL = JH(IROWP)
 1626.            REVBND = FLOAT(NTEMP(1))
 1627.            NTEMP(2) = 1
 1628.            CALL BRANCH
 1629.       3900 CONTINUE
 1630.            IF (NTEMP(2) .GT. 0) GO TO 5000
 1631.      C
 1632.      C              NO FORCED BRANCHES: CHOOSE BRANCHING VAR. AND DIRECTION
 1633.            PEN = 0.
 1634.            IROWP = 0
 1635.      C        DETERMINE BASIC VAR. X(IROWP) WITH MAX. UP- OR DOWN-PENALTY
 1636.            DO 4900 I=1,NROW
 1637.            IF (JF(I) .LE. NROW) GO TO 4900
 1638.            IF (PG(I) .GT. PD(I)) GO TO 4600
 1639.            IF (PD(I) .LE. PEN) GO TO 4900
 1640.            PEN = PD(I)
 1641.            IROWP = I
 1642.            IDIR = -1
 1643.            NTEMP(1) = IPART(I) + 1
 1644.            REVBND = FLOAT(NTEMP(1))
 1645.            GO TO 4900
 1646.       4600 IF (PU(I) .LE. PEN) GO TO 4900
```

```
      PEN = PU(I)
      IROWP = I
      IDIR = 1
      NTEMP(1) = IPART(I)
      REVBND = FLOAT(NTEMP(1))
 4900 CONTINUE
      IF (IROWP .GT. 0) GO TO 4950
C        EACH UP-- AND DOWN-PENALTY .EQ. 0. (DUAL-DEGENERACY)  CHOOSE ANY
C        NONINTEGER BASIC VAR. AS BRANCHING VAR.
      DO 4910 IROWP=1,NROW
      IF (JH(IROWP) .LE. NROW) GO TO 4910
      IF (DFPART(IROWP) .GE. ZTOLZE) GO TO 4920
 4910 CONTINUE
 4920 PEN = PU(IROWP)
      IDIR = 1
      NTEMP(1) = IPART(IROWP)
      REVBND = FLOAT(NTEMP(1))
 4950 ICOL = JH(IROWP)
      DP = X(IOBJ) - PEN + ZTOLZE
      NTEMP(1) = IDINT(DP)
      IF (DP .LT. 0.) NTEMP(1) = NTEMP(1) - 1
      IF (IVAL .GT. NTEMP(1)) IVAL = NTEMP(1)
C        BRANCH ON CHOSEN VARIABLE
      CALL BRANCH
 5000 IF (IDIR .EQ. -1) IPTYPE = 0
      IF (IDIR .EQ. 1) IPTYPE = -1
      RETURN
      END
      SUBROUTINE BRANCH
C
C        BRANCHES ON VARIABLE X(ICOL) AS DETERMINED IN SUBROUTINE PENLTS
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(4000)
      REAL A(2000)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DFPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               IPART(122),INCOMB(122),IVBND(500),IVID(500),
     2               IOBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
      COMMON DSUM,DPROD,DY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVERQ,ITERQ,IFEAS,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
C
C           ICOL INDEXES BRANCHING VARIABLE CHOSEN
C           IDIR INDICATES BRANCHING DIRECTION CHOSEN
C
C           ADD OPPOSITE DIRECTION TO LIST
      LISTL = LISTL + 1
      IF (IDIR .EQ. -1) IVBND(LISTL) = IDINT(XLB(ICOL) + ZTOLZE)
      IF (IDIR .EQ. 1) IVBND(LISTL) = IDINT(XUB(ICOL) - ZTOLZE)
      IVID(LISTL) = IDIR*ICOL
      IOBND(LISTL) = IVAL
C
C           REVISE BOUNDS ON BRANCHING VARIABLE FOR FORWARD DIRECTION
```

```
      IF (IDIR .EQ. -1) XLB(ICOL) = DBLE(REVBND)
      IF (IDIR .EQ. 1) XUB(ICOL) = DBLE(REVBND)
      RETURN
      END
      SUBROUTINE BKTRAK
C
C        BACKTRACKS TO SELECT A PROMISING (UNFATHOMED) NODE FROM THE
C        LIST OF STORED NODES.  LAST-IN-FIRST-OUT (LIFO) SELECTION RULE
C        IS EMPLOYED.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,IOBND
      DOUBLE PRECISION E(4000)
      REAL A(2000)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRG,QMA,QBA,QFI,
     1              QEQ,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DEPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               IPART(122),INCOMB(122),IVBND(500),IVID(500),
     2               IOBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
      COMMON DSUM,DPROD,CY,DE,DP,B(60),X(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFREZ,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NCELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
      NTEMP(3) = 0
C        IF LIST IS EMPTY, RETURN (COMPUTATIONS COMPLETED)
   50 IF (LISTL .EQ. 0) RETURN
      IF (IOBND(LISTL) .LE. INCVAL) GO TO 2000
C
C        GET NEXT NODE FROM LIST
      ICOL = IVID(LISTL)
      IF (ICOL .LT. 0) GO TO 100
C
      NTEMP(1) = IDINT(XLB(ICOL) + ZTOLZE)
      NTEMP(2) = IVBND(LISTL)
      XLB(ICOL) = XUB(ICOL) + 1.
      XUB(ICOL) = FLOAT(NTEMP(2))
      IF (KINBAS(ICOL) .GT. 0) GO TO 1000
      KINBAS(ICOL) = 0
      NTEMP(3) = 1
      GO TO 1000
C
  100 ICOL = -ICOL
      NTEMP(1) = IDINT(XUB(ICOL) + ZTOLZE)
      NTEMP(2) = IVBND(LISTL)
      XUB(ICOL) = XLB(ICOL) - 1.
      XLB(ICOL) = FLOAT(NTEMP(2))
      IF (KINBAS(ICOL) .GT. 0) GO TO 1000
      KINBAS(ICOL) = -1
      NTEMP(3) = 1
C
 1000 IVID(LISTL) = -IVID(LISTL)
      IVBND(LISTL) = NTEMP(1)
      IOBND(LISTL) = -10000
C
C              UPDATE X
      IF (NTEMP(3) .EQ. 0) RETURN
      CALL SHIFTR(1,3)
```

```
      DO 9000 J=1,NCOL
      IF (KINBAS(J)) 8800,8700,9000
 8800 DE = XUB(J)
      GO TO 8750
 8700 DE = XLB(J)
 8750 LL = LA(J)
      KK = LA(J+1) - 1
      DO 8800 I=LL,KK
      IR = IA(I)
 8800 Y(IR) = Y(IR) - A(I)*DE
 9000 CONTINUE
      CALL FTRAN(1)
      CALL SHIFTR(3,2)
      RETURN
C
C        NODE FATHOMED: UPDATE VAR. BOUNDS AND BACKTRACK AGAIN
 2000 ICOL = IVID(LISTL)
      IF (ICOL .LT. 0) GO TO 2100
C
      NTEMP(1) = IVBND(LISTL)
      IF (KINBAS(ICOL)) 2010,2050,2050
 2010 NTEMP(3) = 1
      DP = XUB(ICOL) - XLB(ICOL)
      DY = FLOAT(NTEMP(1)) - XUB(ICOL)
      IF (DP .LT. DY) KINBAS(ICOL) = 0
 2050 XUB(ICOL) = FLOAT(NTEMP(1))
      GO TO 3000
C
 2100 ICOL = -ICOL
      NTEMP(1) = IVBND(LISTL)
      IF (KINBAS(ICOL)) 2150,2110,2150
 2110 NTEMP(3) = 1
      DY = XLB(ICOL) - FLOAT(NTEMP(1))
      DP = XUB(ICOL) - XLB(ICOL)
      IF (DP .LT. DY) KINBAS(ICOL) = -1
 2150 XLB(ICOL) = FLOAT(NTEMP(1))
C
 3000 LISTL = LISTL - 1
      GO TO 50
C
      END
      SUBROUTINE WRAPUP
C
C        OUTPUTS OPTIMAL SOLUTION AND CORRESPONDING OBJECTIVE VALUE.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,IOBND
      DOUBLE PRECISION E(4000)
      REAL A(2000)
      COMMON/BLOCK2/ DEPART(60),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               IPART(122),INCOMB(122),IVBND(500),IVID(500),
     2               IOBND(500),NPIVOT,IPTYPE,RCOST,IFEAS
      COMMON DSJM,DPROD,DY,DE,DP,B(60),XL(60),Y(60),YTEMP(60),A,E,
     1       SUMINF,ICNAM(122,2),NAME(20),NTEMP(20),MSTAT,IOBJ,IROWP,
     2       IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,IFFEZ,JCOLP,NROW,NCOL,NELEM,
     3       NETA,NLELEM,NLETA,NUELEM,NUETA,JH(60),KINBAS(122),LA(122),
     4       LE(1002),IA(2000),IE(4000),XLB(122),XUB(122)
      IF (ITCNT .GE. ITRFRQ) GO TO 20
      IF (IFEAS .EQ. 0) GO TO 10
```

```
      WRITE (6,1)
    1 FORMAT (//' OPTIMAL INTEGER SOLUTION X(J), J=1,...,NCOL'//)
      WRITE (6,2) (INCJAB(J), J=1,NCOL)
    2 FORMAT (20I5)
      WRITE (6,3) INCVAL
    3 FORMAT (//' MAX OBJECTIVE VALUE =', I5)
      RETURN
   10 WRITE (6,4) INCVAL
    4 FORMAT (' NO FEASIBLE SOLUTION FOUND WITH OBJECTIVE VALUE .GT.',
     1        I10)
      RETURN
   20 WRITE (6,100) ITCNT
  100 FORMAT (' SIMPLEX ITERATIONS =',I3,':  COMPUTATIONS TERMINATED.')
      IF (IFEAS .EQ. 0) GO TO 10
      WRITE (6,101)
  101 FORMAT (//' BEST INTEGER SOLUTION FOUND IS'//)
      WRITE (6,2) (INCJAB(J), J=1,NCOL)
      RETURN
      END
```

```
*****SAMPLE INPUT DATA FOR BRANCH-AND-BOUND CODE BB*****


   1   1  1C9999          0
NAME            SAMPLE
ROWS
 N  OBJ
 L  ROW1
 L  ROW11
 L  ROW12
 L  ROW13
 L  ROW14
 L  ROW21
 L  ROW22
 L  ROW23
 L  ROW24
 L  ROW31
 L  ROW32
 L  ROW33
 L  ROW34
 L  ROW41
 L  ROW42
 L  ROW43
 L  ROW44
COLUMNS
    COL11     OBJ       -50.          ROW1      40.
    COL11     ROW11      16.          ROW12     38.
    COL11     ROW13       8.          ROW14     38.
    COL12     OBJ       -115.         ROW1      91.
    COL12     ROW11      92.          ROW12     39.
    COL12     ROW13      71.          ROW14     52.
    COL13     OBJ       -50.          ROW1      10.
    COL13     ROW11      41.          ROW12     32.
    COL13     ROW13      30.          ROW14     30.
    COL14     OBJ       -82.          ROW1      30.
    COL14     ROW11      16.          ROW12     71.
    COL14     ROW13      60.          ROW14     42.
    COL15     OBJ       -210.         ROW1     160.
    COL15     ROW11     150.          ROW12     80.
    COL15     ROW13     200.          ROW14     170.
    COL21     OBJ       -43.          ROW1      20.
    COL21     ROW21      25.          ROW22     20.
    COL21     ROW23      18.          ROW24      9.
    COL22     OBJ       -7.           ROW1       3.
    COL22     ROW21      4.           ROW22      5.
    COL22     ROW23      8.           ROW24      7.
    COL23     OBJ       -33.          ROW1      12.
    COL23     ROW21      16.          ROW22     40.
    COL23     ROW23      30.          ROW24     20.
    COL24     OBJ       -5.           ROW1       3.
    COL24     ROW21      6.           ROW22      8.
    COL24     ROW23      4.
    COL25     OBJ       -12.          ROW1      18.
    COL25     ROW22      14.          ROW23      8.
    COL25     ROW24      3.
    COL31     OBJ       -32.          ROW1       9.
    COL31     ROW31      12.          ROW32     30.
    COL31     ROW33      31.          ROW34     21.
    COL32     OBJ       -20.          ROW1      25.
```

```
    COL32     ROW31     6.              ROW32     15.
    COL32     ROW33     6.              ROW34     4.
    COL33     OBJ       -4.             ROW1      1.
    COL33     ROW31     2.              ROW33     3.
    COL33     ROW34     1.
    COL34     OBJ       -5.             ROW1      1.
    COL34     ROW31     1.              ROW32     1.
    COL34     ROW34     2.
    COL35     OBJ       -43.            ROW1      10.
    COL35     ROW32     25.             ROW33     18.
    COL35     ROW34     14.
    COL41     OBJ       -426.           ROW1      280.
    COL41     ROW41     200.            ROW42     100.
    COL41     ROW43     60.             ROW44     310.
    COL42     OBJ       -42.            ROW1      10.
    COL42     ROW41     20.             ROW43     21.
    COL42     ROW44     8.
    COL43     OBJ       -12.            ROW1      8.
    COL43     ROW41     6.              ROW42     20.
    COL43     ROW43     4.              ROW44     4.
    COL44     OBJ       -8.             ROW1      1.
    COL44     ROW41     2.              ROW42     3.
    COL44     ROW44     6.
    COL45     OBJ       -2.             ROW1      1.
    COL45     ROW41     1.              ROW43     2.
    COL45     ROW44     1.
RHS
    RHS1      ROW1      293.
    RHS1      ROW11     221.            ROW12     182.
    RHS1      ROW13     258.            ROW14     232.
    RHS1      ROW21     36.             ROW22     64.
    RHS1      ROW23     46.             ROW24     27.
    RHS1      ROW31     16.             ROW32     48.
    RHS1      ROW33     41.             ROW34     29.
    RHS1      ROW41     150.            ROW42     86.
    RHS1      ROW43     61.             ROW44     230.
ENDATA
    0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.
    0.   0.   0.   0.   0.
    1.   1.   1.   1.   1.   1.   1.   1.   1.   1.   1.   1.   1.   1.   1.
    1.   1.   1.   1.   1.
```

```
*****OUTPUT GENERATED BY BB ON SAMPLE PROBLEM*****


PROBLEM     1
TIME =    2.94 SECONDS:  INCVAL =        414
TIME =    3.07 SECONDS:  INCVAL =        431
TIME =    3.53 SECONDS:  INCVAL =        445
TIME =    3.56 SECONDS:  INCVAL =        462
TOTAL SOLUTION TIME =    6.65 SECONDS


OPTIMAL INTEGER SOLUTION X(J), J=1,...,NCOL

 462   21   56    2   89   70    3   13   10    8    5    9   14    8  131   63   34  211    1    1
   1    1    0    1    1    0    1    1    0    1    1    1    1    0    1    1    1    1


MAX OBJECTIVE VALUE =  462
```

APPENDIX II

LISTING AND SAMPLE INPUT/OUTPUT FOR COMPUTER PROGRAM DSLC

```
      C
      C         DECOMPOSITION ROUTINE, DSLC
      C         DECOMPOSITION AND SOLUTION OF PURE-INTEGER LINEAR PROGRAMS
      C         (WITH GENERAL INTEGER VARIABLES) WITH BLOCK ANGULAR CONSTRAINT
      C         MATRICES.
      C********SINGLE LINKING CONSTRAINT CASE********
      C         GARY A. KOCHMAN (OPERATIONS RESEARCH), SEPT. 1976, MOD 1
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1         INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            INTEGER*2 IPART,INCOMB,IVBND,IVID,IOBND
            INTEGER*2 IZSTAR,IBID,IBBND,IZBND
            INTEGER*2 IAS,LAS
            DOUBLE PRECISION E(1000)
            DIMENSION A(500),INC(10),AV(10),NTSOL(10)
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
            COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
           1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
           2               IVBND(500),IVID(500),IOBND(500)
            COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
           1IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
           2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
           3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
            COMMON/BLOCK4/ XLBS(50,10),XUBS(50,10),AMATRS(1000),RHS(20,10),
           1NROWS(10),NCOLS(10),JFIRST(11),IAS(1000),LAS(50,10)
            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      C
      C         MAIN PROGRAM
      C         COORDINATES APPLICATION OF THE VARIOUS STEPS OF THE DECOMPOSI-
      C         TION ALGORITHM.
      C
      C               INITIALIZE PARAMETERS
        10 CALL PCLOCK(ITOT)
           READ (5,1,END=9999) NSUBS,IB,IPARM,IZINC
         1 FORMAT (4I10)
           DO 20 J=1,122
           ICNAM(J,1) = 0
        20 ICNAM(J,2) = 0
           IOBJ = 1
           ITRFRQ = 99999
           IBLL = 0
           IPAR = 0
           NJFIX = 0
           JFIXSM = 0
           JFIRST(1) = 1
           TALC = 0.
           TSUBS = 0.
           IFIRST = 0
           NALC = 0
      C
      C               PHASE 1
           DO 100 J=1,NSUBS
           AV(J) = 0.
```

```
      NTSOL(J) = 0
      NSOL(J) = 0
      JFIX(J) = 0
C        INPUT DATA FOR SUBPROBLEM J
      CALL INPUT(J)
C        SOLVE SUBPROBLEM J AS LP, PARAMETRICALLY IN THE RIGHT-HAND
C        SIDE FOR THE SINGLE LINKING CONSTRAINT
  100 CALL PHASE1(J)
      WRITE (6,8)
    8 FORMAT ('1 START NEW PROBLEM')
C        OUTPUT SLOPES AND (ALL-INTEGER) BREAKPOINTS DEFINING THE
C        SEGMENTS OF THE (MODIFIED) CONCAVE AND PIECEWISE LINEAR
C        (PARAMETRIC) OBJECTIVE FUNCTIONS FOR THE SUBPROBLEMS
      DO 200 J=1,NSUBS
      LAST = NLAMDA(J) + 1
      WRITE (6,2) J,(IBP(K,J), K=1,LAST)
    2 FORMAT (//' IBP(K,',I3,') =',2015/2515/515)
      LAST = LAST - 1
      WRITE (6,3) J,(RLAMDA(K,J), K=1,LAST)
    3 FORMAT (' RLAMDA(K,',I3,') =',19F6.2/20F6.2/11F6.2)
  200 WRITE (6,4) J,ZNAUT(J)
    4 FORMAT (' ZNAUT(',I3,') =',F10.4)
      WRITE (6,5)
    5 FORMAT (///)
      CALL PCLOCK(JTIME,ITOT)
      TPH1 = JTIME/100.
      GO TO 2100
C
C              PHASE 2
C
C        BACKTRACK TO LAST PROMISING NODE IN ALLOCATION-VARIABLES SEARCH
C        TREE
 2000 CALL BBKTRK
      IPAR = 0
C        IF LIST OF STORED NODES IS EMPTY, COMPUTATIONS ARE COMPLETED
      IF (IBLL .EQ. 0) GO TO 9000
 2100 NALO = NALO + 1
      CALL PCLOCK(JTIME,ITOT)
C        SOLVE MASTER PROBLEM TO OBTAIN PROMISING NEW ALLOCATION OF
C        LINKING RESOURCE TO SUBPROBLEMS
      CALL ALOCTE(IPAR,IRHS)
      CALL PCLOCK(JTIME1,ITOT)
      TALO = TALO + (JTIME1 - JTIME)/100.
      IPAR = 1
C        TEST FOR FATHOMING OF CURRENT NODE
      IF (INCVAL .LE. IZINC) GO TO 2000
      IZBAR = INCVAL
      IF (IRHS .GT. 0) IZBAR = -10000
C        CHECK IF ANY SUBPROBLEM HAS PREVIOUSLY BEEN SOLVED UNDER ITS
C        CURRENT ALLOCATION
      CALL COMCHK(J,K)
      IF (ICOM .EQ. 0) GO TO 3000
      LAST = NSOL(J) + 1
      IRHS = IBSTAR(LAST,J) - IBUSED(K,J)
C        BRANCH ON ALLOCATION VARIABLE IN SUBPROBLEM J
 2200 CALL CMBRAN(J,K)
      IF (MSTAT .EQ. 0) GO TO 2000
C        FIX ALLOCATION TO SUBPROBLEM J AT LEVEL K
 2300 JFIX(J) = K
      NJFIX = NJFIX + 1
      JFIXSM = JFIXSM + IZSTAR(K,J)
```

```
      IF (NJFIX .LT. NSUBS) GO TO 2100
C           ALL SUBS FIXED: COMPARE SOLUTION WITH INCUMBENT
      IF (JFIXSM .LE. IZINC) GO TO 2500
      DO 2400 I=1,NSUBS
 2400 INC(I) = JFIX(I)
      IZINC = JFIXSM
      CALL PCLOCK(JTIME,ITOT)
      TOPT = JTIME/100.
      WRITE (6,93) TOPT,IZINC
   93 FORMAT (' TIME =',F7.2,' SECONDS:  IZINC =',I10)
      IF (IFIRST .EQ. 1) GO TO 2500
      TFIRST = TOPT
      IFIRST = 1
      IFVAL = IZINC
 2500 JFIX(J) = 0
      JFIXSM = JFIXSM - IZSTAR(K,J)
      NJFIX = NJFIX - 1
      GO TO 2000
C
C         SELECT NEXT SUBPROBLEM TO SOLVE
 3000 AVMAX = -10000.
      J = 0
C         FIRST TRY TO SELECT AN UNFIXED SUBPROBLEM FOR WHICH THE ALLOCA-
C         TION VARIABLE IS NOT ON AN UP-BRANCH
      DO 3100 I=1,NSUBS
      IF (JFIX(I) .NE. 0) GO TO 3100
      IF (AVMAX .GE. AV(I)) GO TO 3100
      J = I
      AVMAX = AV(I)
 3100 CONTINUE
      IF (J .GT. 0) GO TO 3200
C         ALLOCATION VARIABLE FOR EACH UNFIXED SUBPROBLEM CURRENTLY ON AN
C         UP-BRANCH; SELECT UNFIXED SUBPROBLEM IN ACCORDANCE WITH
C         DECISION RULE SPECIFIED BY PARAMETER IPARM ALONE
      DO 3150 I=1,NSUBS
      IF (JFIX(I) .GT. 0) GO TO 3150
      IF (AVMAX .GE. AV(I)) GO TO 3150
      J = I
      AVMAX = AV(I)
 3150 CONTINUE
 3200 NTSOL(J) = NTSOL(J) + 1
      CALL PCLOCK(JTIME,ITOT)
C         LOAD PROBLEM DATA FOR SUBPROBLEM J
      CALL LDDATA(J)
      ITSINV = 99999
      IFEAS = 0
C         GET OPTIMAL LP-SOLUTION TO SUBPROBLEM J
      CALL NORMAL
C         TEST IF SUBPROBLEM J IS LP-FEASIBLE
      IF (MSTAT .EQ. 2) GO TO 3300
C         SUBPROBLEM J LP-FEASIBLE; GET INITIAL LOWER BOUND ESTIMATE ON
C         MAX. IP-OBJ. VAL.
      CALL GETLBD(J)
      INITBD = INCUMB(1)
C         USE BRANCH-AND-BOUND ROUTINE TO COMPUTE OPT. IP-SOLUTION TO
C         SUBPROBLEM J
      CALL BANDB(INITBD)
 3300 CALL PCLOCK(JTIME1,ITOT)
      TIME = (JTIME1 - JTIME)/100.
      TSUBS = TSUBS + TIME
C         UPDATE SUBPROBLEM SELECTION RULE INFORMATION FOR SUBPROBLEM J,
```

```
C         IN ACCORDANCE WITH VALUE SPECIFIED FOR IPARM
      GO TO (3310,3320,3330,3400),IPARM
 3310 AV(J)=((NTSOL(J)-1)*AV(J)+(ZJBAR(J)-INCVAL))/NTSOL(J)
      GO TO 3400
 3320 AV(J) = ((NTSOL(J)-1)*AV(J) + INCUMB(2))/NTSOL(J)
      GO TO 3400
 3330 AV(J) = ((NTSOL(J)-1)*AV(J) - TIME)/NTSOL(J)
C         SAVE ALLOCATION AND SOLUTION INFORMATION FOR SUBPROBLEM J
 3400 CALL SAVER(J)
      IRHS = INCUMB(2)
      K = NTEMP(2)
C         OUTPUT SOLUTION INFORMATION FOR SUBPROBLEM J
      WRITE (6,6) J,K,(INCUMB(KK), KK=1,NCOL)
    6 FORMAT (' SUB',I2,' SOL',I3,':  ',20I5)
C         TEST IF ALL SUBPROBLEMS HAVE BEEN SOLVED; IF SO, NO BRANCHING
C         NECESSARY
      IF (NJFIX - NSUBS + 1) 2200,2300,9999
C
C         COMPUTATIONS COMPLETED; OUTPUT OPTIMAL SOLUTION INFORMATION
 9000 CALL OUTSOL(INC)
      CALL PCLOCK(JTIME,ITOT)
      TTOT = JTIME/100.
C         OUTPUT ALGORITHMIC PERFORMANCE INFORMATION
      WRITE (6,7) TTOT
    7 FORMAT (/' TOTAL SOLUTION TIME =',F7.2,' SECONDS')
      WRITE (6,92) TPH1,NALO,TALO,(NTSOL(J), J=1,NSUBS)
   92 FORMAT (' TIME IN PHASE1 =',F6.2,' SECONDS'/' MASTER PROBLEM SOLVE
     1D',I10,' TIMES:  TOTAL TIME SOLVING MASTER PROBLEM =',F6.2,' SECON
     2DS'/' NUMBER OF TIMES EACH SUBPROBLEM SOLVED =',18X,10I5)
      WRITE (6,9) (NSOL(J), J=1,NSUBS)
    9 FORMAT (' NUMBER OF DISTINCT SOLUTIONS OBTAINED IN EACH SUBPROBLEM
     1 =',10I5)
      WRITE (6,91) TSUBS
   91 FORMAT (' TOTAL TIME SOLVING SUBPROBLEMS =',F6.2,
     1' SECONDS')
      GO TO 10
 9999 STOP
      END
      BLOCK DATA
C
C         INITIALIZES GLOBAL PROGRAM CONSTANTS
C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      REAL ZTOLZE/1.E-5/,ZTOLPV/1.E-4/,ZTCOST/1.E-3/,ZTOLRJ/1.E-4/,
     1     ZTOLSM/1.E-10/
      INTEGER*4 NRMAX/60/,NTMAX/250/,NEMAX/1000/
      INTEGER*4 QRO/'ROW '/,QMA/'MATR'/,QBA/'BASI'/,QFI/'FIRS'/,
     1QEC/'ECF '/,QBL/'    '/,QA/'A   '/,QPL/' +  '/,QMI/' -  '/,
     2QZ/' Z  '/,QI/'I   '/,QF/'F   '/,QN/'N   '/,QU/'U   '/,
     3QB/'B   '/,QC/'C   '/,QE/'E   '/,QH/'H   '/,QL/'L   '/,
     4QO/'O   '/,QR/'R   '/,QM/'M   '/,QG/'G   '/
      END
      SUBROUTINE INPUT(IFPROB)
C
C         INPUTS PROBLEM DATA
```

```
C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
C********DESCRIPTION OF PARAMETERS********
C         IFPROB = INPROB = IDENTIFICATION NUMBER OF SUBPROBLEM BEING
C         INPUT (USER MUST INPUT SUBPROBLEMS IN SEQUENCE 1,2,...,NSUBS)
C**************************************
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IZSTAR,IBID,IBBND,IZBND
      INTEGER*2 IAS,LAS
      DOUBLE PRECISION E(1000),ATEMP1,ATEMP2
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QF,QL,QO,QR,QM,QG,NTMAX
C
      COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
     1IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
     2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
     3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
      COMMON/BLOCK4/ XLBS(50,10),XUBS(50,10),AMATRS(1000),RHS(20,10),
     1NROWS(10),NCOLS(10),JFIRST(11),IAS(1000),LAS(50,10)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      DO 10 I=1,60
   10 B(I) = 0.
      DO 20 J=1,122
   20 KINBAS(J) = 0
      NROW = 0
      ITCNT = 0
      ICS1=0
      ICS2=0
      READ (5,7000) INPROB
 7000 FORMAT (5I4)
      IFPROB = INPROB
      IF (IFPROB .EQ. 0) RETURN
C
    5 READ(5,101) K1,K2,K3,K4,(NAME(I),I=1,4),ATEMP1,NAME(5),NAME(6),
     1ATEMP2
  101 FORMAT(4A1,2A4,2X,2A4,2X,F12.4,3X,2A4,2X,F12.4)
      IF(K1 .EQ. QE) GO TO 600
      IF(K1 .EQ. QBL) GO TO 50
      IF(K1 .EQ. QN) GO TO 100
      IF(K1 .EQ. QR .AND. K2 .EQ. QO) L=1
      IF(K1 .EQ. QR .AND. K2 .EQ. QO) GO TO 150
      IF(K1 .EQ. QC) L=2
      IF(K1 .EQ. QC) GO TO 150
      IF(K1 .EQ. QB .AND. K2 .EQ. QA) L=3
      IF(K1 .EQ. QB .AND. K2 .EQ. QA) GO TO 150
      IF(K1 .EQ. QR .AND. K2 .EQ. QH) L=4
      IF(K1 .EQ. QR .AND. K2 .EQ. QH) GO TO 150
   50 GO TO(210,320,410,500),L
  100 NTEMP(1) = NAME(3)
```

```
305.            NTEMP(2) = NAME(4)
306.          150 CONTINUE
307.            GO TO 5
308.      C
309.          210 NROW=NROW+1
310.            NCOL=NROW
311.            ICNAM(NROW,1) = NAME(1)
312.            ICNAM(NROW,2) = NAME(2)
313.      C
314.      C         TEST ROW TYPE
315.      C
316.            IF(K2.EQ.QL .OR. K3.EQ.QL)  GO TO 220
317.            IF(K2.EQ.QE .OR. K3.EQ.QE)  GO TO 230
318.            IF(K2.EQ.QG .OR. K3.EQ.QG)  GO TO 240
319.            IF(K2.EQ.QN .OR. K3.EQ.QN)  GO TO 250
320.            GO TO 230
321.          220 XLB(NROW) = 0.
322.            XUB(NROW) = 1.E4
323.            GO TO 250
324.          230 XLB(NROW) = 0.
325.            XUB(NROW) = 0.
326.            GO TO 250
327.          240 XLB(NROW) = 0.
328.            XUB(NROW) = 1.E4
329.            A(NROW) = -1.
330.            GO TO 260
331.          250 A(NROW) = 1.
332.          260 IA(NROW) = NROW
333.            LA(NROW) = NROW
334.            JH(NROW) = NROW
335.            KINBAS(NROW) = NROW
336.            NELEM=NROW
337.            GO TO 5
338.      C
339.      C         MATRIX ELEMENTS
340.      C
341.          320 J = 3
342.            K = 4
343.            IF (DABS(ATEMP1) .LE. ZTOLZE)  GO TO 321
344.            GO TO 324
345.          321 J=5
346.            K=6
347.            ATEMP1=ATEMP2
348.            GO TO 330
349.          324 CONTINUE
350.            IF (NAME(1) .EQ. ICS1 .AND. NAME(2) .EQ. ICS2)  GO TO 330
351.      C
352.      C         TEST FOR SPLIT VECTOR
353.      C
354.            DO 325 I = 1,NCOL
355.            IF (NAME(1) .NE. ICNAM(I,1))  GO TO 325
356.            IF (NAME(2) .NE. ICNAM(I,2))  GO TO 325
357.            WRITE(6,8250) NAME(1),NAME(2)
358.         8250 FORMAT(14HOSPLIT VECTOR ,2A4)
359.          325 CONTINUE
360.            NCOL = NCOL + 1
361.            ICS1 = NAME(1)
362.            ICS2 = NAME(2)
363.            ICNAM(NCOL,1) = ICS1
364.            ICNAM(NCOL,2) = ICS2
365.            LA(NCOL) = NELEM + 1
```

```
      C
      C          TEST FOR ROW MATCH
      C
  330 DO 340 I = 1,NROW
          IF(NAME(J) .NE. ICNAM(I,1) .OR. NAME(K) .NE. ICNAM(I,2))GO TO 340
          NELEM = NELEM + 1
          IA(NELEM) = I
          A(NELEM) = ATEMP1
          LA(NCOL+1)=NELEM+1
  335 IF(K .GT. 5)  GO TO 5
          IF(DABS(ATEMP2) .LE. ZTOLZE)  GO TO 5
          J = 5
          K = 6
          ATEMP1 = ATEMP2
          GO TO 330
  340 CONTINUE
          WRITE(6,8300) NAME(J),NAME(K),NAME(1),NAME(2)
 8300 FORMAT(18HONO MATCH FOR ROW ,2A4,10HAT COLUMN ,2A4)
          STOP
      C
      C          BASIS CARDS
      C
  410 DO 420 I = 1,NCOL
          IF(NAME(1) .NE. ICNAM(I,1) .OR. NAME(2) .NE. ICNAM(I,2)) GO TO 420
          IBVEC = I
          GO TO 425
  420 CONTINUE
          WRITE(6,8400) NAME(1),NAME(2)
 8400 FORMAT(21HONO MATCH FOR VECTOR ,2A4)
          GO TO 5
  425 DO 430 I = 1,NROW
          IF (NAME(3).NE.ICNAM(I,1).OR.NAME(4).NE.ICNAM(I,2))  GO TO 430
          IBROW=I
          GO TO 440
  430 CONTINUE
          WRITE(6,8300) NAME(3),NAME(4)
          GO TO 5
  440 JH(IBROW) = IBVEC
          KINBAS(IBROW) = 0
          KINBAS(IBVEC) = IBROW
          GO TO 5
      C
      C          RHS
      C
  500 J = 3
          K = 4
          IF (DABS(ATEMP1) .LE. ZTOLZE)  GO TO 521
          GO TO 524
  521 J=5
          K=6
          ATEMP1=ATEMP2
          GO TO 530
  524 CONTINUE
          IF (NAME(1) .EQ. ICS1 .AND. NAME(2) .EQ. ICS2)  GO TO 530
      C
      C          TEST FOR SPLIT VECTOR
      C
          DO 525 I = 1,NCOL
          IF (NAME(1) .NE. ICNAM(I,1))  GO TO 525
          IF (NAME(2) .NE. ICNAM(I,2))  GO TO 525
          WRITE(6,8250) NAME(1),NAME(2)
```

```
  525 CONTINUE
      ICS1 = NAME(1)
      ICS2 = NAME(2)
C
C        TEST FOR ROW MATCH
C
  530 DO 540 I = 1,NROW
      IF(NAME(J) .NE. ICNAM(I,1) .OR. NAME(K) .NE. ICNAM(I,2))GO TO 540
      B(I) = ATEMP1
  535 IF(K .GT. 5)  GO TO 5
      IF(DABS(ATEMP2) .LE. ZTOLZE)  GO TO 5
      J = 5
      K = 6
      ATEMP1 = ATEMP2
      GO TO 530
  540 CONTINUE
      WRITE(6,8300) NAME(J),NAME(K)
      STOP
C
C           END OF INPUT
C
  600 NSCOL = NCOL - NROW
      K = NROW + 1
C        INPUT LOWER AND UPPER BOUNDS ON DECISION VARIABLES FOR SUBPROB-
C        LEM INPROB
      READ (5,650) (XLB(J), J=K,NCOL)
      READ (5,650) (XUB(J), J=K,NCOL)
  650 FORMAT (15F5.0)
      XLB(ICBJ) = -10000.
      XUB(ICBJ) = 10000.
C        INPUT LOWER AND UPPER BOUNDS ON ALLOCATION VARIABLE FOR SUB-
C        PROBLEM INPROB
      READ (5,651) IBLB(INPROB),IBJB(INPROB)
  651 FORMAT (8I10)
      TEMP = FLOAT(IBLB(INPROB))
      B(2) = DBLE(TEMP)
      INVERC = NROW
C              STORE A,IA,LA,XUB,XLB,B,NCOL,NROW FOR SUBPROBLEM INPROB
      LL = JFIRST(INPROB) - 1
      DO 1010 I=1,NELEM
      LLPI = LL + I
      AMATRS(LLPI) = A(I)
 1010 IAS(LLPI) = IA(I)
      DO 1020 K=1,NCOL
      LAS(K,INPROB) = LA(K)
      XLBS(K,INPROB) = XLB(K)
 1020 XUBS(K,INPROB) = XUB(K)
      LL = NROW - 2
      DO 1030 I=1,LL
 1030 RHS(I,INPROB) = SNGL(B(I+2))
      NCOLS(INPROB) = NCOL
      NROWS(INPROB) = NROW
      JFIRST(INPROB+1) = JFIRST(INPROB) + NELEM
C
      NELEM = NELEM - NROW
      RELEM = NELEM
      RDENS = RELEM/(NROW*NSCOL)
      RETURN
      END
      SUBROUTINE FTRAN(IPAR)
C
```

2660

```
488.      C         PERFORMS FORWARD  TRANSFORMATION ON COLUMN STORED IN VECTOR Y
489.      C********DESCRIPTION OF PARAMETERS********
490.      C         IPAR = PARAMETER INDICATING BY WHICH ETA-VECTORS THE MATRIX
491.      C         COLUMN IS TO BE UPDATED
492.      C*****************************************
493.      C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
494.      C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
495.      C
496.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
497.           1          INTEGER*4 (I-N,Q)
498.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
499.            DOUBLE PRECISION E(1000)
500.            REAL A(500)
501.      C
502.            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
503.           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
504.           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
505.      C
506.            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
507.           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
508.           2       IOBJ,IRCWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
509.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
510.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
511.      C
512.            GO TO (100,110), IPAR
513.        100 NFE = 1
514.            NLE = NETA
515.            GO TO 200
516.        110 NFE = NLETA + 1
517.            NLE = NETA
518.        200 IF (NFE .GT. NLE) GO TO 9000
519.            DO 1000 IK = NFE,NLE
520.            LL = LE(IK)
521.            KK = LE(IK+1) - 1
522.            IPIV = IE(LL)
523.            DY = Y(IPIV)
524.            DY = DY/E(LL)
525.            Y(IPIV) = DY
526.            IF (KK .LE. LL) GO TO 1000
527.            LL = LL + 1
528.            DO 500 J = LL,KK
529.            IR = IE(J)
530.            Y(IR) = Y(IR) - E(J) * DY
531.        500 CONTINUE
532.       1000 CONTINUE
533.       9000 CONTINUE
534.            RETURN
535.            END
536.            SUBROUTINE BTRAN
537.      C
538.      C         PERFORMS BACKWARD TRANSFORMATION ON COLUMN STORED IN VECTOR Y
539.      C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
540.      C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
541.      C
542.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
543.           1          INTEGER*4 (I-N,Q)
544.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
545.            DOUBLE PRECISION E(1000)
546.            REAL A(500)
547.      C
548.            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
```

```
     1               QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2               QE,QH,QL,QO,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      IF (NETA .LE. 0) GO TO 9000
      DO 1000 I = 1,NETA
      IK = NETA - I + 1
      LL = LE(IK)
      KK = LE(IK+1) - 1
      IPIV = IE(LL)
      DP = E(LL)
      DY = Y(IPIV)
      DSUM = 0.
      IF (KK .LE. LL) GO TO 600
      LL = LL + 1
      DO 500 J = LL,KK
      IR = IE(J)
      DE = E(J)
      DPROD = DE * Y(IR)
      DSUM = DSUM + DPROD
  500 CONTINUE
C
  600 Y(IPIV) = (DY - DSUM) / DP
 1000 CONTINUE
 9000 RETURN
      END
      SUBROUTINE FORMC
C
C        FORMS COST VECTOR FOR CURRENT PRIMAL-SIMPLEX ITERATION; USED IN
C        CONJUNCTION WITH SUBROUTINE PRICE
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      MSTAT = QI
      IFFEZ = 0
      SUMINF = 0.
      Y(IOBJ) = 0.
      DO 30 I=1,NROW
      IF (I .EQ. IOBJ) GO TO 30
      ICOL = JH(I)
```

```
      IF (X(I) .LE. (XLB(ICOL) - ZTOLZE)) GO TO 10
      IF (X(I) .GE. (XUB(ICOL) + ZTOLZE)) GO TO 20
      Y(I) = 0.
      GO TO 30
   10 Y(I) = 1.
      SUMINF = SUMINF + XLB(ICOL) - X(I)
      GO TO 30
   20 Y(I) = -1.
      SUMINF = SUMINF + X(I) - XUB(ICOL)
   30 CONTINUE
      IF (SUMINF .GT. ZTOLSM) RETURN
      Y(IOBJ) = 1.
      IFEZ = 1
      MSTAT = 0
      RETURN
      END
      SUBROUTINE PRICE
C
C        PRICES OUT NONBASIC COLUMNS, CHOOSES PIVOT COLUMN JCOLP FOR
C        CURRENT PRIMAL-SIMPLEX ITERATION
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,IGBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QF,QL,QO,QR,QM,QG,NTMAX
C
      COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
     2               IVBND(500),IVID(500),IGBND(500)
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      CMIN = 1.E10
      CMAX = -1.E10
      DO 1000 J=1,NCOL
      IF (KINBAS(J) .GT. 0) GO TO 1000
      IF ((XUB(J) - XLB(J)) .LT. ZTOLZE) GO TO 1000
      DSUM = 0.
      LL = LA(J)
      KK = LA(J+1) - 1
      DO 500 I = LL,KK
      IR = IA(I)
      DE = A(I)
      DPROD = DE * Y(IR)
      DSUM = DSUM + DPROD
  500 CONTINUE
      IF (KINBAS(J) .EQ. -1) GO TO 600
      IF (DSUM .GE. CMIN) GO TO 1000
      CMIN = DSUM
```

```
671.            JCOL1 = J
672.            GO TO 1000
673.      600   IF (DSUM .LE. CMAX) GO TO 1000
674.            CMAX = DSUM
675.            JCOL2 = J
676.     1000   CONTINUE
677.      C
678.            IF (CMIN .LE. -ZTCOST) GO TO 1500
679.            IF (CMAX .GE. ZTCOST) GO TO 2000
680.            JCOLP = 0
681.            RCOST = 0.
682.            RETURN
683.     1500   IF (CMAX .GE. ZTCOST) GO TO 2500
684.     1600   JCOLP = JCOL1
685.            RCOST = CMIN
686.            RETURN
687.     2000   JCOLP = JCOL2
688.            RCOST = CMAX
689.            RETURN
690.     2500   IF (ABS(CMIN) - CMAX) 2000,2000,1600
691.            END
692.            SUBROUTINE CHUZR
693.      C
694.      C        PERFORMS MIN-RATIO TEST FOR PIVOT COLUMN JCOLP DETERMINED IN
695.      C        SUBROUTINE PRICE.  SELECTS PIVOT ROW IROWP FOR CURRENT PRIMAL-
696.      C        SIMPLEX ITERATION.
697.      C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
698.      C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
699.      C
700.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
701.           1         INTEGER*4 (I-N,Q)
702.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
703.            INTEGER*2 IPART,INCOMB,IVBND,IVID,IOBND
704.            DOUBLE PRECISION E(1000)
705.            REAL A(500)
706.      C
707.            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
708.           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
709.           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
710.      C
711.            COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
712.           1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
713.           2               IVBND(500),IVID(500),IOBND(500)
714.      C
715.            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
716.           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
717.           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVERQ,ITRERQ,ITSINV,IFEAZ,
718.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
719.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
720.      C
721.            IF (KINBAS(JCOLP) .EQ. -1) GO TO 1000
722.      C
723.      C               INCOMING VARIABLE AT LOWER BOUND
724.      C
725.            DP = 1.E10
726.            DO 500 I=1,NROW
727.            IF (I .EQ. IOBJ) GO TO 500
728.            ICOL = JH(I)
729.            IF (Y(I) .GT. ZTOLPV) GO TO 100
730.            IF (Y(I) .LT. -ZTOLPV) GO TO 200
731.            GO TO 500
```

```
C                POSITIVE COEFFICIENT
  100 IF (X(I) .LT. (XLB(ICOL) - ZTOLZE)) GO TO 500
      DE = (X(I) - XLB(ICOL))/Y(I)
      IF (DE .GE. DP) GO TO 500
      IPTYPE = 0
      GO TO 250
C                NEGATIVE COEFFICIENT
  200 IF (X(I) .GT. (XUB(ICOL) + ZTOLZE)) GO TO 500
      DE = (X(I) - XUB(ICOL))/Y(I)
      IF (DE .GE. DP) GO TO 500
      IPTYPE = -1
  250 DP = DE
      IROWP = I
  500 CONTINUE
      DE = DP + XLB(JCOLP)
      IF (DE .LT. XUB(JCOLP)) GO TO 600
      DP = XUB(JCOLP) - XLB(JCOLP)
      NPIVOT = 0
      RETURN
  600 NPIVOT = 1
      RETURN
C
C                INCOMING VARIABLE AT UPPER BOUND
C
 1000 DP = -1.E10
      DO 1500 I=1,NROW
      IF (I .EQ. IOBJ) GO TO 1500
      ICOL = JH(I)
      IF (Y(I) .GT. ZTOLPV) GO TO 1100
      IF (Y(I) .LT. -ZTOLPV) GO TO 1200
      GO TO 1500
C                POSITIVE COEFFICIENT
 1100 IF (X(I) .GT. (XUB(ICOL) + ZTOLZE)) GO TO 1500
      DE = (X(I) - XUB(ICOL))/Y(I)
      IF (DE .LE. DP) GO TO 1500
      IPTYPE = -1
      GO TO 1250
C                NEGATIVE COEFFICIENT
 1200 IF (X(I) .LT. (XLB(ICOL) - ZTOLZE)) GO TO 1500
      DE = (X(I) - XLB(ICOL))/Y(I)
      IF (DE .LE. DP) GO TO 1500
      IPTYPE = 0
 1250 DP = DE
      IROWP = I
 1500 CONTINUE
      DE = DP + XUB(JCOLP)
      IF (DE .GT. XLB(JCOLP)) GO TO 1600
      DP = XLB(JCOLP) - XUB(JCOLP)
      NPIVOT = 0
      RETURN
 1600 NPIVOT = 1
      RETURN
      END
      SUBROUTINE WRETA
C
C        FORMS NEW ETA-VECTOR CORRESPONDING TO CURRENT SIMPLEX PIVOT
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
```

```
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      NELEM = NELEM + 1
      IE(NELEM) = IROWP
      E(NELEM) = Y(IROWP)
C
      DO 1000 I = 1,NROW
      IF (I .EQ. IROWP) GO TO 1000
      IF (DABS(Y(I)) .LE. ZTOLZE) GO TO 1000
      NELEM = NELEM + 1
      IE(NELEM) = I
      E(NELEM) = Y(I)
 1000 CONTINUE
C
      NETA = NETA + 1
      LE(NETA+1) = NELEM + 1
      RETURN
      END
      SUBROUTINE SHIFTR(IOLD,INEW)
C
C        REARRANGES DATA STORAGE
C********DESCRIPTION OF PARAMETERS********
C        IOLD, INEW = PARAMETERS INDICATING STORAGE LOCATIONS FROM AND
C        TO WHICH DATA IS TO BE TRANSFERRED, RESPECTIVELY
C****************************************
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      DIMENSION BARRAY(240)
      EQUIVALENCE (BARRAY(1),B(1))
      IFO = (IOLD - 1) * NRMAX
      IFN = (INEW - 1) * NRMAX
C
```

```
      DO 1000 I = 1,NROW
      BARRAY(IFN + I) = BARRAY(IFO + I)
 1000 CONTINUE
      RETURN
      END
      SUBROUTINE INVERT
C
C        COMPUTES INVERSE OF CURRENT BASIS BY LU DECOMPOSITION
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      INTEGER*2 MREG,HREG,VREG
      DIMENSION MREG(60),HREG(60),VREG(60)
      EQUIVALENCE (MREG(1),YTEMP(1)),(HREG(1),YTEMP(31)),(VREG(1),X(1))
C
C            SET PARAMETERS
C
      NETA = 0
      NLETA = 0
      NUETA = 0
      NELEM = 0
      NLELEM = 0
      NUELEM = 0
      NABOVE = 0
      LE(1) = 1
      LR1 = 1
      KR1 = 0
      LR4 = NROW + 1
      KR4 = NROW
C
C        PUT SLACKS AND ARTIFICIALS IN PART 4 AND REST IN PART 1
C
      DO 100 I = 1,NROW
      IF (JH(I) .GT. NROW) GO TO 50
      LR4 = LR4 - 1
      MREG(LR4) = JH(I)
      VREG(LR4) = JH(I)
      GO TO 90
   50 KR1 = KR1 + 1
      VREG(KR1) = JH(I)
   90 HREG(I) = -1
      JH(I) = 0
  100 CONTINUE
C
      KR3 = LR4 -1
```

```
915.            LR3 = LR4
916.      C
917.            DO 200 I = LR4,KR4
918.            IR = MREG(I)
919.            MREG(IR)= 0
920.            JH(IR) = IR
921.            KINBAS(IR) = IR
922.        200 CONTINUE
923.      C
924.      C               PULL OUT VECTORS BELOW BUMP AND GET ROW COUNTS
925.      C
926.            NBNONZ = KR4 - LR4 + 1
927.            IF (KR1 .EQ. 0) GO TO 1190
928.            J = LR1
929.        210 IV = VREG(J)
930.            LL = LA(IV)
931.            KK = LA(IV+1) -1
932.            IRCNT = 0
933.            DO 220 I = LL,KK
934.            NBNONZ = NBNONZ + 1
935.            IR = IA(I)
936.            IF (MREG(IR) .GE. 0) GO TO 220
937.            IRCNT = IRCNT + 1
938.            MREG(IR) = MREG(IR) - 1
939.            IRP = IR
940.        220 CONTINUE
941.            IF (IRCNT - 1) 230,250,300
942.        230 WRITE(6,8000)
943.       8000 FORMAT(16HOMATRIX SINGULAR )
944.            KINBAS(IV) = 0
945.            VREG(J) = VREG(KR1)
946.            KR1 = KR1 - 1
947.            IF (J .GT. KR1) GO TO 310
948.            GO TO 210
949.      C
950.        250 VREG(J) = VREG(KR1)
951.            KR1 = KR1 - 1
952.            LR3 = LR3 - 1
953.            VREG(LR3) = IV
954.            MREG(LR3) = IRP
955.            MREG(IRP) = 0
956.            JH(IRP) = IV
957.            KINBAS(IV) = IRP
958.            IF (J .GT. KR1) GO TO 310
959.            GO TO 210
960.        300 IF (J .GE. KR1) GO TO 310
961.            J = J+1
962.            GO TO 210
963.      C
964.      C                       PULL OUT REMAINING VECTORS ABOVE AND BELOW THE
965.      C                       BUMP AND ESTABLISH MERIT COUNTS OF COLUMNS
966.      C
967.        310 NVREM = 0
968.            IF(KR1 .EQ. 0) GO TO 1190
969.            J = LR1
970.        320 IV = VREG(J)
971.            LL = LA(IV)
972.            KK = LA(IV+1) - 1
973.            IRCNT = 0
974.            DO 800 I = LL,KK
975.            IR = IA(I)
```

```
      IF(HREG(IR) .NE. -2) GO TO 400
C
C            PIVOT ABOVE BUMP (PART OF L)
C
      NABOVE = NABOVE + 1
      IROWP = IR
      CALL UNPACK(IV)
      CALL WRETA
      NLETA = NETA
      JH(IR) = IV
      KINBAS(IV) = IR
      VREG(J) = VREG(KR1)
      KR1 = KR1 - 1
      NVREM = NVREM + 1
      HREG(IR) = IV
      GO TO 940
C
  400 IF (HREG(IR) .GE. 0) GO TO 800
      IRCNT = IRCNT + 1
      IRP = IR
  800 CONTINUE
C
      IF (IRCNT - 1) 810,900,1000
  810 WRITE(6,8000)
      KINBAS(IV) = 0
      VREG(J) = VREG(KR1)
      NVREM = NVREM + 1
      KR1 = KR1 - 1
      IF (J .GT. KR1) GO TO 1010
      GO TO 320
C
C                  PUT VECTOR BELOW BUMP
C
  900 VREG(J) = VREG(KR1)
      NVREM = NVREM + 1
      KR1 = KR1 - 1
      LR3 = LR3 - 1
      VREG(LR3) = IV
      MREG(LR3) = IRP
      HREG(IRP) = 0
      JH(IRP) = IV
      KINBAS(IV) = IRP
C
C           CHANGE ROW COUNTS
C
  940 DO 950 II = LL,KK
      IIR = IA(II)
      IF (HREG(IIR) .GE. 0) GO TO 950
      HREG(IIR) = HREG(IIR) + 1
  950 CONTINUE
      IF (J .GT. KR1) GO TO 1010
      GO TO 320
 1000 IF (J .GE. KR1) GO TO 1010
      J = J+1
      GO TO 320
 1010 IF(NVREM .GT. 0) GO TO 310
C
C              GET MERIT COUNTS
C
 1020 IF (KR1 .EQ. 0) GO TO 1190
      DO 1100 J = LR1,KR1
```

```
      IV = VREG(J)
      LL = LA(IV)
      KK = LA(IV+1) - 1
      IMCNT = 0
      DO 1050 I = LL,KK
      IR = IA(I)
      IF (HREG(IR) .GE. 0) GO TO 1050
      IMCNT = IMCNT - (HREG(IR) +1)
 1050 CONTINUE
      MREG(J) = IMCNT
 1100 CONTINUE
C
C                SORT COLUMNS INTO MERIT ORDER
C                USING SHELL SORT
C
      ISD = 1
 1106 IF (KR1 .LT. 2*ISD) GO TO 1108
      ISD = 2*ISD
      GO TO 1106
 1108 ISD = ISD - 1
C                        END OF INITIALIZATION
 1101 IF (ISD .LE. 0) GO TO 1107
      ISK = 1
 1102 ISJ = ISK
      ISL = ISK + ISD
      ISY = MREG(ISL)
      ISZ = VREG(ISL)
 1103 IF (ISY .LT. MREG(ISJ)) GO TO 1104
 1105 ISL = ISJ + ISD
      MREG(ISL) = ISY
      VREG(ISL) = ISZ
      ISK = ISK + 1
      IF ((ISK + ISD) .LE. KR1) GO TO 1102
      ISD = (ISD - 1) / 2
      GO TO 1101
 1104 ISL = ISJ + ISD
      MREG(ISL) = MREG(ISJ)
      VREG(ISL) = VREG(ISJ)
      ISJ = ISJ - ISD
      IF (ISJ .GT. 0) GO TO 1103
      GO TO 1105
 1107 CONTINUE
C                          END OF SORT ROUTINE
C
C          PUT OUT BELOW BUMP ETAS (PART OF U)
C
 1150 NSLCK = 0
      NBELOW = 0
      NELAST = NEMAX
      NTLAST = NTMAX
      LE(NTLAST + 1) = NELAST + 1
C
      LR = LR3
      IF (LR3 .GE. LR4) LR = LR4
      IF (LR .GT. KR4) GO TO 2050
      JK = KR4 + 1
      DO 2000 JJ= LR,KR4
      JK = JK - 1
      IV = VREG(JK)
      I = MREG(JK)
      NBELOW = NBELOW + 1
```

```
      IF (IV .GT. NROW) GO TO 1200
      NSLCK = NSLCK + 1
 1200 LL = LA(IV)
      KK = LA(IV+1) -1
      IF (KK .GT. LL) GO TO 1300
 1250 IF (ABS(A(LL) - 1.) .LE. ZTOLZE) GO TO 2000
C
 1300 NUETA = NUETA + 1
      DO 1400 J = LL,KK
      IR = IA(J)
      IF (IR .EQ. 1)GO TO 1390
      IE(NELAST) = IR
      E(NELAST) = A(J)
      NELAST = NELAST - 1
      NUELEM = NUELEM + 1
      GO TO 1400
 1390 EP = A(J)
 1400 CONTINUE
      IE(NELAST) = 1
      E(NELAST) = EP
      LE(NTLAST) = NELAST
      NELAST = NELAST - 1
      NTLAST = NTLAST - 1
      NUELEM = NUELEM + 1
 2000 CONTINUE
 2050 IF(KR1 .EQ. 0) GO TO 3500
C
C          DO L-U DECOMPOSITION OF BUMP
C
      DO 3000 J = LR1,KR1
      IV = VREG(J)
      CALL LAPACK(IV)
      CALL FTRAN(2)
      IROWP = 0
      IRCMIN = -999999
      DO 2100 I = 1,NROW
      IF (DABS(Y(I)) .LE. ZTOLPV) GO TO 2100
      IF (HREG(I) .GE.0) GO TO 2100
      IF (HREG(I) .LE. IRCMIN) GO TO 2100
      IRCMIN = HREG(I)
      IROWP = I
 2100 CONTINUE
      IF (IROWP .GT. 0) GO TO 2150
      WRITE(6,8000)
      KINBAS(IV) = 0
      GO TO 3000
C
 2150 INCR = HREG(IROWP) + 3
C
C            WRITE L AND U ETAS
C
      IF (J .EQ. KR1) GO TO 2160
      NELEM = NELEM + 1
      IE(NELEM) = IROWP
      E(NELEM) = Y(IROWP)
 2160 DO 2300 I = 1,NROW
      IF (I .EQ. IROWP) GO TO 2300
      IF(DABS(Y(I)) .LE. ZTOLZE) GO TO 2300
      IF (HREG(I) .GE. 0) GO TO 2200
C
C              L ETA ELEMENTS
```

```
1159.      C
1160.            NELEM = NELEM + 1
1161.            IE(NELEM) = I
1162.            E(NELEM) = Y(I)
1163.            GO TO 2300
1164.      C
1165.      C                 U ETA ELEMENTS
1166.      C
1167.       2200 IE(NELAST) = I
1168.            E(NELAST) = Y(I)
1169.            NELAST = NELAST - 1
1170.            NUELEM = NUELEM + 1
1171.       2300 CONTINUE
1172.      C
1173.            JH(IROWP) = IV
1174.            KINBAS(IV) = IROWP
1175.            NUETA = NUETA + 1
1176.            IE(NELAST) = IROWP
1177.            IF (J .NE. KR1) GO TO 2330
1178.            E(NELAST) = Y(IROWP)
1179.            GO TO 2340
1180.       2330 E(NELAST) = 1.
1181.            NETA = NETA + 1
1182.            LE(NETA+1) = NELEM + 1
1183.       2340 NUELEM = NUELEM + 1
1184.            LE(NTLAST) = NELAST
1185.            NELAST = NELAST - 1
1186.            NTLAST = NTLAST - 1
1187.      C
1188.      C           UPDATE ROW COUNTS
1189.      C
1190.            DO 2350 I = 1,NROW
1191.            IF (DABS(Y(I)) .LE. ZTOLZE) GO TO 2350
1192.            IF (HREG(I) .GE. 0) GO TO 2350
1193.            HREG(I) = HREG(1) - INCR
1194.            IF (HREG(I) .GE. 0) HREG(I) = -1
1195.       2350 CONTINUE
1196.            HREG(IROWP) = 0
1197.       3000 CONTINUE
1198.      C
1199.      C                 MERGE L AND U ETAS
1200.      C
1201.       3500 NLETA = NETA
1202.            NETA = NLETA + NUETA
1203.            NLELEM = NELEM
1204.            NELEM = NLELEM + NUELEM
1205.            IF (NUELEM .EQ. 0) GO TO 3550
1206.            CALL SHFTE
1207.      C
1208.      C                   INSERT SLACKS FOR DELETED COLUMNS
1209.      C
1210.       3550 DO 3600 I = 1,NROW
1211.            IF (JH(I) .NE. 0) GO TO 3600
1212.            JH(I) = I
1213.            IROWP = I
1214.            CALL UNPACK(I)
1215.            CALL FTRAN(1)
1216.            CALL WRETA
1217.       3600 CONTINUE
1218.      C
1219.      C                 UPDATE X
```

```
      C
            CALL SHIFTR(1,3)
            DO 9000 J=1,NCOL
            IF (KINBAS(J)) 8800,8700,9000
 8800 DE = XUB(J)
            GO TO 8750
 8700 DE = XLB(J)
 8750 LL = LA(J)
            KK = LA(J+1) - 1
            DO 8800 I=LL,KK
            IR = IA(I)
 8800 Y(IR) = Y(IR) - A(I)*DE
 9000 CONTINUE
            CALL FTRAN(1)
            CALL SHIFTR(3,2)
            RETURN
            END
            SUBROUTINE UNPACK(IV)
      C
      C        EXPANDS COMPRESSED MATRIX COLUM
      C********DESCRIPTION OF PARAMETERS********
      C        IV = PARAMETER INDEXING COLUMN TO BE EXPANDED
      C*************************************************
      C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            DOUBLE PRECISION E(1000)
            REAL A(500)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      C
            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVERQ,ITREQ,ITSINV,IFREZ,
           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      C
            DO 100 I = 1,NROW
            Y(I) = 0.
  100 CONTINUE
      C
            LL = LA(IV)
            KK = LA(IV+1) - 1
            DO 200 I = LL,KK
            IR = IA(I)
            Y(IR) = A(I)
  200 CONTINUE
      C
            RETURN
            END
            SUBROUTINE SHFTE
      C
      C        SUBROUTINE FOR INVERT
      C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
```

```
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
C               SHIFT IE AND E OF U ELEMENTS
C
      NF = NEMAX - NUELEM + 1
      INCR = 0
      DO 1000 I = NF,NEMAX
      INCR = INCR + 1
      IE(NLELEM + INCR) = IE(I)
      E(NLELEM + INCR) = E(I)
 1000 CONTINUE
C
      IDIF = NEMAX - NLELEM - NUELEM
      NF = NTMAX - NUETA + 1
      INCR = 0
      DO 2000 I = NF,NTMAX
      INCR = INCR + 1
      LE(NLETA + INCR) = LE(I) - IDIF
 2000 CONTINUE
      LE(NETA+1) = NELEM + 1
      RETURN
      END
      SUBROUTINE UPBETA
C
C        UPDATES RIGHT-HAND SIDES TO REFLECT NEW BASIS RESULTING FROM
C        CURRENT SIMPLEX PIVOT
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
C
      COMMON/BLOCK2/ DPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
     2               IVBND(500),IVID(500),ICBND(500)
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAZ,
```

```
1342.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
1343.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
1344.     C
1345.           DO 1000 I=1,NROW
1346.      1000 X(I) = X(I) - Y(I)*DP
1347.           IF (NPIVOT .EQ. 1) GO TO 2000
1348.           KINBAS(JCOLP) = -(KINBAS(JCOLP) + 1)
1349.           IVOUT = JCOLP
1350.           RETURN
1351.      2000 X(IROWP) = DE
1352.           IVOUT = JH(IROWP)
1353.           KINBAS(JCOLP) = IROWP
1354.           KINBAS(IVOUT) = IPTYPE
1355.           JH(IROWP) = JCOLP
1356.           RETURN
1357.           END
1358.           SUBROUTINE NORMAL
1359.     C
1360.     C         SERVES AS MASTER PROGRAM FOR LINEAR PROGRAMMING COMPONENT
1361.     C         (REVISED PRIMAL-SIMPLEX METHOD).
1362.     C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
1363.     C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
1364.     C
1365.           IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
1366.          1          INTEGER*4 (I-N,Q)
1367.           INTEGER*2 JH,KINBAS,LA,LE,IA,IE
1368.           INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
1369.           DOUBLE PRECISION E(1000)
1370.           REAL A(500)
1371.     C
1372.           COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
1373.          1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
1374.          2              QE,QH,QL,QO,QR,QM,QG,NTMAX
1375.     C
1376.           COMMON/BLOCK2/ DPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
1377.          1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
1378.          2               IVBND(500),IVID(500),ICBND(500)
1379.     C
1380.           COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
1381.          1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
1382.          2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVERQ,ITRERQ,ITSINV,IFEEZ,
1383.          3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
1384.          4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
1385.     C
1386.           IF (ITSINV .LT. INVERQ) GO TO 1500
1387.      1000 CALL INVERT
1388.           ITSINV = 0
1389.     C
1390.     C               SIMPLEX CYCLE
1391.     C
1392.      1500 CALL FORMC
1393.           CALL BTRAN
1394.           CALL PRICE
1395.           IF (JCOLP .GT. 0) GO TO 3000
1396.           IF (MSTAT .EQ. 0)     ) GO TO 2000
1397.           MSTAT = QBL
1398.           GO TO 6000
1399.      2000 MSTAT = QN
1400.           GO TO 6000
1401.      3000 IVIN = JCOLP
1402.           CALL UNPACK(JCOLP)
```

```
1403.            CALL FTRAN(1)
1404.            CALL CHUZR
1405.            CALL UPBETA
1406.            ITCNT = ITCNT + 1
1407.            ITSINV = ITSINV + 1
1408.            IF (NPIVOT .EQ. 0) GO TO 4010
1409.            IF (NELEM .GT. 5000) GO TO 1000
1410.            CALL WBETA
1411.       4010 IF (ITSINV .GE. INVFRQ) GO TO 1000
1412.            IF (ITCNT .GE. ITRFRQ) GO TO 6000
1413.            GO TO 1500
1414.      C
1415.       6000 RETURN
1416.            END
1417.            SUBROUTINE BANDB(INITBD)
1418.      C
1419.      C        MASTER PROGRAM FOR BRANCH-AND-BOUND INTEGER PROGRAMMING ROUTINE
1420.      C        ALSO SERVES AS MASTER PROGRAM FOR REOPTIMIZATION VIA DUAL-
1421.      C        SIMPLEX METHOD AFTER A FORWARD BRANCH.
1422.      C********DESCRIPTION OF PARAMETERS********
1423.      C        INITBD = INITIAL LOWER BOUND ON MAXIMAL OBJECTIVE VALUE (INPUT)
1424.      C**************************************
1425.      C
1426.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
1427.           1          INTEGER*4 (I-N,Q)
1428.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
1429.            INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
1430.            DOUBLE PRECISION E(1000)
1431.            REAL A(500)
1432.            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
1433.           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
1434.           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
1435.            COMMON/BLOCK2/ DPPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
1436.           1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
1437.           2               IVBND(500),IVID(500),ICBND(500)
1438.            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
1439.           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
1440.           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAS,
1441.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
1442.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
1443.            LISTL = 0
1444.            INCVAL = INITBD
1445.      C        TEST FOR FATHOMING
1446.         100 CALL TESTX
1447.            IF (MSTAT .EQ. QBL) GO TO 200
1448.      C        CURRENT NODE FATHOMED; BACKTRACK TO LAST PROMISING NODE ON LIST
1449.         150 CALL BKTRAK
1450.      C        IF LIST IS EMPTY, RETURN TO MAIN (COMPUTATIONS COMPLETED)
1451.            IF (LISTL .EQ. 0) RETURN
1452.      C        USE PRIMAL-SIMPLEX METHOD FOR REOPTIMIZATION AT NEW NODE
1453.            CALL NORMAL
1454.            GO TO 100
1455.      C        CURRENT NODE NOT FATHOMED; COMPUTE PENALTIES
1456.      C        BRANCHING AT CURRENT NODE IS DONE FROM SUBROUTINE PENLTS.
1457.         200 CALL PENLTS
1458.            IF (IDIR) 400,150,400
1459.      C
1460.      C        REINVERT CURRENT BASIS
1461.       1000 CALL INVERT
1462.            ITSINV = 0
1463.      C
```

```
1464.      C              DUAL SIMPLEX CYCLE
1465.      C
1466.      C         CHOOSE PIVOT ROW IROWP
1467.        300 CALL DCHUZR
1468.            IF (IROWP .GT. 0) GO TO 400
1469.            MSTAT = QBL
1470.            GO TO 100
1471.      C         CHOOSE PIVOT COLUMN JCOLP
1472.        400 CALL DCHUZC
1473.            IF (JCOLP .EQ. 0) GO TO 150
1474.            IVIN = JCOLP
1475.      C         UPDATE RIGHT-HAND SIDES TO REFLECT NEW BASIS RESULTING FROM
1476.      C         CURRENT SIMPLEX PIVOT
1477.            CALL UPBETA
1478.            ITCNT = ITCNT + 1
1479.            ITSINV = ITSINV + 1
1480.      C         DECIDE WHETHER TO REINVERT CURRENT BASIS
1481.            IF ((NELEM .GT. 5600) .OR. (ITSINV .GE. INVFRQ)) GO TO 1000
1482.      C         REINVERSION NOT NECESSARY YET; WRITE OUT NEW ETA-VECTOR FOR
1483.      C         CURRENT SIMPLEX PIVOT
1484.            CALL WRETA
1485.            GO TO 300
1486.            END
1487.            SUBROUTINE DCHUZR
1488.      C
1489.      C         SELECTS PIVOT ROW IROWP FOR CURRENT DUAL-SIMPLEX ITERATION.
1490.      C         SETS IROWP = 0 IF CURRENT BASIS IS OPTIMAL.  OTHERWISE, IROWP
1491.      C         IS CHOSEN TO BE THE ROW WITH GREATEST PRIMAL-INFEASIBILITY
1492.      C
1493.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
1494.           1          INTEGER*4 (I-N,Q)
1495.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
1496.            INTEGER*2 IPART,INCUMB,IVBND,IVID,IQBND
1497.            DOUBLE PRECISION E(1000)
1498.            REAL A(500)
1499.            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
1500.           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
1501.           2              QE,QH,QL,QQ,QR,QM,QG,NTMAX
1502.            COMMON/BLOCK2/ DEPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
1503.           1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
1504.           2               IVBND(500),IVID(500),IQBND(500)
1505.            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
1506.           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
1507.           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAZ,
1508.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
1509.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
1510.      C
1511.      C              CHOOSE ROW WITH GREATEST INFEASIBILITY
1512.      C
1513.            IROWP = 0
1514.            DP = -1.E10
1515.            DO 1000 I=1,NROW
1516.            IF (I .EQ. IOBJ) GO TO 1000
1517.            ICOL = JH(I)
1518.            IF (X(I) .LT. (XLB(ICOL) - ZTOLZE)) GO TO 100
1519.            IF (X(I) .GT. (XUB(ICOL) + ZTOLZE)) GO TO 200
1520.            GO TO 1000
1521.      C
1522.      C         BASIC VARIABLE ON ROW I FALLS BELOW ITS LOWER BOUND
1523.        100 DE = XLB(ICOL) - X(I)
1524.            IF (DE .LE. DP) GO TO 1000
```

```
1525.            IPTYPE = 0
1526.            GO TO 250
1527.      C
1528.      C        BASIC VARIABLE ON ROW I EXCEEDS ITS UPPER BOUND
1529.        200 DE = X(I) - XUB(ICOL)
1530.            IF (DE .LE. DP) GO TO 1000
1531.            IPTYPE = -1
1532.      C
1533.        250 IROWP = I
1534.       1000 CONTINUE
1535.            RETURN
1536.            END
1537.            SUBROUTINE DCHUZC
1538.      C
1539.      C        SELECTS PIVOT COLUMN JCOLP FOR CURRENT DUAL-SIMPLEX ITERATION.
1540.      C        SETS JCOLP = 0 IF LP-PROBLEM AT CURRENT NODE IS INFEASIBLE.
1541.      C        OTHERWISE, CHOOSES JCOLP TO MAINTAIN PRIMAL-OPTIMALITY
1542.      C
1543.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
1544.           1          INTEGER*4 (I-N,Q)
1545.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
1546.            INTEGER*2 IPART,INCUMB,IVBND,IVID,IOBND
1547.            DOUBLE PRECISION E(1000)
1548.            REAL A(500)
1549.            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
1550.           1              QEO,QBL,QA,QPL,QM1,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
1551.           2              QE,QH,QL,QO,QR,QM,QG,NTMAX
1552.            COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
1553.           1               NPIVOT,IPTYPE,KCOST,IFEAS,IPART(122),INCUMB(122),
1554.           2               IVBND(500),IVID(500),IOBND(500)
1555.            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
1556.           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
1557.           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
1558.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
1559.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
1560.      C
1561.            JCOLP = 0
1562.            IF (IPTYPE .EQ. -1) GO TO 1000
1563.      C
1564.      C              LEAVING VARIABLE FALLS BELOW ITS LOWER BOUND
1565.      C
1566.            DP = -1.E10
1567.            DO 500 J=1,NCOL
1568.            IF (KINBAS(J) .GT. 0) GO TO 500
1569.            IF ((XUB(J) - XLB(J)) .LE. ZTOLZE) GO TO 500
1570.            CALL UNPACK(J)
1571.            CALL FTRAN(1)
1572.            IF (KINBAS(J) .EQ. -1) GO TO 200
1573.            IF (Y(IROWP) + ZTOLPV) 225,225,500
1574.        200 IF (Y(IROWP) - ZTOLPV) 500,225,225
1575.      C
1576.        225 DE = Y(IOBJ)/Y(IROWP)
1577.            IF (DE - DP) 500,500,250
1578.        250 JCOLP = J
1579.            DP = DE
1580.        500 CONTINUE
1581.      C
1582.            IF (JCOLP .EQ. 0) RETURN
1583.            CALL UNPACK(JCOLP)
1584.            CALL FTRAN(1)
1585.            ICOL = JH(IROWP)
```

```
      DP = (X(IROWP) - XLB(ICOL))/Y(IROWP)
      GO TO 2000
C
C              LEAVING VARIABLE EXCEEDS ITS UPPER BOUND
C
 1000 DP = 1.E10
      DO 1500 J=1,NCOL
      IF (KINBAS(J) .GT. 0) GO TO 1500
      IF ((XUB(J) - XLB(J)) .LE. ZTOLZE) GO TO 1500
      CALL UNPACK(J)
      CALL FTRAN(1)
      IF (KINBAS(J) .EQ. -1) GO TO 1200
      IF (Y(IROWP) - ZTOLPV) 1500,1225,1225
 1200 IF (Y(IROWP) + ZTOLPV) 1225,1225,1500
C
 1225 DE = Y(IOBJ)/Y(IROWP)
      IF (DE - DP) 1250,1500,1500
 1250 JCOLP = J
      DP = DE
 1500 CONTINUE
C
      IF (JCOLP .EQ. 0) RETURN
      CALL UNPACK(JCOLP)
      CALL FTRAN(1)
      ICOL = JH(IROWP)
      DP = (X(IROWP) - XUB(ICOL))/Y(IROWP)
C
 2000 IF (KINBAS(JCOLP) .EQ. 0) DE = DP + XLB(JCOLP)
      IF (KINBAS(JCOLP) .EQ. -1) DE = DP + XUB(JCOLP)
      NPIVOT = 1
      RETURN
      END
      SUBROUTINE TESTX
C
C        TESTS LP-OPTIMAL SOLUTION AT CURRENT NODE FOR FATHOMING.
C        FATHOMING OCCURS IF
C        (1) LP PROBLEM AT CURRENT NODE IS INFEASIBLE (MSTAT = ON); OR
C        (2) LP-OPTIMAL OBJECTIVE VALUE (IVAL) .LE. OBJECTIVE VALUE OF
C            CURRENT INCUMBENT SOLUTION (INCVAL); OR
C        (3) LP-OPTIMAL SOLUTION SATISFIES INTEGER RESTRICTIONS.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,IOBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DPPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
     2               IVBND(500),IVID(500),IOBND(500)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      DP = X(IOBJ) + ZTOLZE
      IVAL = IDINT(DP)
      IF (DP .LT. 0.) IVAL = IVAL - 1
```

```
      IF ((MSTAT .EQ. QN) .OR. (IVAL .LE. INCVAL)) GO TO 2000
C
C               COMPUTE INTEGER AND FRACTIONAL PARTS OF EACH BASIC VAR.
C
      DO 100 I=1,NROW
      IPART(I) = IDINT(X(I) + ZTOLZE)
      NTEMP(I) = IPART(I)
  100 DFPART(I) = X(I) - FLOAT(NTEMP(I))
C
C               CHECK FOR ALL-INTEGER SOLUTION
C
      DO 200 I=1,NROW
      IF (JH(I) .LE. NROW) GO TO 200
      IF (DFPART(I) .GE. ZTOLZE) RETURN
  200 CONTINUE
C
C               SOLUTION ALL-INTEGER: INSTALL AS NEW INCUMBENT
C
      IFEAS = 1
      INCVAL = IVAL
      DO 1000 J=1,NCOL
      IF (KINBAS(J))600,700,1000
  600 INCUMB(J) = IDINT(XUB(J))
      GO TO 1000
  700 INCUMB(J) = IDINT(XLB(J))
 1000 CONTINUE
C
      DO 1100 I=1,NROW
      ICOL = JH(I)
      INCUMB(ICOL) = IPART(I)
 1100 CONTINUE
C
C               CURRENT PROBLEM NO LONGER OF INTEREST
C
 2000 MSTAT = QI
      RETURN
      END
      SUBROUTINE PENLTS
C
C        COMPUTES TOMLIN'S IMPROVED UP- AND DOWN-PENALTIES AND THE
C        GOMORY PENALTY FOR EACH NONINTEGER BASIC VARIABLE.  ALSO CHECKS
C        FOR FORCED BRANCHES ON BOTH BASIC AND NONBASIC VARIABLES.  IN
C        THE ABSENCE OF FORCED BRANCHES ON BASIC VARIABLES, THE
C        BRANCHING VARIABLE IS CHOSEN TO BE THE ONE WITH LARGEST
C        ASSOCIATED UP- OR DOWN-PENALTY.  THE FORWARD BRANCH IS MADE IN
C        THE DIRECTION OPPOSITE TO THE MAXIMUM PENALTY.  THE NODE
C        CORRESPONDING TO THE BRANCH IN THE SAME DIRECTION AS THE
C        MAXIMUM PENALTY IS ADDED TO THE LIST, TO BE EXAMINED LATER.
C        THE BRANCHING PROCESS ITSELF IS CARRIED OUT IN SUBROUTINE
C        BRANCH, WHICH IS CALLED FROM SUBROUTINE PENLTS.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
      REAL PU(60),PD(60),PG(60)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QF,QL,QO,QR,QM,QG,NTMAX
```

```
      COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NP(VOT,IPTYPE,KCOST,IFEAS,LPART(122),INCUMB(122),
     2               IVBND(500),IVID(500),IOBND(500)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      DO 10 I=1,NROW
      IF (DFPART(I) .LT. ZTOLZE) GO TO 5
      PU(I) = 1.E6
      PD(I) = 1.E6
      PG(I) = 1.E6
      GO TO 10
    5 PU(I) = 0.
      PD(I) = 0.
      PG(I) = 0.
   10 CONTINUE
C
      DO 1000 J=1,NCOL
      IF (KINBAS(J) .GT. 0) GO TO 1000
      IF ((XUB(J) - XLB(J)) .LE. ZTOLZE) GO TO 1000
      CALL UNPACK(J)
      CALL FTRAN(I)
      IF (KINBAS(J) .EQ. 0) GO TO 30
      DO 20 I=1,NROW
   20 Y(I) = -Y(I)
C
C                 CHECK FOR FORCED BRANCH ON XJ
   30 IF (J .LE. NROW) GO TO 50
      DP = X(IOBJ) - Y(IOBJ) + ZTOLZE
      IVAL = IDINT(DP)
      IF (DP .LT. 0.) IVAL = IVAL - 1
      IF (IVAL .GT. INCVAL) GO TO 50
      IDIR = 2*KINBAS(J) + 1
      IF (IDIR .EQ. -1) REVBND = SNGL(XUB(J))
      IF (IDIR .EQ. 1) REVBND = SNGL(XLB(J))
      ICOL = J
      CALL BRANCH
      GO TO 1000
C
   50 DO 500 I=1,NROW
      IF (IP(I) .LE. NROW) GO TO 500
      IF (DFPART(I) .LT. ZTOLZE) GO TO 500
C
C                 COMPUTE UP PENALTY FOR X(I), XJ
  100 IF (Y(I) .GT. -ZTOLPV) GO TO 200
      DE = Y(IOBJ)*(DFPART(I) - 1.)/Y(I)
      IF (DE .LT. Y(IOBJ)) DE = Y(IOBJ)
      IF (DE .LT. PU(I)) PU(I) = DE
      GO TO 300
C
C                 COMPUTE DOWN PENALTY FOR X(I), XJ
  200 IF (Y(I) .LT. ZTOLPV) GO TO 300
      DE = Y(IOBJ)*DFPART(I)/Y(I)
      IF (DE .LT. Y(IOBJ)) DE = Y(IOBJ)
      IF (DE .LT. PD(I)) PD(I) = DE
C
C                 COMPUTE GOMORY PENALTY FOR X(I), XJ
  300 DP = DABS(Y(I))
      NTEMP(I) = IDINT(DP)
```

```
1769.            DP = DP - FLOAT(NTEMP(1))
1770.            IF ((DP .LE. ZTOLZE) .OR. (DP .GE. 1.-ZTOLZE)) GO TO 500
1771.            IF (Y(I) .LT. 0.) DP = 1. - DP
1772.            IF (DP .GT. DFPART(I)) GO TO 310
1773.            DE = Y(IOBJ)*DFPART(I)/DP
1774.            GO TO 320
1775.        310 DE = Y(IOBJ)*(1. - DFPART(I))/(1. - DP)
1776.        320 IF (DE .LT. PG(I)) PG(I) = DE
1777.      C
1778.        500 CONTINUE
1779.       1000 CONTINUE
1780.      C
1781.      C                 COMPUTE LARGEST GOMORY PENALTY AND TEST FOR FATHOMING
1782.            PEN = 0.
1783.            DO 2000 I=1,NROW
1784.            IF (JH(I) .LE. NROW) GO TO 2000
1785.            IF (PG(I) .GT. PEN) PEN = PG(I)
1786.       2000 CONTINUE
1787.            DP = X(IOBJ) - PEN + ZTOLZE
1788.            IVAL = IDINT(DP)
1789.            IF (DP .LT. 0.) IVAL = IVAL - 1
1790.            IF (IVAL .GT. INCVAL) GO TO 3000
1791.            IDIR = 0
1792.            RETURN
1793.      C
1794.      C                 PROBLEM NOT FATHOMED: CHECK FOR FORCED BRANCHES ON X(I)
1795.       3000 NTEMP(2) = 0
1796.            DO 3900 I=1,NROW
1797.            IF (JH(I) .LE. NROW) GO TO 3900
1798.            IF (PG(I) .GT. PD(I)) GO TO 3600
1799.            DP = X(IOBJ) - PD(I) + ZTOLZE
1800.            NTEMP(1) = IDINT(DP)
1801.            IF (DP .LT. 0.) NTEMP(1) = NTEMP(1) - 1
1802.            IF (NTEMP(1) .GT. INCVAL) GO TO 3900
1803.      C                 FORCED BRANCH UP ON X(I)
1804.            IVAL = NTEMP(1)
1805.            IDIR = -1
1806.            NTEMP(1) = IPART(I) + 1
1807.            GO TO 3700
1808.      C
1809.       3600 DP = X(IOBJ) - PU(I) + ZTOLZE
1810.            NTEMP(1) = IDINT(DP)
1811.            IF (DP .LT. 0.) NTEMP(1) = NTEMP(1) - 1
1812.            IF (NTEMP(1) .GT. INCVAL) GO TO 3900
1813.      C                 FORCED BRANCH DOWN ON X(I)
1814.            IVAL = NTEMP(1)
1815.            IDIR = 1
1816.            NTEMP(1) = IPART(I)
1817.       3700 IROWP = I
1818.            ICOL = JH(IROWP)
1819.            REVBND = FLOAT(NTEMP(1))
1820.            NTEMP(2) = 1
1821.            CALL BRANCH
1822.       3900 CONTINUE
1823.            IF (NTEMP(2) .GT. 0) GO TO 5000
1824.      C
1825.      C                 NO FORCED BRANCHES: CHOOSE BRANCHING VAR. AND DIRECTION
1826.            PEN = 0.
1827.            IROWP = 0
1828.      C        DETERMINE BASIC VAR. X(IROWP) WITH MAX. UP- OR DOWN-PENALTY
1829.            DO 4900 I=1,NROW
```

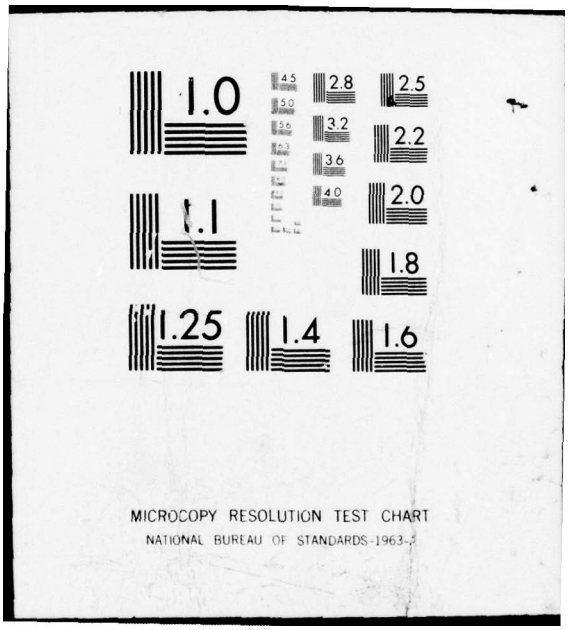
AD-A038 322 STANFORD UNIV CALIF SYSTEMS OPTIMIZATION LAB F/G 9/2
COMPUTER PROGRAMS FOR DECOMPOSITION IN INTEGER PROGRAMMING.(U)
SEP 76 G A KOCHMAN N00014-76-C-0418

UNCLASSIFIED SOL-76-20 ARO-12215.13-M NL

2 OF 2
AD A038 322

END

DATE FILMED
5-77


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
      IF (JH(I) .LE. NROW) GO TO 4900
      IF (PU(I) .GT. PD(I)) GO TO 4600
      IF (PD(I) .LE. PEN) GO TO 4900
      PEN = PD(I)
      IROWP = I
      IDIR = -1
      NTEMP(1) = IPART(I) + 1
      REVBND = FLOAT(NTEMP(1))
      GO TO 4900
 4600 IF (PU(I) .LE. PEN) GO TO 4900
      PEN = PU(I)
      IROWP = I
      IDIR = 1
      NTEMP(1) = IPART(I)
      REVBND = FLOAT(NTEMP(1))
 4900 CONTINUE
      IF (IROWP .GT. 0) GO TO 4950
C        EACH UP- AND DOWN-PENALTY .EQ. 0. (DUAL-DEGENERACY).  CHOOSE
C        ANY NONINTEGER BASIC VAR. AS BRANCHING VAR.
      DO 4910 IROWP=1,NROW
      IF (JH(IROWP) .LE. NROW) GO TO 4910
      IF (DFPART(IROWP) .GE. ZTOLZE) GO TO 4920
 4910 CONTINUE
 4920 PEN = PU(IROWP)
      IDIR = 1
      NTEMP(1) = IPART(IROWP)
      REVBND = FLOAT(NTEMP(1))
 4950 ICOL = JH(IROWP)
      DP = X(IOBJ) - PEN + ZTOLZE
      NTEMP(1) = IDINT(DP)
      IF (DP .LT. 0.) NTEMP(1) = NTEMP(1) - 1
      IF (IVAL .GT. NTEMP(1)) IVAL = NTEMP(1)
C        BRANCH ON CHOSEN VAR.
      CALL BRANCH
 5000 IF (IDIR .EQ. -1) IPTYPE = 0
      IF (IDIR .EQ. 1) IPTYPE = -1
      RETURN
      END
      SUBROUTINE BRANCH
C
C        BRANCHES ON VARIABLE X(ICOL) AS DETERMINED IN SUBROUTINE PENLTS
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,IOBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEQ,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
     2               IVBND(500),IVID(500),IOBND(500)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
C          ICOL INDEXES BRANCHING VARIABLE CHOSEN
```

```
C              IDIR INDICATES BRANCHING DIRECTION CHOSEN
C
C              ADD OPPOSITE DIRECTION TO LIST
      LISTL = LISTL + 1
      IF (IDIR .EQ. -1) IVBND(LISTL) = IDINT(XLB(ICOL) + ZTOLZE)
      IF (IDIR .EQ. 1) IVBND(LISTL) = IDINT(XUB(ICOL) + ZTOLZE)
      IVID(LISTL) = IDIR*ICOL
      IOBND(LISTL) = IVAL
C
C              REVISE BOUNDS ON BRANCHING VARIABLE FOR FORWARD DIRECTION
      IF (IDIR .EQ. -1) XLB(ICOL) = DBLE(REVBND)
      IF (IDIR .EQ. 1) XUB(ICOL) = DBLE(REVBND)
      RETURN
      END
      SUBROUTINE BKTRAK
C
C        BACKTRACKS TO SELECT A PROMISING (UNFATHOMED) NODE FROM THE
C        LIST OF STORED NODES.  LAST-IN-FIRST-OUT (LIFO) SELECTION RULE
C        IS EMPLOYED.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QF,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVCT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
     2               IVBND(500),IVID(500),IOBND(500)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      NTEMP(3) = 0
C        IF LIST IS EMPTY, RETURN (COMPUTATIONS COMPLETED)
   50 IF (LISTL .EQ. 0) RETURN
      IF (ICBND(LISTL) .LE. INCVAL) GO TO 2000
C
C        GET NEXT NODE FROM LIST
      ICOL = IVID(LISTL)
      IF (ICOL .LT. 0) GO TO 100
C
      NTEMP(1) = IDINT(XLB(ICOL) + ZTOLZE)
      NTEMP(2) = IVBND(LISTL)
      XLB(ICOL) = XUB(ICOL) + 1.
      XUB(ICOL) = FLOAT(NTEMP(2))
      IF (KINBAS(ICOL) .GT. 0) GO TO 1000
      KINBAS(ICOL) = 0
      NTEMP(3) = 1
      GO TO 1000
C
  100 ICOL = -ICOL
      NTEMP(1) = IDINT(XUB(ICOL) + ZTOLZE)
      NTEMP(2) = IVBND(LISTL)
      XUB(ICOL) = XLB(ICOL) - 1.
      XLB(ICOL) = FLOAT(NTEMP(2))
      IF (KINBAS(ICOL) .GT. 0) GO TO 1000
```

```
      KINBAS(ICOL) = -1
      NTEMP(3) = 1
C
 1000 IVID(LISTL) = -IVID(LISTL)
      IVBND(LISTL) = NTEMP(1)
      IOBND(LISTL) = -10000
C
C             UPDATE X
      IF (NTEMP(3) .EQ. 0) RETURN
      CALL SHIFTR(1,3)
      DO 9000 J=1,NCOL
      IF (KINBAS(J)) 8600,8700,9000
 8600 DE = XUB(J)
      GO TO 8750
 8700 DE = XLB(J)
 8750 LL = LA(J)
      KK = LA(J+1) - 1
      DO 8800 I=LL,KK
      IR = IA(I)
 8800 Y(IR) = Y(IR) - A(I)*DE
 9000 CONTINUE
      CALL FTRAN(1)
      CALL SHIFTR(3,2)
      RETURN
C
C        NODE FATHOMED:  UPDATE VAR. BOUNDS AND BACKTRACK AGAIN
 2000 ICOL = IVID(LISTL)
      IF (ICOL .LT. 0) GO TO 2100
C
      NTEMP(1) = IVBND(LISTL)
      IF (KINBAS(ICOL)) 2010,2050,2050
 2010 NTEMP(3) = 1
      DP = XUB(ICOL) - XLB(ICOL)
      DY = FLOAT(NTEMP(1)) - XUB(ICOL)
      IF (DP .LT. DY) KINBAS(ICOL) = 0
 2050 XUB(ICOL) = FLOAT(NTEMP(1))
      GO TO 3000
C
 2100 ICOL = -ICOL
      NTEMP(1) = IVBND(LISTL)
      IF (KINBAS(ICOL)) 2150,2110,2150
 2110 NTEMP(3) = 1
      DY = XLB(ICOL) - FLOAT(NTEMP(1))
      DP = XUB(ICOL) - XLB(ICOL)
      IF (DP .LT. DY) KINBAS(ICOL) = -1
 2150 XLB(ICOL) = FLOAT(NTEMP(1))
C
 3000 LISTL = LISTL - 1
      GO TO 50
C
      END
      SUBROUTINE ALOCTE(IPAR,IRHS)
C
C        SOLVES MASTER PROBLEM (CONCAVE-SEPARABLE INTEGER PROGRAM);
C        REALLOCATES LINKING RESOURCE AMONG UNFIXED SUBPROBLEMS
C********DESCRIPTION OF PARAMETERS********
C        IPAR = PARAMETER INDICATING WHETHER LAST BRANCH WAS IN FORWARD
C               (IPAR = 1) OR BACKTRACKING DIRECTION (IPAR = 0).
C        IRHS = AMOUNT OF LINKING RESOURCE LEFT (USED WHEN IPAR = 1)
C*************************************************
C
```

```
C         WHEN IPAR = 1, MASTER PROBLEM RECPTIMIZED BY REALLOCATING IRHS,
C         STARTING FROM PREVIOUSLY OPTIMAL SOLUTION TO MASTER PROBLEM.
C         WHEN IPAR = 0, MASTER PROBLEM RESOLVED FROM SCRATCH (I.E., ALL
C         B(J) SET TO LOWER BOUNDS IBLB(J) INITIALLY.)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
      INTEGER*2 IZSTAR,IBID,IBBND,IZBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPE,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVCT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
     2               IVBND(500),IVID(500),ICBND(500)
      COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
     1IZBAR,IDOM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
     2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
     3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),USUM,OPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C         TEST WHETHER LAST BRANCH WAS IN FORWARD OR BACKTRACKING DIREC-
C         TION
      IF (IPAR .EQ. 0) GO TO 50
C         LAST BRANCH WAS IN FORWARD DIRECTION:  INITIALIZE B(J) TO
C         PREVIOUSLY OPTIMAL SOLUTION
      DO 10 J=1,NSUBS
      LAST = NSOL(J) + 1
      IF (JFIX(J) .GT. 0) LAST = JFIX(J)
   10 NTEMP(J) = IBSTAR(LAST,J)
      GO TO 200
C
C         LAST BRANCH WAS IN BACKTRACKING DIRECTION
C              SET ALL B(J) = IBLB(J) AND IRHS = AMT OF RESOURCE LEFT
   50 INCVAL = -10000
      IRHS = IB
      DO 90 J=1,NSUBS
   90 IRHS = IRHS - IBLB(J)
      IF (IRHS .LT. 0) RETURN
C
      DO 130 J=1,NSUBS
      IF (JFIX(J) .GT. 0) GO TO 130
      ZJBAR(J) = ZNAUT(J)
      LAST = NLAMDA(J) + 1
      DO 110 K=2,LAST
      KM1 = K - 1
      IF (IBP(K,J) .GT. IBLB(J)) GO TO 120
  110 ZJBAR(J) = ZJBAR(J) + RLAMDA(KM1,J)*(IBP(K,J) - IBP(KM1,J))
      KSEG(J) = LAST
      GO TO 130
  120 KSEG(J) = KM1
      ZJBAR(J) = ZJBAR(J) + RLAMDA(KM1,J)*(IBLB(J) - IBP(KM1,J))
  130 NTEMP(J) = IBLB(J)
C
C              ALLOCATE:  GET LARGEST UP-SLOPE
```

```
2074.        200 RLMAX = -1.
2075.            DO 220 J=1,NSUBS
2076.            IF (JFIX(J) .GT. 0) GO TO 220
2077.            IF (NTEMP(J) .EQ. IBUB(J)) GO TO 220
2078.            K = KSEG(J)
2079.            IF (RLMAX .GE. RLAMDA(K,J)) GO TO 220
2080.            RLMAX = RLAMDA(K,J)
2081.            JSTORE = J
2082.        220 CONTINUE
2083.            IF (RLMAX .LE. -0.5) GO TO 1100
2084.      C              INCREMENT CORRESPONDING B(J)
2085.            K = KSEG(JSTORE)
2086.            KP1 = K + 1
2087.            IDELTA = IBP(KP1,JSTORE) - NTEMP(JSTORE)
2088.            IF (IBP(KP1,JSTORE) .GT. IBUB(JSTORE))
2089.           1                        IDELTA = IBUB(JSTORE) - NTEMP(JSTORE)
2090.            IRHS = IRHS - IDELTA
2091.            NTEMP(JSTORE) = NTEMP(JSTORE) + IDELTA
2092.            ZJBAR(JSTORE) = ZJBAR(JSTORE) + RLAMDA(K,JSTORE)*IDELTA
2093.            IF(NTEMP(JSTORE).EQ.IBP(KP1,JSTORE))KSEG(JSTORE)=KSEG(JSTORE)+1
2094.            IF (IRHS .LE. 0) GO TO 1000
2095.            GO TO 200
2096.      C
2097.      C              END OF ALLOCATION
2098.       1000 NTEMP(JSTORE) = NTEMP(JSTORE) + IRHS
2099.            ZJBAR(JSTORE) = ZJBAR(JSTORE) + RLAMDA(K,JSTORE)*IRHS
2100.            K = KSEG(JSTORE)
2101.            IF(NTEMP(JSTORE).LT.IBP(K,JSTORE))KSEG(JSTORE)=K-1
2102.      C              INSTALL OPTIMAL ALLOCATION
2103.       1100 ZBAR = ZTOLZE
2104.            DO 1110 J=1,NSUBS
2105.            IF (JFIX(J) .GT. 0) GO TO 1105
2106.            LAST = NSOL(J) + 1
2107.            IBSTAR(LAST,J) = NTEMP(J)
2108.            ZBAR = ZBAR + ZJBAR(J)
2109.            GO TO 1110
2110.       1105 K = JFIX(J)
2111.            IDELTA = IZSTAR(K,J)
2112.            ZBAR = ZBAR + IDELTA
2113.       1110 CONTINUE
2114.            INCVAL = INT(ZBAR)
2115.            IF (ZBAR .LT. 0.) INCVAL = INCVAL - 1
2116.            RETURN
2117.            END
2118.            SUBROUTINE COMCHK(J,K)
2119.      C
2120.      C         COMPARES CURRENT ALLOCATION (FROM SUBROUTINE ALOCTE) WITH
2121.      C         PREVIOUSLY EXAMINED ALLOCATIONS IN EACH UNFIXED SUBPROBLEM
2122.      C         TO DETERMINE IF OPT. SOL. CORRESPONDING TO CURRENT ALLOCATION
2123.      C         IS ALREADY KNOWN FOR ANY SUBPROBLEM
2124.      C********DESCRIPTION OF PARAMETERS********
2125.      C         J = PARAMETER INDEXING FIRST SUBPROBLEM FOUND FOR WHICH OPTIMAL
2126.      C             SOLUTION CORRESPONDING TO CURRENT ALLOCATION IS ALREADY
2127.      C             KNOWN (OUTPUT)
2128.      C         K = PARAMETER INDEXING THE PREVIOUSLY EXAMINED ALLOCATION IN
2129.      C             SUBPROBLEM J WHICH LEADS TO SAME OPT. SOL. AS WOULD
2130.      C         CURRENT ALLOCATION (OUTPUT)
2131.      C*****************************************
2132.      C
2133.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
2134.           1         INTEGER*4 (I-N,Q)
```

```
      INTEGER*2 IZSTAR,IBID,IBBND,IZBND
      COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
     1IZBAR,IDOM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
     2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
     3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
      DO 1000 J=1,NSUBS
      IF (JFIX(J) .GT. 0) GO TO 1000
      IF (NSOL(J) .EQ. 0) GO TO 1000
      L = NSOL(J)
      LP1 = L + 1
      DO 500 K=1,L
C             TEST ALLOCATION AGAINST UPPER BOUNDS OF
C             BIJ IN KTH SOL. OF SUB J
      IF (IBSTAR(LP1,J) .GT. IBSTAR(K,J)) GO TO 500
C             TEST AGAINST LOWER BOUNDS
      IF(IBSTAR(LP1,J).LT.IBUSED(K,J)) GO TO 500
C
C             ALLOCATION DOMINATED
      IDOM = 1
      RETURN
C
  500 CONTINUE
 1000 CONTINUE
C
C             ALLOCATION NOT DOMINATED
      IDOM = 0
      RETURN
      END
      SUBROUTINE LDDATA(J)
C
C        LOADS DATA FOR SUBPROBLEM J INTO APPROPRIATE STORAGE LOCATIONS
C        FOR SOLUTION BY SUBROUTINES NORMAL (LP) AND BANDB (ILP)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IZSTAR,IBID,IBBND,IZBND
      INTEGER*2 IAS,LAS
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
     1IZBAR,IDOM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
     2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
     3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
      COMMON/BLOCK4/ XLBS(50,10),XUBS(50,10),AMATRS(1000),RHS(20,10),
     1NROWS(10),NCOLS(10),JFIRST(11),IAS(1000),LAS(50,10)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IRGWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      NCOL = NCOLS(J)
      NROW = NROWS(J)
      NELEM = JFIRST(J+1) - JFIRST(J)
      LL = JFIRST(J) - 1
C             LOAD A,IA,LA,XLB,XUB,B,JH,KINBAS
      DO 10 I=1,NELEM
      A(I) = AMATRS(LL+I)
   10 IA(I) = IAS(LL+I)
      DO 20 K=1,NCOL
      LA(K) = LAS(K,J)
```

```
2196.            XLB(K) = XLBS(K,J)
2197.         20 XUB(K) = XUBS(K,J)
2198.            LA(NCOL+1) = NELEM + 1
2199.            B(1) = 0.
2200.            LL = NSOL(J) + 1
2201.            B(2) = FLOAT(IBSTAR(LL,J))
2202.            LL = NROW - 2
2203.            DO 40 I=1,LL
2204.         40 B(I+2) = DBLE(RHS(I,J))
2205.            DO 50 I=1,NROW
2206.            JH(I) = I
2207.         50 KINBAS(I) = I
2208.            LL = NROW + 1
2209.            DO 60 I=LL,NCOL
2210.         60 KINBAS(I) = 0
2211.            ITCNT = 0
2212.            INVFRQ = NROW
2213.            RETURN
2214.            END
2215.            SUBROUTINE GETLBD(J)
2216.      C
2217.      C        COMPARES CURRENT ALLOCATION IN SUBPROBLEM J WITH PREVIOUSLY
2218.      C        EXAMINED ALLOCATIONS IN ORDER TO DETERMINE AN INITIAL LOWER
2219.      C        BOUND ON THE MAXIMAL OBJECTIVE VALUE IN SUBPROBLEM J,
2220.      C        CORRESPONDING TO CURRENT ALLOCATION
2221.      C
2222.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
2223.           1          INTEGER*4 (I-N,Q)
2224.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
2225.            INTEGER*2 IPART,INCUMB,IVBND,IVIO,IOBND
2226.            INTEGER*2 IZSTAR,IBID,IBBND,IZBND
2227.            DOUBLE PRECISION E(1000)
2228.            REAL A(500)
2229.            COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
2230.           1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
2231.           2               IVBND(500),IVID(500),IOBND(500)
2232.            COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
2233.           1IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
2234.           2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
2235.           3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
2236.            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
2237.           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
2238.           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
2239.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
2240.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
2241.            INCUMB(1) = -10000
2242.            KSTORE = 0
2243.            LL = NSOL(J)
2244.            IF (LL .EQ. 0) RETURN
2245.            NEW = LL + 1
2246.      C              SEARCH PREVIOUS ALLOCATIONS FOR BEST LOWER BOUND ON ZJ
2247.            DO 100 K=1,LL
2248.            IF (INCUMB(1) .GE. IZSTAR(K,J))   GO TO 100
2249.            IF  (IBUSED(K,J) .GT. IBSTAR(NEW,J)) GO TO 100
2250.      C              BIJ* >= KTH BIJ: UPDATE INCUMB(1)
2251.            INCUMB(1) = IZSTAR(K,J)
2252.            KSTORE = K
2253.        100 CONTINUE
2254.      C
2255.      C        KSTORE INDEXES THE  PREVIOUSLY EXAMINED ALLOCATION GIVING THE
2256.      C        GREATEST LOWER BOUND ON ZJ
```

```
2257.            IF (KSTORE .EQ. 0) RETURN
2259.            INCUMB(2) = IBSTAR(NEW,J) - IBUSED(KSTORE,J)
2259.            DO 210 K=3,NCOL
2260.          210 INCUMB(K) = 0
2261.            IFEAS = 1
2262.            RETURN
2263.            END
2264.            SUBROUTINE SAVER(J)
2265.      C
2266.      C        STORES ALLOCATION AND SOLUTION INFORMATION FOR SUBPROBLEM J
2267.      C
2268.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
2269.           1          INTEGER*4 (I-N,Q)
2270.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
2271.            INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
2272.            INTEGER*2 IZSTAR,IBID,IBBND,IZBND
2273.            DOUBLE PRECISION E(1000)
2274.            REAL A(500)
2275.            COMMON/BLOCK2/ DEPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
2276.           1               NPIVCT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
2277.           2               IVBND(500),IVID(500),IOBND(500)
2278.            COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
2279.           1IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
2280.           2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
2281.           3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
2282.            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
2283.           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
2284.           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAZ,
2285.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
2286.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
2287.      C        TEST WHETHER SUBPROBLEM J WAS FEASIBLE (IFEAS = 1)
2288.            IF (IFEAS .EQ. 1) GO TO 90
2289.            INCVAL = -10000
2290.            INCUMB(1) = -10000
2291.            INCUMB(2) = 10000
2292.      C
2293.           50 LL = NSOL(J)
2294.            NEW = LL + 1
2295.            IBUSED(NEW,J) = IBSTAR(NEW,J) - INCUMB(2)
2296.            IF (LL .EQ. 0) GO TO 1000
2297.      C             COMPARE NEW SOLUTION IN SUB J WITH PREVIOUS SOLUTIONS
2298.            DO 500 K=1,LL
2299.            IF (IBUSED(NEW,J) .NE. IBUSED(K,J)) GO TO 500
2300.      C          NEW SOLUTION SAME AS KTH:  REWRITE OVER KTH REGION
2301.            IBSTAR(K,J) = IBSTAR(NEW,J)
2302.            NTEMP(2) = K
2303.            RETURN
2304.          500 CONTINUE
2305.      C
2306.      C             NEW SOLUTION DIFFERS FROM ALL PREVIOUS SOLUTIONS
2307.         1000 NSOL(J) = NEW
2308.            IZSTAR(NEW,J) = INCUMB(1)
2309.            NTEMP(2) = NEW
2310.            RETURN
2311.            END
2312.            SUBROUTINE DMBRAN(J,K)
2313.      C
2314.      C        BRANCHES ON ALLOCATION VARIABLES BIJ IN SUBPROBLEM J
2315.      C********DESCRIPTION OF PARAMETERS********
2316.      C        J = PARAMETER INDEXING SUBPROBLEM IN WHICH BRANCHING IS DONE
2317.      C        K = PARAMETER INDEXING A PREVIOUSLY EXAMINED ALLOCATION IN
```

```
C            SUBPROBLEM J.  THE CURRENT ALLOCATION IN SUBPROBLEM J IS TO
C         BE FIXED AT THE KTH LEVEL FOLLOWING THE BRANCHING PROCEDURE
C*****************************************
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IZSTAR,IBID,IBBND,IZBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
     1IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
     2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
     3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      NTEMP(1) = IBUSED(K,J)
      IF (IBSTAR(K,J) .GE. IBUB(J)) GO TO 200
C            ADD UP BRANCH ON BIJ TO LIST
      IBLL = IBLL + 1
      IBID(IBLL) = J
      IBBND(IBLL) = IBUB(J)
      IZBND(IBLL) = IZBAR
C
  200 IF (NTEMP(1) .LE. IBLB(J)) GO TO 500
C            ADD DOWN BRANCH ON BIJ TO LIST
      IBLL = IBLL + 1
      IBID(IBLL) = -J
      IBBND(IBLL) = IBLB(J)
      IZBND(IBLL) = IZBAR
C
C            REVISE BOUNDS ON BIJ FOR FORWARD BRANCH
  500 IF (IBSTAR(K,J).LT.IBUB(J)) IBUB(J)=IBSTAR(K,J)
      IF (NTEMP(1) .GT. IBLB(J)) IBLB(J) = NTEMP(1)
C
      IF (NTEMP(1) .LT. IBLB(J)) GO TO 2000
      MSTAT = 1
      RETURN
C
C            FORWARD BRANCH DOMINATED
 2000 MSTAT = 0
      RETURN
      END
      SUBROUTINE BBKTRK
C
C        BACKTRACKS IN ALLOCATION-VARIABLES SEARCH TREE TO LOCATE NEXT
C        PROMISING NODE TO BE EXAMINED.  LAST-IN-FIRST-OUT (LIFO)
C        SELECTION RULE USED.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 IZSTAR,IBID,IBBND,IZBND
      COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
     1IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
     2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
     3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
C        RETURN TO MAIN IF LIST OF STORED NODES IS EMPTY (COMPUTATIONS
C        COMPLETED)
```

```
      50 IF (IBLL .EQ. 0) RETURN
C        IDENTIFY BRANCHING VARIABLE BIJ CORRESPONDING TO LAST NODE ON
C        LIST
         J = IBID(IBLL)
         J = IABS(J)
C        TEST WHETHER SUBPROBLEM J IS CURRENTLY FIXED
         IF (JFIX(J) .LE. 0) GO TO 100
C        "UNFIX" ALLOCATION TO SUBPROBLEM J NOW
         NJFIX = NJFIX - 1
         K = JFIX(J)
         JFIXSM = JFIXSM - IZSTAR(K,J)
C
  100 JFIX(J) = 0
         IF (IZBND(IBLL) .LE. IZINC) GO TO 2000
C
C              GET NEXT NODE FROM LIST
         IF (IBID(IBLL) .LT. 0) GO TO 200
C        NODE WAS CREATED BY UP-BRANCH ON BIJ
         ITEMP = IBLB(J)
         IBLB(J) = IBUB(J) + 1
         IBUB(J) = IBBND(IBLL)
         IZBAR = IZBND(IBLL)
         JFIX(J) = -1
         GO TO 1000
C
C        NODE WAS CREATED BY DOWN-BRANCH ON BIJ
  200 ITEMP = IBUB(J)
         IBUB(J) = IBLB(J) - 1
         IBLB(J) = IBBND(IBLL)
         IZBAR = IZBND(IBLL)
C
C        "BOOKKEEPING" TO ALLOW ORIGINAL BOUNDS ON BIJ TO BE RETRIEVED
C        LATER
 1000 IBID(IBLL) = -IBID(IBLL)
         IBBND(IBLL) = ITEMP
         IZBND(IBLL) = -10000
         RETURN
C
C              NODE FATHOMED: UPDATE BOUNDS AND BACKTRACK AGAIN
 2000 IF (IBID(IBLL) .LT. 0) GO TO 2200
         IBUB(J) = IBBND(IBLL)
         GO TO 2500
C
 2200 IBLB(J) = IBBND(IBLL)
C
 2500 IBLL = IBLL - 1
         GO TO 50
         END
         SUBROUTINE OUTSOL(INC)
C
C        OUTPUTS SOLUTION INFORMATION
C********DESCRIPTION OF PARAMETERS********
C        INC(J) = PARAMETER INDEXING THE SOLUTION IN SUBPROBLEM J
C                 (J = 1,2,...,NSUBS) FROM WHICH THE OVERALL OPTIMUM IS
C        COMPRISED (INPUT)
C*************************************************
C
         IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
        1          INTEGER*4 (I-N,Q)
         INTEGER*2 IZSTAR,IBID,IBBND,IZBND
         INTEGER*2 IAS,LAS
```

```
      DIMENSION INC(10)
      COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
     1IZBAR,IOOM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
     2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
     3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
      COMMON/BLOCK4/ XLBS(50,10),XUBS(50,10),AMATRS(1000),RHS(20,10),
     1NROWS(10),NCOLS(10),JFIRST(11),IAS(1000),LAS(50,10)
      WRITE (6,10)
   10 FORMAT ('1 OPTIMAL SOLUTION'//' SUBPROBLEM',T20,'SOL. NUMBER')
      DO 100 J=1,NSUBS
      K = INC(J)
  100 WRITE (6,11) J,K
   11 FORMAT (I7,T20,I5)
      WRITE (6,12) IZINC
   12 FORMAT (//' MAXIMUM OBJECTIVE VALUE =',I10)
      RETURN
      END
      SUBROUTINE PHASE1(J)
C
C        SOLVES SUBPROBLEM J AS A LINEAR PROGRAM PARAMETRICALLY IN THE
C        LINKING CONSTRAINT RIGHT-HAND SIDE.  SIMULTANEOUSLY MODIFIES
C        THE RESULTING CONCAVE AND PIECEWISE-LINEAR PARAMETRIC OBJECTIVE
C        FUNCTION TO RENDER ALL BREAKPOINTS IBP(K,J) (K = 1,2,...,
C        NLAMDA(J)) INTEGRAL.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IZSTAR,IBID,IBBND,IZBND
      DOUBLE PRECISION E(1000)
      DIMENSION A(500),INC(10)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEU,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK3/ RLAMDA(50,10),ZNAUT(10),ZJBAR(10),NSUBS,IZINC,
     1IZBAR,IOOM,IBLL,NJFIX,JFIXSM,IB,JFIX(10),NSOL(10),NLAMDA(10),
     2KSEG(10),IBSTAR(200,10),IBUB(10),IBLB(10),IBP(51,10),
     3IBUSED(200,10),IZSTAR(200,10),IBID(200),IBBND(200),IZBND(200)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C              GET OPTIMAL LP SOLUTION WITH RHS = IBLB(J)
      ITSINV = 99999
      CALL NORMAL
      ZNAUT(J) = SNGL(X(IOBJ))
      IBBAR = IBLB(J)
      K = 0
C
   25 K = K + 1
      IBP(K,J) = IBBAR
C
C              COMPUTE NEXT BREAKPOINT VALUE BBAR
      CALL UNPACK(2)
      CALL FTRAN(1)
      IF (Y(IOBJ) .LE. ZTCOST) GO TO 1000
      DP = 1.E10
      DO 100 I=1,NROW
      IF (I .EQ. IOBJ) GO TO 100
      JCOLP = JH(I)
```

```
2501.            IF (Y(I) .GE. ZTOLPV) GO TO 60
2502.            IF (Y(I) .LE. -ZTOLPV) GO TO 50
2503.            GO TO 100
2504.         50 DY = XLB(JCOLP)
2505.            GO TO 70
2506.         60 DY = XUB(JCOLP)
2507.         70 DE = (DY - X(I))/Y(I)
2508.            IF (DP .LE. DE) GO TO 100
2509.            DP = DE
2510.            IROWP = I
2511.        100 CONTINUE
2512.            DTEMP = FLOAT(IBBAR) + DP
2513.            DE = DTEMP + ZTOLZE
2514.            IBBAR = IDINT(DE)
2515.            IF (DE .LT. 0.) IBBAR = IBBAR - 1
2516.            IF (IBBAR .GE. IBUB(J)) GO TO 1000
2517.            TEMP = FLOAT(IBBAR)
2518.            DE = DTEMP - TEMP
2519.            IF (IBBAR .EQ. IBP(K,J)) GO TO 500
2520.            IF (DE .LT. ZTOLZE) GO TO 400
2521.            GO TO 600
2522.      C
2523.      C                  NEW BREAKPOINT BBAR NATURALLY INTEGER
2524.        400 RLAMDA(K,J) = SNGL(Y(IOBJ))
2525.      C                  UPDATE X TO REFLECT VALUES AT NEW BBAR, THEN PIVOT
2526.            DO 410 I=1,NROW
2527.        410 X(I) = X(I) + Y(I)*DP
2528.            B(2) = FLOAT(IBBAR)
2529.            CALL NORMAL
2530.            GO TO 25
2531.      C
2532.      C                  NEW BREAKPOINT BBAR < PREVIOUS BBAR + 1
2533.        500 DTEMP = X(IOBJ)
2534.            IBBAR = IBBAR + 1
2535.      C                  UPDATE X:  NEW BBAR = OLD BBAR + 1
2536.            DO 510 I=1,NROW
2537.        510 X(I) = X(I) + Y(I)
2538.            B(2) = FLOAT(IBBAR)
2539.            CALL NORMAL
2540.            RLAMDA(K,J) = SNGL(X(IOBJ) - DTEMP)
2541.            GO TO 25
2542.      C
2543.      C                  NEW BREAKPOINT BBAR > OLD BBAR + 1, BUT NON-INTEGER
2544.        600 RLAMDA(K,J) = SNGL(Y(IOBJ))
2545.            K = K + 1
2546.            IBP(K,J) = IBBAR
2547.            DTEMP = X(IOBJ) + Y(IOBJ)*(DP - DE)
2548.            IBBAR = IBBAR + 1
2549.            DP = DP + (1. - DE)
2550.            DO 610 I=1,NROW
2551.        610 X(I) = X(I) + Y(I)*DP
2552.            B(2) = FLOAT(IBBAR)
2553.            CALL NORMAL
2554.            RLAMDA(K,J) = SNGL(X(IOBJ) - DTEMP)
2555.            GO TO 25
2556.      C
2557.      C                  END OF PLP
2558.       1000 RLAMDA(K,J) = SNGL(Y(IOBJ))
2559.            NLAMDA(J) = K
2560.            IBP(K+1,J) = IBUB(J)
2561.            RETURN
```

```
2562.          END
```

```
*****SAMPLE INPUT DATA FOR DECOMPOSITION CODE DSLC*****

         4        293          1          0
    1
NAME            SUB1
ROWS
 N  OBJ
 L  ROW1
 L  ROW11
 L  ROW12
 L  ROW13
 L  ROW14
COLUMNS
     COL11      OBJ        -56.          ROW1       40.
     COL11      ROW11       16.          ROW12      38.
     COL11      ROW13        8.          ROW14      38.
     COL12      OBJ       -113.          ROW1       91.
     COL12      ROW11       92.          ROW12      39.
     COL12      ROW13       71.          ROW14      52.
     COL13      OBJ        -30.          ROW1       10.
     COL13      ROW11       41.          ROW12      32.
     COL13      ROW13       30.          ROW14      30.
     COL14      OBJ        -62.          ROW1       30.
     COL14      ROW11       16.          ROW12      71.
     COL14      ROW13       60.          ROW14      42.
     COL15      OBJ       -210.          ROW1      160.
     COL15      ROW11      150.          ROW12      80.
     COL15      ROW13      200.          ROW14     170.
RHS
     RHS1       ROW11      221.          ROW12     182.
     RHS1       ROW13      258.          ROW14     232.
ENDATA
    0.    0.    0.    0.    0.
    1.    1.    1.    1.    1.
          0         200
    2
NAME            SUB2
ROWS
 N  OBJ
 L  ROW1
 L  ROW21
 L  ROW22
 L  ROW23
 L  ROW24
COLUMNS
     COL21      OBJ        -43.          ROW1       20.
     COL21      ROW21       23.          ROW22      26.
     COL21      ROW23       18.          ROW24       9.
     COL22      OBJ         -7.          ROW1        3.
     COL22      ROW21        4.          ROW22       5.
     COL22      ROW23        6.          ROW24       7.
     COL23      OBJ        -33.          ROW1       12.
     COL23      ROW21       18.          ROW22      40.
     COL23      ROW23       30.          ROW24      20.
     COL24      OBJ         -5.          ROW1        3.
     COL24      ROW21        6.          ROW22       8.
     COL24      ROW23        4.
     COL25      OBJ        -12.          ROW1       18.
```

```
    COL25     ROW22     12.          ROW23     8.
    COL25     ROW24     3.
RHS
    RHS1      ROW21     36.          ROW22     64.
    RHS1      ROW23     46.          ROW24     27.
ENDATA
   0.   0.   0.   0.   0.
   1.   1.   1.   1.   1.
         0         44
   3
NAME          SUB3
ROWS
 N  OBJ
 L  ROW1
 L  ROW31
 L  ROW32
 L  ROW33
 L  ROW34
COLUMNS
    COL31     OBJ       -32.         ROW1      9.
    COL31     ROW31     12.          ROW32     30.
    COL31     ROW33     31.          ROW34     21.
    COL32     OBJ       -20.         ROW1      25.
    COL32     ROW31     8.           ROW32     15.
    COL32     ROW33     6.           ROW34     4.
    COL33     OBJ       -4.          ROW1      1.
    COL33     ROW31     2.           ROW33     3.
    COL33     ROW34     1.
    COL34     OBJ       -3.          ROW1      1.
    COL34     ROW31     1.           ROW32     1.
    COL34     ROW34     2.
    COL35     OBJ       -43.         ROW1      10.
    COL35     ROW32     23.          ROW33     18.
    COL35     ROW34     14.
RHS
    RHS1      ROW31     16.          ROW32     48.
    RHS1      ROW33     41.          ROW34     29.
ENDATA
   0.   0.   0.   0.   0.
   1.   1.   1.   1.   1.
         0         37
   4
NAME          SUB4
ROWS
 N  OBJ
 L  ROW1
 L  ROW41
 L  ROW42
 L  ROW43
 L  ROW44
COLUMNS
    COL41     OBJ       -426.        ROW1      280.
    COL41     ROW41     200.         ROW42     100.
    COL41     ROW43     60.          ROW44     310.
    COL42     OBJ       -42.         ROW1      10.
    COL42     ROW41     20.          ROW43     21.
    COL42     ROW44     8.
    COL43     OBJ       -12.         ROW1      8.
    COL43     ROW41     6.           ROW42     20.
    COL43     ROW43     4.           ROW44     4.
    COL44     OBJ       -8.          ROW1      1.
```

```
    COL44     ROW41     2.          ROW42     3.
    COL44     ROW44     6.
    COL45     OBJ       -2.         ROW1      1.
    COL45     ROW41     1.          ROW43     2.
    COL45     ROW44     1.
RHS
    RHS1      ROW41     160.        ROW42     86.
    RHS1      ROW43     61.         ROW44     230.
ENDATA
   0.   0.   0.   0.   0.
   1.   1.   1.   1.   1.
         0         20
```

```
*****OUTPUT GENERATED BY DSLC ON SAMPLE PROBLEM*****


 START NEW PROBLEM


IBP(K,  1) =    0   10   11   40   41   80   81  162  163  185  186  193  194  200
RLAMDA(K,  1) =  3.00  2.07  2.07  1.40  1.40  1.31  1.31  1.22  1.22  1.18  1.03  1.03  1.01
ZNAUT(  1) =   -0.0000


IBP(K,  2) =    0   12   13   15   16   27   28   36   37   39   40   42   43   44   44
RLAMDA(K,  2) =  2.75  2.33  2.33  2.11  2.11  1.60  0.64  0.63  0.62  0.62  0.61  0.59  0.37  0.26
ZNAUT(  2) =   -0.0000


IBP(K,  3) =    0   10   11   12   16   17   18   30   31   34   35   37
RLAMDA(K,  3) =  4.30  4.00  3.56  3.56  3.45  0.67  0.60  0.59  0.57  0.32  0.20
ZNAUT(  3) =   -0.0000


IBP(K,  4) =    0    1    2   11   12   13   20
RLAMDA(K,  4) =  8.00  4.20  4.20  2.00  1.52  1.52
ZNAUT(  4) =   -0.0000



SUB 1 SOL  1:    266    0   55   64   50   24    1    0    0    0    1
SUB 2 SOL  1:     62    1    9   21   14    8    1    1    0    0    1
SUB 3 SOL  1:     50   20   13   24   20   12    0    0    1    1    1
SUB 4 SOL  1:     64    0  131   63   34  211    0    1    1    1    1
TIME =    1.10 SECONDS:  IZINC =       442
SUB 4 SOL  2:     62    0  132   63   36  212    0    1    1    1    0
SUB 3 SOL  1:     50   21    0    0    0    0    0    0    0    0    0
SUB 3 SOL  1:     50   22    0    0    0    0    0    0    0    0    0
SUB 4 SOL  3:     52    5  137   83   38  215    0    1    0    1    1
SUB 3 SOL  2:     70    0    5    9   14    8    0    1    1    1    1
TIME =    1.62 SECONDS:  IZINC =       450
SUB 2 SOL  2:     55    2   13   26   20   15    1    0    0    0    1
SUB 3 SOL  3:     63    0    8   10   17   11    0    1    0    0    1
SUB 3 SOL  4:     67    0    6   10   14   10    0    1    1    0    1
SUB 4 SOL  4:     54    0  134   66   36  218    0    1    1    0    0
SUB 2 SOL  3:     55   11    3   25   18   11    1    1    0    1    0
TIME =    2.26 SECONDS:  IZINC =       452
TIME =    2.29 SECONDS:  IZINC =       455
SUB 2 SOL  4:     50    2    9   33   22   11    1    1    0    0    0
SUB 2 SOL  5:     67    0    3   13   10    8    1    1    0    1    1
SUB 2 SOL  1:     62    2    0    0    0    0    0    0    0    0    0
SUB 1 SOL  2:    261   28   56    2   89   70    1    1    1    1    0
TIME =    2.92 SECONDS:  IZINC =       457
SUB 1 SOL  3:    240    0   30   70   28   32    0    0    1    0    1
TIME =    3.10 SECONDS:  IZINC =       462


 OPTIMAL SOLUTION
```

```
SUBPROBLEM        SOL. NUMBER
     1                2
     2                5
     3                2
     4                1


MAXIMUM OBJECTIVE VALUE =       462

TOTAL SOLUTION TIME =   3.12 SECONDS
TIME IN PHASE1 =  0.60 SECONDS
MASTER PROBLEM SOLVED        91 TIMES:  TOTAL TIME SOLVING MASTER PROBLEM =  0.22 SECONDS
NUMBER OF TIMES EACH SUBPROBLEM SOLVED =                    3    6    6    4
NUMBER OF DISTINCT SOLUTIONS OBTAINED IN EACH SUBPROBLEM =  3    5    4    4
TOTAL TIME SOLVING SUBPROBLEMS =  2.20 SECONDS
```

APPENDIX III

LISTING AND SAMPLE INPUT/OUTPUT FOR COMPUTER PROGRAM DMLC

*****OUTPUT GENERATED BY DMEE ON SAMPLE PROBLEM*****

```
SUB 1 SOL  1:    266    0   55   64   50   24    1    0    0    0    1
SUB 2 SOL  1:     62    1    9   21   14    8    1    1    0    0    1
SUB 3 SOL  1:     50   20   13   24   20   12    0    0    1    1    1
SUB 4 SOL  1:     64    0  131   63   34  211    0    1    1    1    1
TIME =    1.64 SECONDS:  IZINC =       442
SUB 3 SOL  2:     47    0   14   25   20   14    0    0    1    0    1
SUB 3 SOL  3:     50    0    0    0    0    0    0    0    0    0    0
SUB 3 SOL  4:     50    0    0    0    0    0    0    0    0    0    0
SUB 3 SOL  5:     63    0    8   10   17   11    0    1    0    0    1
SUB 4 SOL  2:     52    5  137   83   38  215    0    1    0    1    1
TIME =    2.44 SECONDS:  IZINC =       443
SUB 3 SOL  6:     70    0    5    9   14    8    0    1    1    1    1
SUB 4 SOL  3:     50    0  138   83   40  216    0    1    0    1    0
TIME =    2.69 SECONDS:  IZINC =       448
SUB 3 SOL  7:     67    0    6   10   14   10    0    1    1    0    1
SUB 2 SOL  2:     55    2   13   26   20   15    1    0    0    0    1
SUB 2 SOL  3:     55   11    3   25   18   11    1    1    0    1    0
SUB 4 SOL  4:     62    0  132   63   36  212    0    1    1    1    0
TIME =    4.19 SECONDS:  IZINC =       450
SUB 4 SOL  5:     54    0  134   60   36  218    0    1    1    0    0
TIME =    4.45 SECONDS:  IZINC =       455
SUB 2 SOL  4:     50    2    9   33   22   11    1    1    0    0    0
SUB 2 SOL  5:     67    0    3   13   10    8    1    1    0    1    1
SUB 2 SOL  6:     62    0    0    0    0    0    0    0    0    0    0
SUB 1 SOL  2:    261   28   56    2   89   70    1    1    1    1    0
TIME =    7.05 SECONDS:  IZINC =       457
SUB 1 SOL  3:    240    0   36   70   28   32    0    0    1    0    1
SUB 1 SOL  4:    231    8   57   34  119  100    1    1    0    1    0
TIME =   10.25 SECONDS:  IZINC =       462
 OPTIMAL SOLUTION
```

SUBPROBLEM	SOL. NUMBER
1	2
2	5
3	6
4	1

```
MAXIMUM OBJECTIVE VALUE =       462

TOTAL SOLUTION TIME =  10.96SECONDS
TIME IN PHASE1 =  0.85 SECONDS
MASTER PROBLEM SOLVED       229 TIMES:  TOTAL TIME SOLVING MASTER PROBLEM =   7.62 SECONDS
NUMBER OF TIMES EACH SUBPROBLEM SOLVED =                          4    6    7    5
NUMBER OF DISTINCT SOLUTIONS OBTAINED IN EACH SUBPROBLEM =    4    6    7    5
TOTAL TIME SOLVING SUBPROBLEMS =  2.35 SECONDS
```

```
    CCL42       CBJ         -42.            ROW1        10.
    CCL42       RCW41       20.             ROW43       21.
    CCL42       RCW44       8.
    CCL43       CBJ         -12.            ROW1        8.
    CCL43       RCW41       6.              ROW42       20.
    CCL43       RCW43       4.              ROW44       4.
    CCL44       CBJ         -3.             ROW1        1.
    CCL44       RCW41       2.              ROW42       3.
    CCL44       RCW44       6.
    CCL45       CBJ         -2.             ROW1        1.
    CCL45       RCW41       1.              ROW43       2.
    CCL45       RCW44       1.
RHS
    RHS1        RCW1        10.
    RHS1        RCW41       100.            ROW42       86.
    RHS1        RCW43       61.             ROW44       230.
ENDATA
      0.    0.    0.    0.    0.
      1.    1.    1.    1.    1.
              0
             20
```

```
    COL24     ROW21     6.              ROW22     8.
    COL24     ROW23     4.
    COL25     OBJ       -12.            ROW1      18.
    COL25     ROW22     12.             ROW23     8.
    COL25     ROW24     3.
RHS
    RHS1      ROW1      22.
    RHS1      ROW21     36.             ROW22     64.
    RHS1      ROW23     46.             ROW24     27.
ENDATA
   0.   0.   0.   0.   0.
   1.   1.   1.   1.   1.
         0
        44

   3
NAME          SLB3
ROWS
 N  OBJ
 L  ROW1
 L  ROW31
 L  ROW32
 L  ROW33
 L  ROW34
COLUMNS
    COL31     OBJ       -32.            ROW1      9.
    COL31     ROW31     12.             ROW32     30.
    COL31     ROW33     31.             ROW34     21.
    COL32     OBJ       -20.            ROW1      25.
    COL32     ROW31     8.              ROW32     15.
    COL32     ROW33     6.              ROW34     4.
    COL33     OBJ       -4.             ROW1      1.
    COL33     ROW31     2.              ROW33     3.
    COL33     ROW34     1.
    COL34     OBJ       -3.             ROW1      1.
    COL34     ROW31     1.              ROW32     1.
    COL34     ROW34     2.
    COL35     OBJ       -43.            ROW1      10.
    COL35     ROW32     23.             ROW33     18.
    COL35     ROW34     14.
RHS
    RHS1      ROW1      18.
    RHS1      ROW31     16.             ROW32     48.
    RHS1      ROW33     42.             ROW34     29.
ENDATA
   0.   0.   0.   0.   0.
   1.   1.   1.   1.   1.
         0
        37

   4
NAME          SLB4
ROWS
 N  OBJ
 L  ROW1
 L  ROW41
 L  ROW42
 L  ROW43
 L  ROW44
COLUMNS
    COL41     OBJ       -420.           ROW1      280.
    COL41     ROW41     200.            ROW42     100.
    COL41     ROW43     60.             ROW44     310.
```

41 57

*****SAMPLE INPUT DATA FOR DECOMPOSITION CODE DMLC*****

```
         4          1          1          0
       293
   1
NAME           SLB1
ROWS
 N  OBJ
 L  ROW1
 L  ROW11
 L  ROW12
 L  ROW13
 L  ROW14
COLUMNS
    COL11      OBJ        -50.            ROW1       40.
    COL11      ROW11      10.             ROW12      38.
    COL11      ROW13      3.              ROW14      38.
    COL12      OBJ        -113.           ROW1       91.
    COL12      ROW11      92.             ROW12      39.
    COL12      ROW13      71.             ROW14      52.
    COL13      OBJ        -30.            ROW1       10.
    COL13      ROW11      41.             ROW12      32.
    COL13      ROW13      30.             ROW14      30.
    COL14      OBJ        -62.            ROW1       30.
    COL14      ROW11      16.             ROW12      71.
    COL14      ROW13      60.             ROW14      42.
    COL15      OBJ        -210.           ROW1       160.
    COL15      ROW11      150.            ROW12      80.
    COL15      ROW13      200.            ROW14      170.
RHS
    RHS1       ROW1       100.
    RHS1       ROW11      221.            ROW12      182.
    RHS1       ROW13      258.            ROW14      232.
ENDATA
   0.    0.    0.    0.    0.
   1.    1.    1.    1.    1.
         0
       200
   2
NAME           SLB2
ROWS
 N  OBJ
 L  ROW1
 L  ROW21
 L  ROW22
 L  ROW23
 L  ROW24
COLUMNS
    COL21      OBJ        -43.            ROW1       20.
    COL21      ROW21      23.             ROW22      26.
    COL21      ROW23      10.             ROW24      9.
    COL22      OBJ        -7.             ROW1       3.
    COL22      ROW21      4.              ROW22      5.
    COL22      ROW23      6.              ROW24      7.
    COL23      OBJ        -33.            ROW1       12.
    COL23      ROW21      18.             ROW22      40.
    COL23      ROW23      30.             ROW24      20.
    COL24      OBJ        -5.             ROW1       3.
```

```
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 IZSTAR,IBID,IBBND,IZBND
      INTEGER*2 IAS,LAS
      DIMENSION INC(10)
      COMMON/BLOCK3/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
     1IZINC,IZBAR,IDUM,IELL,NJFIX,JFIXSW,IB(5),JFIX(10),NSOL(10),
     2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
     3IZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
      COMMON/BLOCK4/ XLBS(20,10),XUBS(20,10),AMATRS(1000),RHS(10,10),
     1NROWS(10),NCOLS(10),JFIRST(11),IAS(1000),LAS(20,10)
      WRITE (6,10)
   10 FORMAT ('1 OPTIMAL SOLUTION'//' SUBPROBLEM',T20,'SOL. NUMBER')
      DO 100 J=1,NSUBS
      K = INC(J)
  100 WRITE (6,11) J,K
   11 FORMAT (I7,T20,I5)
      WRITE (6,12) IZINC
   12 FORMAT (//' MAXIMUM OBJECTIVE VALUE =',I10)
      RETURN
      END
```

```
      3IZSTAR(4C,1C),IBID(2,2CC),IBBND(2CC),IZBND(2CC)
C        RETURN TC MAIN IF LIST OF STORED NODES IS EMPTY (COMPUTATIONS
C        COMPLETED)
    5C IF (IBLL .EQ. C) RETURN
C        IDENTIFY BRANCHING VARIABLE BIJ CORRESPONDING TO LAST NODE ON
C        LIST
       I = IBID(1,IBLL)
       J = IBID(2,IBLL)
C        TEST WHETHER SUBPROBLEM J IS CURRENTLY FIXED
       IF (JFIX(J) .EQ. C) GO TO 100
C        "UNFIX" ALLOCATION TO SUBPROBLEM J NOW
       NJFIX = NJFIX - 1
       K = JFIX(J)
       JFIXSM = JFIXSM - IZSTAR(K,J)
       JFIX(J) = C
C
  1CC IF (IZBND(IBLL) .LE. IZINC) GO TO 2CCC
C
C              GET NEXT NODE FROM LIST
       IF (I .LT. C) GO TO 200
C        NODE WAS CREATED BY UP-BRANCH ON BIJ
       ITEMP = IBLB(I,J)
       IBLB(I,J) = IBLB(I,J) + 1
       IBLB(I,J) = IBBND(IBLL)
       IZBAR = IZBND(IBLL)
       GO TO 1000
C
C        NODE WAS CREATED BY DOWN-BRANCH ON BIJ
  2CC I = -I
       ITEMP = IBLB(I,J)
       IBLB(I,J) = IBLB(I,J) - 1
       IBLB(I,J) = IBBND(IBLL)
       IZBAR = IZBND(IBLL)
C
C        "BOOKKEEPING" TO ALLOW ORIGINAL BOUNDS ON BIJ TO BE RETRIEVED
C        LATER
 1CCC IBID(1,IBLL) = -IBID(1,IBLL)
       IBBND(IBLL) = ITEMP
       IZBND(IBLL) = -10CC
       RETURN
C
C              NODE FATHOMED: UPDATE BOUNDS AND BACKTRACK AGAIN
 2CCC IF (I .LT. C) GO TO 2200
       IBLB(I,J) = IBBND(IBLL)
       GO TO 2500
C
 22CC I = -I
       IBLB(I,J) = IBBND(IBLL)
C
 25CC IBLL = IBLL - 1
       GO TO 50
       END
       SUBROUTINE OUTSOL(INC)
C
C        OUTPUTS SOLUTION INFORMATION
C********DESCRIPTION OF PARAMETERS********
C        INC(J) = PARAMETER INDEXING THE SOLUTION IN SUBPROBLEM J
C                 (J = 1,2,...,NSUBS) FROM WHICH THE OVERALL OPTIMUM IS
C        COMPRISED (INPUT)
C*************************************************
C
```

```
      C
            IMPLICIT REAL*4 (A,C,E-H,L,P,R-W,Z), REAL*8 (B,D,X,Y),
           1        INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            INTEGER*2 IZSTAR,IBID,IBBND,IZBND
            DOUBLE PRECISION E(1000)
            REAL A(500)
            COMMON/BLOCKS/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
           1IZINC,IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
           2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
           3IZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
           1       DY,DE,DP,A,ICNAME(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
           2       IUBD,IRGAP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
           3       JDELF,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
            DO 1000 I=1,NLINK
            NTEMP(I) = IBUSED(I,K,J)
            IF (IBSTAR(I,K,J) .EQ. IBUB(I,J)) GO TO 200
      C              ADD UP BRANCH ON BIJ TO LIST
            IBLL = IBLL + 1
            IBID(1,IBLL) = I
            IBID(2,IBLL) = J
            IBBND(IBLL) = IBUB(I,J)
            IZBND(IBLL) = IZBAR
      C
        200 IF (NTEMP(I) .LE. IBLB(I,J)) GO TO 900
      C              ADD DOWN BRANCH ON BIJ TO LIST
            IBLL = IBLL + 1
            IBID(1,IBLL) = -I
            IBID(2,IBLL) = J
            IBBND(IBLL) = IBLB(I,J)
            IZBND(IBLL) = IZBAR
      C
      C              REVISE BOUNDS ON BIJ FOR FORWARD BRANCH
        500 IF (IBSTAR(I,K,J) .LT. IBUB(I,J)) IBUB(I,J) = IBSTAR(I,K,J)
            IF (NTEMP(I) .GT. IBLB(I,J)) IBLB(I,J) = NTEMP(I)
       1000 CONTINUE
      C
            DO 1500 I=1,NLINK
            IF (NTEMP(I) .LT. IBLB(I,J)) GO TO 2000
       1500 CONTINUE
            MSTAT = 1
            RETURN
      C
      C              FORWARD BRANCH DOMINATED
       2000 MSTAT = 0
            RETURN
            END
            SUBROUTINE BKTRK
      C
      C        BACKTRACKS IN ALLOCATION-VARIABLES SEARCH TREE TO LOCATE NEXT
      C        PROMISING NODE TO BE EXAMINED.  LAST-IN-FIRST-OUT (LIFO)
      C        SELECTION RULE USED.
      C
            IMPLICIT REAL*4 (A,C,E-H,L,P,R-W,Z), REAL*8 (B,D,X,Y),
           1        INTEGER*4 (I-N,Q)
            INTEGER*2 IZSTAR,IBID,IBBND,IZBND
            COMMON/BLOCKS/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
           1IZINC,IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
           2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
```

```
      DOUBLE PRECISION B(1000)
      REAL A(500)
      COMMON/BLOCK2/ DPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
     2               IVBND(500),IVID(500),IOBND(500)
      COMMON/BLOCK3/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
     1IIZINC,IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
     2RLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
     3IZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
      COMMON B,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),USUM,UPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFREZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C          TEST WHETHER SUBPROBLEM J WAS FEASIBLE (IFEAS = 1)
      IF (IFEAS .EQ. 1) GO TO 90
      NP1 = NLINK + 1
      INCVAL = -10000
      INCOMB(1) = -10000
      DO 10 L = 2,NP1
   10 INCOMB(L) = 10000
      NP1 = NP1 + 1
      DO 20 L = NP1,NCOL
   20 INCOMB(L) = -1
C
   90 LL = NSOL(J)
      NEW = LL + 1
      DO 95 I=1,NLINK
   95 IBUSED(I,NEW,J) = IBSTAR(I,NEW,J) - INCOMB(I+1)
      IF (LL .EQ. 0) GO TO 1000
C               COMPARE NEW SOLUTION IN SUB J WITH PREVIOUS SOLUTIONS
      DO 500 K=1,LL
      DO 200 L=1,NLINK
      IF (IBUSED(L,NEW,J) .NE. IBUSED(L,K,J)) GO TO 500
  200 CONTINUE
C          NEW SOLUTION SAME AS KTH: CHECK IF NEW REGION INCLUDES KTH
      DO 300 I=1,NLINK
      IF (IBSTAR(I,K,J) - IBSTAR(I,NEW,J)) 300,300,500
  300 CONTINUE
C          NEW REGION INCLUDES KTH:  REWRITE OVER KTH REGION
      DO 400 I=1,NLINK
  400 IBSTAR(I,K,J) = IBSTAR(I,NEW,J)
      NTEMP(2) = K
      RETURN
  500 CONTINUE
C
C          NEW SOLUTION DIFFERS FROM ALL PREVIOUS SOLUTIONS
 1000 NSOL(J) = NEW
      IZSTAR(NEW,J) = INCOMB(1)
      NTEMP(2) = NEW
      RETURN
      END
      SUBROUTINE BRANCH(J,K)
C
C          BRANCHES ON ALLOCATION VARIABLES BIJ IN SUBPROBLEM J
C********DESCRIPTION OF PARAMETERS********
C          J = PARAMETER INDEXING SUBPROBLEM IN WHICH BRANCHING IS DONE
C          K = PARAMETER INDEXING A PREVIOUSLY EXAMINED ALLOCATION IN
C              SUBPROBLEM J.  THE CURRENT ALLOCATION IN SUBPROBLEM J IS TO
C          BE FIXED AT THE KTH LEVEL FOLLOWING THE BRANCHING PROCEDURE
C*************************************************
```

```
      DO 210 K=LL,NCOL
  210 INCOMB(K) = 0
      IFEAS = 1
      RETURN
      END
      SUBROUTINE MISAVR(J)
C
C        CHECKS IF LP-OPTIMAL DUAL MULTIPLIERS JUST OBTAINED IN
C        SUBPROBLEM J ARE DISTINCT FROM PREVIOUS SETS OF MULTIPLIERS.
C        IF DISTINCT, STORES NEW MULTIPLIERS ASSOCIATED WITH LINKING
C        CONSTRAINTS.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IZSTAR,IB1D,IBBND,IZBND
      DOUBLE PRECISION B(1000)
      REAL A(900)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,CRO,OMA,OBA,OFI,
     1              OBO,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OO,NRMAX,NEMAX,OB,OC,
     2              OE,OF,OL,OO,OR,OM,OG,NTMAX
      COMMON/BLOCK2/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
     1IIZINC,IZBAR,IOOM,IBLL,NOFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
     2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
     3IZSTAR(40,10),IB1D(2,200),IBBND(200),IZBND(200)
      COMMON B,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),USUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IREWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IPFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(900),LE(252),LA(122),KINBAS(122),JH(60)
      LL = NLAMDA(J)
      IF (LL .EQ. 0) GO TO 1000
C        TEST IF OPTIMAL DUAL MULTIPLIERS DISTINCT FROM PREVIOUS SETS
      DO 100 K=1,LL
      DO 90 I=1,NLINK
      IF (DABS(Y(I+1) - RLAMDA(I,K,J)) .GT. ZTOLZE) GO TO 100
   90 CONTINUE
      RETURN
  100 CONTINUE
C
C        NEW MULTIPLIERS DISTINCT;  STORE THEM
 1000 LL = LL + 1
      DO 1150 I=1,NLINK
 1150 RLAMDA(I,LL,J) = SNGL(Y(I+1))
      DP = X(100J)
      K = NLINK + 1
      DO 1200 I=2,K
 1200 DP = DP - Y(I)*B(I)
      ZNAUT(LL,J) = SNGL(DP)
      NLAMDA(J) = LL
      RETURN
      END
      SUBROUTINE SAVER(J)
C
C        STORES ALLOCATION AND SOLUTION INFORMATION FOR SUBPROBLEM J
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVIO,ICBND
      INTEGER*2 IZSTAR,IB1D,IBBND,IZBND
```

```
      LL = NROW - K
      DO 40 I=1,LL
   40 B(K+1) = DBLE(RHSIDE(I,J))
      DO 50 I=1,NROW
      JH(I) = 1
   50 KINBAS(I) = I
      LL = NROW + 1
      DO 60 I=LL,NCOL
   60 KINBAS(I) = 0
      ITCNT = 0
      INVFRQ = NROW
      RETURN
      END
      SUBROUTINE GETLOB(J)
C
C        COMPARES CURRENT ALLOCATION IN SUBPROBLEM J WITH PREVIOUSLY
C        EXAMINED ALLOCATIONS IN ORDER TO DETERMINE AN INITIAL LOWER
C        BOUND ON THE MAXIMAL OBJECTIVE VALUE IN SUBPROBLEM J,
C        CORRESPONDING TO CURRENT ALLOCATION
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
      INTEGER*2 IZSTAR,IBID,IBBND,IZBND
      DOUBLE PRECISION BB(500)
      REAL A(500)
      COMMON/BLOCK2/ DPPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
     2               IVBND(500),IVID(500),ICBND(500)
      COMMON/BLOCK3/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
     1IZINC,IZBAR,ISUM,IBLE,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
     2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
     3IZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
      COMMON B,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IUBD,IRCAP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFREZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      INCUMB(1) = -10000
      KSTORE = 0
      LL = NSOL(J)
      IF (LL .EQ. 0) RETURN
      NEW = LL + 1
C           SEARCH PREVIOUS ALLOCATIONS FOR BEST LOWER BOUND ON ZJ
      DO 100 K=1,LL
      IF (INCUMB(1) .GE. IZSTAR(K,J)) GO TO 100
      DO 90 I=1,NLINK
      IF (IBUSED(I,K,J) .GT. IBSTAR(I,NEW,J)) GO TO 100
   90 CONTINUE
C        BIJ* >= KTH BIJ USED:  UPDATE INCUMB(1)
      INCUMB(1) = IZSTAR(K,J)
      KSTORE = K
  100 CONTINUE
C
C        KSTORE INDEXES THE PREVIOUSLY EXAMINED ALLOCATION GIVING THE
C        GREATEST LOWER BOUND ON ZJ
      IF (KSTORE .EQ. 0) RETURN
      DO 200 I=1,NLINK
  200 INCUMB(I+1) = IBSTAR(I,NEW,J) - IBUSED(I,KSTORE,J)
      LL = NLINK + 1
```

```
      DO 400 I=1,NLINK
      IF (IESTAR(I,LPI,J) .GT. IBSTAR(I,K,J)) GO TO 500
  400 CONTINUE
C                 TEST AGAINST LOWER BOUNDS
      DO 450 I=1,NLINK
      IF (IESTAR(I,LPI,J) .LT. IBUSED(I,K,J)) GO TO 500
  450 CONTINUE
C
C                 ALLOCATION DOMINATED
      IDOM = 1
      RETURN
C
  500 CONTINUE
 1000 CONTINUE
C
C                 ALLOCATION NOT DOMINATED
      IDOM = 0
      RETURN
      END
      SUBROUTINE LODATA(J)
C
C        LOADS DATA FOR SUBPROBLEM J INTO APPROPRIATE STORAGE LOCATIONS
C        FOR SOLUTION BY SUBROUTINES NORMAL (LP) AND BAND8 (ILP)
C
      IMPLICIT REAL*4 (A,C,E-H,C,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,G)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IESTAR,IBID,IBBND,IZBND
      INTEGER*2 IAS,LAS
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK3/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
     1IZINC,IZBAR,IDOM,IBLL,NJFIX,JFIXSW,IB(5),JFIX(10),NSOL(10),
     2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
     3IZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
      COMMON/BLOCK4/ XLBS(20,10),XUBS(20,10),AMATRS(1000),RHS(10,10),
     1NROWS(10),NCOLS(10),JFIRST(11),IAS(1000),LAS(20,10)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IRHSP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      NCOL = NCOLS(J)
      NROW = NROWS(J)
      NELEM = JFIRST(J+1) - JFIRST(J)
      LL = JFIRST(J) - 1
C                 LOAD A,IA,LA,XLB,XUB,B,JH,KINBAS
      DO 10 I=1,NELEM
      A(I) = AMATRS(LL+I)
   10 IA(I) = IAS(LL+I)
      DO 20 K=1,NCOL
      LA(K) = LAS(K,J)
      XLB(K) = XLBS(K,J)
   20 XUB(K) = XUBS(K,J)
      LA(NCOL+1) = NELEM + 1
      B(1) = 0.
      LL = NSOL(J) + 1
      DO 30 I=1,NLINK
      K = IESTAR(I,LL,J)
   30 B(I+1) = FLOAT(K)
      K = NLINK + 1
```

```
C              ROUND BIJ'S (IF NECESSARY) TO INTEGER VALUES
C
      LL = NROW + 2
      KK = NLINK + 1
      NROUND = 0
      DO 600 I=1,NLINK
      DSUM = DFPART(I+1) + 0.01
      DO 510 J=LL,NCOL,KK
  510 DSUM = DSUM + DFPART(J)
      ISUM = IDINT(DSUM)
      IF (ISUM .EQ. 0) GO TO 560
      NROUND = 1
      DO 550 II=1,ISUM
      ICOL = I + 1
      DP = DFPART(ICOL)
      DO 520 J=LL,NCOL,KK
      IF (DFPART(J) .LE. DP) GO TO 520
      DP = DFPART(J)
      ICOL = J
  520 CONTINUE
      IPART(ICOL) = IPART(ICOL) + 1
      DFPART(ICOL) = 0.
  550 CONTINUE
  560 LL = LL + 1
  600 CONTINUE
C
C              INSTALL NEW INCUMBENT
      DO 650 J=1,NCOL
  650 INCUMB(J) = IPART(J)
      RETURN
      END
      SUBROUTINE COMCHK(J,K)
C
C         COMPARES CURRENT ALLOCATION (FROM SUBROUTINE ALOCTE) WITH
C         PREVIOUSLY EXAMINED ALLOCATIONS IN EACH UNFIXED SUBPROBLEM
C         TO DETERMINE IF OPT. SOL. CORRESPONDING TO CURRENT ALLOCATION
C         IS ALREADY KNOWN FOR ANY SUBPROBLEM
C********DESCRIPTION OF PARAMETERS********
C         J - PARAMETER INDEXING FIRST SUBPROBLEM FOUND FOR WHICH OPTIMAL
C             SOLUTION CORRESPONDING TO CURRENT ALLOCATION IS ALREADY
C             KNOWN (OUTPUT)
C         K = PARAMETER INDEXING THE PREVIOUSLY EXAMINED ALLOCATION IN
C             SUBPROBLEM J WHICH LEADS TO SAME OPT. SOL. AS WOULD
C         CURRENT ALLOCATION (OUTPUT)
C*************************************************
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 IZSTAR,IELB,IBBND,IZBND
      COMMON/BLOCK3/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
     1IIZINC,IZBAR,IBUM,IELL,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
     2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
     3IZSTAR(40,10),IBIB(2,200),IBBND(200),IZBND(200)
      DO 1000 J=1,NSUBS
      IF (JFIX(J) .GT. 0) GO TO 1000
      IF (NSOL(J) .EQ. 0) GO TO 1000
      L = NSOL(J)
      LP1 = L + 1
      DO 500 K=1,L
C             TEST ALLOCATION AGAINST UPPER BOUNDS OF
C             BIJ IN KTH SOL. OF SUB J
```

```
2196.            ITEMP = ITEMP + N
2197.          500 CONTINUE
2198.            LA(NCOL+1) = NELEM + 1
2199.            ITSINV = 99999
2200.            INVERQ = NREW
2201.            RETURN
2202.            END
2203.            SUBROUTINE TEST2
2204.      C
2205.      C        TESTS LP-OPTIMAL SOLUTION TO MASTER PROBLEM FOR FATHOMING AT
2206.      C        CURRENT NODE IN ALLOCATION-VARIABLES SEARCH TREE.  IF NOT
2207.      C        FATHOMED, ROUNDS LP-OPTIMAL ALLOCATION TO OBTAIN AN INTEGER-
2208.      C        FEASIBLE ALLOCATION.
2209.      C
2210.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
2211.           1          INTEGER*4 (I-N,Q)
2212.            INTEGER*2 DP,KINBAS,LA,LE,IA,IE
2213.            INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
2214.            INTEGER*2 IZSTAR,IBID,IBBND,IZBND
2215.            DOUBLE PRECISION E(1000)
2216.            REAL A(500)
2217.            COMMON/BLOCK1/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
2218.           1               QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
2219.           2               QE,QP,QL,QC,QR,QM,QG,NTMAX
2220.            COMMON/BLOCK2/ DEPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
2221.           1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
2222.           2               IVBND(500),IVID(500),ICBND(500)
2223.            COMMON/BLOCK3/ FLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
2224.           1IZINC,IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
2225.           2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
2226.           3IZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
2227.            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
2228.           1        DY,CC,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
2229.           2        IOBJ,IRLWP,IVIN,IVOUT,ITCNT,INVERQ,ITRERQ,ITSINV,IFEAZ,
2230.           3        JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
2231.           4        IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
2232.            DP = X(1000) + ZTOLZE
2233.            INCVAL = IDINT(DP)
2234.            IF (DP .LT. 0.) INCVAL = INCVAL - 1
2235.            IF (MSTAT .EQ. QN) INCVAL = -10000
2236.      C        TEST FOR FATHOMING
2237.            IF (INCVAL .LE. IZINC) RETURN
2238.      C
2239.      C              COMPUTE INTEGER AND FRACTIONAL PARTS OF EACH ZJ, BIJ
2240.      C
2241.            DO 100 J=1,NCOL
2242.            IF (KINBAS(J)) 40,50,60
2243.         40 DEPART(J) = 0.
2244.            IPART(J) = IDINT(XLB(J))
2245.            GO TO 100
2246.         50 DEPART(J) = 0.
2247.            IPART(J) = IDINT(XLB(J))
2248.            GO TO 100
2249.         60 I = KINBAS(J)
2250.            DP = X(I) + ZTOLZE
2251.            IPART(J) = IDINT(DP)
2252.            IF (DP .LT. 0.) IPART(J) = IPART(J) - 1
2253.            K = IPART(J)
2254.            DEPART(J) = X(I) - FLOAT(K)
2255.          100 CONTINUE
2256.      C
```

```
      SIZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
      COMMON C,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,CC,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IRCWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFREZ,
     3       JCOLP,NROWM,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IC(1000),IA(500),LC(252),LA(122),KINBAS(122),JH(60)
      NROW = NLINK + 1
      B(1) = 0.
C          INSTALL RHS FOR LINKING CONSTRAINTS
      DO 10 I=2,NROW
      ITEMP = IB(I-1)
   10 B(I) = RLOAT(ITEMP)
C                COMPUTE NROW,NCOL
      DO 10C J=1,NSUBS
      NROW = NROW + NLAMDA(J)
  10C CONTINUE
      NCOL = NROW + NSUBS*(NLINK+1)
C
C                INSTALL SLACKS
C
      DO 20C I=1,NROW
      A(I) = 1.
      LA(I) = I
      IA(I) = 1
      KINBAS(I) = 1
      JH(I) = I
      XLB(I) = 0.
  20C XUB(I) = 10000.
C
C                INSTALL ZJ,EJ COLUMNS FOR UNFIXED SUBPROBLEMS
      NELEM = NROW
      ICOL = NROW
      ITEMP = NLINK + 1
      DO 50C J=1,NSUBS
      NELEM = NELEM + 1
      ICOL = ICOL + 1
      A(NELEM) = -1.
      IA(NELEM) = 1
      LA(ICOL) = NELEM
      K = NLAMDA(J)
      DO 31C K=1,K
      NELEM = NELEM + 1
      A(NELEM) = 1.
      II = ITEMP + K
      IA(NELEM) = II
  31C B(II) = DBLE(ZRAUT(K,J))
      II = ITEMP + 1
      KINBAS(ICOL) = II
      KINBAS(II) = J
      JH(II) = ICOL
      DO 35C I=1,NLINK
      NELEM = NELEM + 1
      ICOL = ICOL + 1
      LA(ICOL) = NELEM
      A(NELEM) = 1.
      IA(NELEM) = I + 1
      DO 33C K=1,K
      NELEM = NELEM + 1
      A(NELEM) = -RLAMDA(I,K,J)
  33C IA(NELEM) = ITEMP + K
  35C KINBAS(ICOL) = 0
```

```
2074.            GO TO 8750
2075.       8700 DE = XLB(J)
2076.       8750 LL = IA(J)
2077.            KK = IA(J+1) - 1
2078.            DO 8800 I=LL,KK
2079.            IR = IA(I)
2080.       8800 Y(IR) = Y(IR) - A(I)*DE
2081.       9000 CONTINUE
2082.            CALL FIRAN(1)
2083.            CALL SPIPIR(0,2)
2084.       C         SOLVE MASTER PROBLEM AS LP
2085.          90 CALL NORMAL
2086.       C         TEST FOR FATHOMING; ROUND BIJ'S TO GET INTEGER-FEASIBLE SOL.
2087.            CALL TEST2
2088.       C
2089.       2000 IF (INCVAL .LE. IZINC) RETURN
2090.            ICOL = NROW + 1
2091.       C         COMPUTE LP-BOUND ON OBJ. VAL. IN UNFIXED SUBPROBLEMS FOR USE
2092.       C         IN SUBPROBLEM SELECTION RULE WHEN IPARM = 1
2093.            DO 18 J=1,NSUBS
2094.            IF (JFIX(J) .GT. 0) GO TO 18
2095.            I = KINBAS(ICOL)
2096.            ZJBAR(J) = SNGL(X(I))
2097.         18 ICOL = ICOL + NLINK + 1
2098.       C         INSTALL NEW ALLOCATION IN IBSTAR
2099.            IDIR = NROW + 1
2100.            DO 2100 J=1,NSUBS
2101.            IF (JFIX(J) .GT. 0) GO TO 2100
2102.            LISTL = NSOL(J) + 1
2103.            DO 2050 I=1,NLINK
2104.       2050 IBSTAR(I,LISTL,J) = INCUMB(IDIR+I)
2105.       2100 IDIR = IDIR + NLINK + 1
2106.            DO 2200 I=1,NLINK
2107.            IP1 = I + 1
2108.            IF (INCUMB(IP1) .EQ. 0) GO TO 2200
2109.            DO 2150 J=1,NSUBS
2110.            IF (JFIX(J) .GT. 0) GO TO 2150
2111.            LISTL = NSOL(J) + 1
2112.            IBSTAR(I,LISTL,J) = IBSTAR(I,LISTL,J) + INCUMB(IP1)
2113.            II = IBSTAR(I,LISTL,J) - IBUB(I,J)
2114.            IF (II .LE. 0) GO TO 2200
2115.            IBSTAR(I,LISTL,J) = IBUB(I,J)
2116.            INCUMB(IP1) = II
2117.       2150 CONTINUE
2118.       2200 CONTINUE
2119.            RETURN
2120.            END
2121.            SUBROUTINE LOMSTR
2122.       C
2123.       C         LOADS DATA FOR MASTER PROBLEM INTO APPROPRIATE STORAGE
2124.       C         LOCATIONS FOR SOLUTION BY SUBROUTINE NORMAL
2125.       C
2126.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
2127.           1         INTEGER*4 (I-N,Q)
2128.            INTEGER*2 JM,KINBAS,LA,LE,IA,IE
2129.            INTEGER*2 IBSTAR,IBIU,IBBND,IZBND
2130.            DOUBLE PRECISION E(1000)
2131.            REAL A(500)
2132.            COMMON/BLOCK3/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
2133.           1IZINC,IZBAR,IOUM,IBLL,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
2134.           2NLAMBA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
```

```
C                REOPTIMIZED STARTING FROM PREVIOUSLY OPTIMAL SOLUTION
C                (I.E., DON'T CALL LDMSTR)
C**********************************************
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
      INTEGER*2 IZSTAR,IEID,IBBND,IZBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOLST,ZTOLRJ,ZTOLSM,ORG,OMA,OBA,OFI,
     1              OEC,OBE,OA,OPE,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
     2              OE,OF,OL,OO,OR,OM,OG,NTMAX
      COMMON/BLOCK2/ DPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,KOLST,IFEAS,IPART(122),INCOMB(122),
     2               IVBND(500),IVID(500),ICBND(500)
      COMMON/BLOCK3/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
     1IZINC,IZBAR,IDOM,IBLL,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
     2RLAMDA(110),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
     3IZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
      COMMON B,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C         IF NECESSARY, READ DATA FOR SOLUTION OF MASTER PROBLEM
      IF (IPAR .EQ. 0) CALL LDMSTR
      ICOL = NROW
      DO 50 J=1,NSUBS
      IF (JFIX(J) .GT. 0) GO TO 40
      ICOL = ICOL + 1
C         INSTALL BOUNDS ON ZJ COLUMNS
      XLB(ICOL) = -10000.
      XUB(ICOL) = 10000.
C         INSTALL BOUNDS ON BIJ COLUMNS
      DO 30 I=1,NLINK
      ICOL = ICOL + 1
      II = IBLB(I,J)
      XLB(ICOL) = FLOAT(II)
      II = IBUB(I,J)
   30 XUB(ICOL) = FLOAT(II)
      GO TO 50
C         INSTALL BOUNDS ON COLUMNS ASSOCIATED WITH FIXED SUBPROBLEMS
   40 K = JFIX(J)
      ICOL = ICOL + 1
      II = IZSTAR(K,J)
      XLB(ICOL) = FLOAT(II)
      XUB(ICOL) = XLB(ICOL)
      DO 45 I=1,NLINK
      ICOL = ICOL + 1
      II = IBLB(I,J)
      XLB(ICOL) = FLOAT(II)
   45 XUB(ICOL) = XLB(ICOL)
   50 CONTINUE
      ITCNT = 0
      IF (IPAR .EQ. 0) GO TO 90
      CALL SHIFTR(1,3)
      DO 8000 J=1,NCOL
      IF (KINBAS(J))8000,8700,9000
 8000 DE = XLB(J)
```

```
1952.            XLB(ICOL) = FLOAT(NTEMP(2))
1953.            IF (KINBAS(ICOL) .LT. 0) GO TO 1000
1954.            KINBAS(ICOL) = -1
1955.            NTEMP(3) = 1
1956.      C
1957.       1000 IVID(LISTL) = -IVID(LISTL)
1958.            IVBND(LISTL) = NTEMP(1)
1959.            ICBND(LISTL) = -1000
1960.      C
1961.      C              UPDATE X
1962.            IF (NTEMP(3) .EQ. 0) RETURN
1963.            CALL SHIFTR(1,3)
1964.            DO 9000 J=1,NCOL
1965.            IF (KINBAS(J))8000,8700,9000
1966.       8000 DE = XLB(J)
1967.            GO TO 8750
1968.       8700 DE = XUB(J)
1969.       8750 LL = IA(J)
1970.            KK = IA(J+1) - 1
1971.            DO 8800 I=LL,KK
1972.            IR = JA(I)
1973.       8800 Y(IR) = Y(IR) - A(I)*DE
1974.       9000 CONTINUE
1975.            CALL FTRAN(1)
1976.            CALL SHIFTR(3,2)
1977.            RETURN
1978.      C
1979.      C         NODE FATHOMED:  UPDATE VAR. BOUNDS AND BACKTRACK AGAIN
1980.       2000 ICOL = IVID(LISTL)
1981.            IF (ICOL .LT. 0) GO TO 2100
1982.      C
1983.            NTEMP(1) = IVBND(LISTL)
1984.            IF (KINBAS(ICOL)) 2010,2050,2050
1985.       2010 NTEMP(3) = 1
1986.            DP = XUB(ICOL) - XLB(ICOL)
1987.            DY = FLOAT(NTEMP(1)) - XUB(ICOL)
1988.            IF (DP .LT. DY) KINBAS(ICOL) = 0
1989.       2050 XUB(ICOL) = FLOAT(NTEMP(1))
1990.            GO TO 3000
1991.      C
1992.       2100 ICOL = -ICOL
1993.            NTEMP(1) = IVBND(LISTL)
1994.            IF (KINBAS(ICOL)) 2150,2110,2150
1995.       2110 NTEMP(3) = 1
1996.            DY = XLB(ICOL) - FLOAT(NTEMP(1))
1997.            DP = XUB(ICOL) - XLB(ICOL)
1998.            IF (DP .LT. DY) KINBAS(ICOL) = -1
1999.       2150 XLB(ICOL) = FLOAT(NTEMP(1))
2000.      C
2001.       3000 LISTL = LISTL - 1
2002.            GO TO 50
2003.      C
2004.            END
2005.            SUBROUTINE ALLOTE(IPAR)
2006.      C
2007.      C         SOLVES MASTER PROBLEM;  REALLOCATES LINKING RESOURCE AMONG
2008.      C         UNFIXED SUBPROBLEMS
2009.      C********DESCRIPTION OF PARAMETERS********
2010.      C         IPAR = PARAMETER INDICATING WHETHER OR NOT ANY SUBPROBLEM HAS
2011.      C                BEEN EXPLICITLY SOLVED SINCE LAST MASTER PROBLEM
2012.      C                SOLUTION.  IF NOT (IPAR=1), MASTER PROBLEM CAN BE
```

```
1891.      C
1892.      C              ICOL INDEXES BRANCHING VARIABLE CHOSEN
1893.      C              IDIR INDICATES BRANCHING DIRECTION CHOSEN
1894.      C
1895.      C              ADD OPPOSITE DIRECTION TO LIST
1896.            LISTL = LISTL + 1
1897.            IF (IDIR .EQ. -1) IVBND(LISTL) = IDINT(XLB(ICOL) + ZTOLZE)
1898.            IF (IDIR .EQ. 1) IVBND(LISTL) = IDINT(XUB(ICOL) + ZTOLZE)
1899.            IVID(LISTL) = IDIR*ICOL
1900.            ICBND(LISTL) = IVAL
1901.      C
1902.      C              REVISE BOUNDS ON BRANCHING VARIABLE FOR FORWARD DIRECTION
1903.            IF (IDIR .EQ. -1) XLB(ICOL) = DBLE(REVBND)
1904.            IF (IDIR .EQ. 1) XUB(ICOL) = DBLE(REVBND)
1905.            RETURN
1906.            END
1907.            SUBROUTINE BKTRAK
1908.      C
1909.      C        BACKTRACKS TO SELECT A PROMISING (UNFATHOMED) NODE FROM THE
1910.      C        LIST OF STORED NODES.  LAST-IN-FIRST-OUT (LIFO) SELECTION RULE
1911.      C        IS EMPLOYED.
1912.      C
1913.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
1914.           1         INTEGER*4 (I-N,Q)
1915.            INTEGER*2 JF,KINBAS,LA,LE,IA,IE
1916.            INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
1917.            DOUBLE PRECISION B(1000)
1918.            REAL A(900)
1919.            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
1920.           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QO,NRMAX,NEMAX,QB,QC,
1921.           2              QE,QP,QL,QD,QR,QM,QG,NTMAX
1922.            COMMON/BLOCK2/ DEPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
1923.           1               NPIVCT,IPTYPE,RCCST,IFEAS,IPART(122),INCOMB(122),
1924.           2               IVBND(500),IVID(500),ICBND(500)
1925.            COMMON B,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
1926.           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
1927.           2       IOBJ,IRCWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
1928.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
1929.           4       IE(1000),IA(900),LE(252),LA(122),KINBAS(122),JH(60)
1930.            NTEMP(3) = 0
1931.      C        IF LIST IS EMPTY, RETURN (COMPUTATIONS COMPLETED)
1932.         50 IF (LISTL .EQ. 0) RETURN
1933.            IF (ICBND(LISTL) .LE. INCVAL) GO TO 2000
1934.      C
1935.      C        GET NEXT NODE FROM LIST
1936.            ICOL = IVID(LISTL)
1937.            IF (ICOL .LT. 0) GO TO 100
1938.      C
1939.            NTEMP(1) = IDINT(XLB(ICOL) + ZTOLZE)
1940.            NTEMP(2) = IVBND(LISTL)
1941.            XLB(ICOL) = XUB(ICOL) + 1.
1942.            XUB(ICOL) = FLOAT(NTEMP(2))
1943.            IF (KINBAS(ICOL) .GT. 0) GO TO 1000
1944.            KINBAS(ICOL) = 0
1945.            NTEMP(3) = 1
1946.            GO TO 1000
1947.      C
1948.        100 ICOL = -ICOL
1949.            NTEMP(1) = IDINT(XLB(ICOL) + ZTOLZE)
1950.            NTEMP(2) = IVBND(LISTL)
1951.            XUB(ICOL) = XLB(ICOL) - 1.
```

```
      C        DETERMINE BASIC VAR. X(IROWP) WITH MAX. UP- OR DOWN-PENALTY
            DO 4900 I=1,NROW
            IF (JF(I) .LE. NROW) GO TO 4900
            IF (PL(I) .GT. PU(I)) GO TO 4800
            IF (PL(I) .LE. PEN) GO TO 4900
            PEN = PU(I)
            IROWP = I
            IDIR = -1
            NTEMP(1) = IPART(I) + 1
            REVBND = FLOAT(NTEMP(1))
            GO TO 4900
       4800 IF (PU(I) .LE. PEN) GO TO 4900
            PEN = PU(I)
            IROWP = I
            IDIR = 1
            NTEMP(1) = IPART(I)
            REVBND = FLOAT(NTEMP(1))
       4900 CONTINUE
            IF (IROWP .GT. 0) GO TO 4950
      C        EACH UP- AND DOWN-PENALTY .EQ. 0. (DUAL-DEGENERACY).  CHOOSE
      C        ANY NONINTEGER BASIC VAR. AS BRANCHING VAR.
            DO 4910 IROWP=1,NROW
            IF (JF(IROWP) .LE. NROW) GO TO 4910
            IF (DPART(IROWP) .GE. ZTOLZE) GO TO 4920
       4910 CONTINUE
       4920 PEN = PU(IROWP)
            IDIR = 1
            NTEMP(1) = IPART(IROWP)
            REVBND = FLOAT(NTEMP(1))
       4950 ICOL = JH(IROWP)
            DP = X(IOBJ) - PEN + ZTOLZE
            NTEMP(1) = IDINT(DP)
            IF (DP .LT. 0.) NTEMP(1) = NTEMP(1) - 1
            IF (IVAL .GT. NTEMP(1)) IVAL = NTEMP(1)
      C        BRANCH ON CHOSEN VAR.
            CALL BRANCH
       5000 IF (IDIR .EQ. -1) IPTYPE = 0
            IF (IDIR .EQ. 1) IPTYPE = -1
            RETURN
            END
            SUBROUTINE BRANCH
      C
      C        BRANCHES ON VARIABLE X(ICOL) AS DETERMINED IN SUBROUTINE PENALTS
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
            DOUBLE PRECISION E(1000)
            REAL A(500)
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,ORO,OMA,OBA,OFL,
           1              OEO,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
           2              OE,OF,OL,OO,OR,OM,OG,NTMAX
            COMMON/BLOCK2/ DPPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
           1               NPIVCT,IPTYPE,ROCST,IFEAS,IPART(122),INCOMB(122),
           2               IVBND(500),IVID(500),ICBND(500)
            COMMON E,XLE(122),XLB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVERQ,ITRERQ,ITSINV,IFEAZ,
           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
```

```
      300 DP = (ABS(Y(I))
          NTEMP(1) = IDINT(DP)
          DP = DP - FLOAT(NTEMP(1))
          IF ((DP .LE. ZTOLZE) .OR. (DP .GE. 1.-ZTOLZE)) GO TO 500
          IF (Y(I) .LT. 0.) DP = 1. - DP
          IF (DP .GT. DPART(I)) GO TO 310
          DE = Y(IOBJ)*DPART(I)/DP
          GO TO 320
      310 DE = Y(IOBJ)*(1. - DPART(I))/(1. - DP)
      320 IF (DE .LT. PG(I)) PG(I) = DE
C
      400 CONTINUE
     1000 CONTINUE
C
C                 COMPUTE LARGEST SUMORY PENALTY AND TEST FOR FATHOMING
          PEN = 0.
          DO 2000 I=1,NROW
          IF (JF(I) .EQ. NROW) GO TO 2000
          IF (PG(I) .GT. PEN) PEN = PG(I)
     2000 CONTINUE
          DP = X(IOBJ) - PEN + ZTOLZE
          IVAL = IDINT(DP)
          IF (DP .LT. 0.) IVAL = IVAL - 1
          IF (IVAL .GT. INCVAL) GO TO 3000
          IDIR = 0
          RETURN
C
C                 PROBLEM NOT FATHOMED: CHECK FOR FORCED BRANCHES ON X(I)
     3000 NTEMP(2) = 0
          DO 3900 I=1,NROW
          IF (JF(I) .EQ. NROW) GO TO 3900
          IF (PG(I) .GT. PU(I)) GO TO 3800
          DP = X(IOBJ) - PU(I) + ZTOLZE
          NTEMP(1) = IDINT(DP)
          IF (DP .LT. 0.) NTEMP(1) = NTEMP(1) - 1
          IF (NTEMP(1) .GT. INCVAL) GO TO 3900
C                 FORCED BRANCH UP ON X(I)
          IVAL = NTEMP(1)
          IDIR = -1
          NTEMP(1) = IPART(I) + 1
          GO TO 3700
C
     3800 DP = X(IOBJ) - PU(I) + ZTOLZE
          NTEMP(1) = IDINT(DP)
          IF (DP .LT. 0.) NTEMP(1) = NTEMP(1) - 1
          IF (NTEMP(1) .GT. INCVAL) GO TO 3900
C                 FORCED BRANCH DOWN ON X(I)
          IVAL = NTEMP(1)
          IDIR = 1
          NTEMP(1) = IPART(I)
     3700 IROWP = I
          ICOL = JH(IROWP)
          REVBND = FLOAT(NTEMP(1))
          NTEMP(2) = 1
          CALL BRANCH
     3900 CONTINUE
          IF (NTEMP(2) .GT. 0) GO TO 5000
C
C                 NO FORCED BRANCHES: CHOOSE BRANCHING VAR. AND DIRECTION
          PEN = 0.
          IROWP = 0
```

```
1708.           1                 GEC,GBC,GA,GPE,GMI,GZ,GI,GF,GN,GO,NRMAX,NEMAX,GB,GC,
1709.           2                 GE,GF,GL,GC,GR,GM,GG,NTMAX
1710.            COMMON/BLOCK2/ DPPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
1711.           1                NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCJMB(122),
1712.           2                IVBND(500),IVID(500),ICBND(500)
1713.            COMMON B,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),USUM,DPROD,
1714.           1       DY,DC,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
1715.           2       ICBJ,IRCAP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
1716.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
1717.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
1718.            DO 10 I=1,NROW
1719.            IF (DPPART(I) .LT. ZTOLZE) GO TO 5
1720.            PO(I) = 1.E6
1721.            PD(I) = 1.E6
1722.            PG(I) = 1.E6
1723.            GO TO 10
1724.          5 PO(I) = 0.
1725.            PD(I) = 0.
1726.            PG(I) = 0.
1727.         10 CONTINUE
1728.      C
1729.            DO 1000 J=1,NCOL
1730.            IF (KINBAS(J) .GT. 0) GO TO 1000
1731.            IF ((XUB(J) - XLB(J)) .LE. ZTOLZE) GO TO 1000
1732.            CALL UNPACK(J)
1733.            CALL FTRAN(1)
1734.            IF (KINBAS(J) .EQ. 0) GO TO 30
1735.            DO 20 I=1,NROW
1736.         20 Y(I) = -Y(I)
1737.      C
1738.      C                  CHECK FOR FORCED BRANCH ON XJ
1739.         30 IF (J .LE. NROW) GO TO 50
1740.            DP = X(ICBJ) - Y(ICBJ) + ZTOLZE
1741.            IVAL = IDINT(DP)
1742.            IF (DP .LT. 0.) IVAL = IVAL - 1
1743.            IF (IVAL .GT. INCVAL) GO TO 50
1744.            IDIR = 2*KINBAS(J) + 1
1745.            IF (IDIR .EQ. -1) REVBND = SNGL(XUB(J))
1746.            IF (IDIR .EQ. 1) REVBND = SNGL(XLB(J))
1747.            ICOL = J
1748.            CALL BRANCH
1749.            GO TO 1000
1750.      C
1751.         50 DO 500 I=1,NROW
1752.            IF (JH(I) .LE. NROW) GO TO 500
1753.            IF (DPPART(I) .LT. ZTOLZE) GO TO 500
1754.      C
1755.      C                  COMPUTE UP PENALTY FOR X(I), XJ
1756.        100 IF (Y(I) .GT. -ZTOLPV) GO TO 200
1757.            DE = Y(ICBJ)*(DPPART(I) - 1.)/Y(I)
1758.            IF (DE .LT. Y(ICBJ)) DE = Y(ICBJ)
1759.            IF (DE .LT. PO(I)) PO(I) = DE
1760.            GO TO 300
1761.      C
1762.      C                  COMPUTE DOWN PENALTY FOR X(I), XJ
1763.        200 IF (Y(I) .LT. ZTOLPV) GO TO 300
1764.            DE = Y(ICBJ)*DPPART(I)/Y(I)
1765.            IF (DE .LT. Y(ICBJ)) DE = Y(ICBJ)
1766.            IF (DE .LT. PD(I)) PD(I) = DE
1767.      C
1768.      C                  COMPUTE GOMORY PENALTY FOR X(I), XJ
```

```
      IVAL = IDINT(DP)
      IF (DP .LT. 0.) IVAL = IVAL - 1
      IF ((MSTAT .EQ. 3) .OR. (IVAL .LE. INCVAL)) GO TO 2000
C
C           COMPUTE INTEGER AND FRACTIONAL PARTS OF EACH BASIC VAR.
C
      DO 100 I=1,NROW
      IPART(I) = IDINT(X(I) + ZTOLZE)
      NTEMP(I) = IPART(I)
  100 DFPART(I) = X(I) - FLOAT(NTEMP(I))
C
C           CHECK FOR ALL-INTEGER SOLUTION
C
      DO 200 I=1,NROW
      IF (JH(I) .LE. NROW) GO TO 200
      IF (DFPART(I) .GE. ZTOLZE) RETURN
  200 CONTINUE
C
C           SOLUTION ALL-INTEGER: INSTALL AS NEW INCUMBENT
C
      IFEAS = 1
      INCVAL = IVAL
      DO 1000 J=1,NCOL
      IF (KINBAS(J))600,700,1000
  600 INCLMB(J) = IDINT(XUB(J))
      GO TO 1000
  700 INCLMB(J) = IDINT(XLB(J))
 1000 CONTINUE
C
      DO 1100 I=1,NROW
      ICOL = JH(I)
      INCLMB(ICOL) = IPART(I)
 1100 CONTINUE
C
C           CURRENT PROBLEM NO LONGER OF INTEREST
C
 2000 MSTAT = 1
      RETURN
      END
      SUBROUTINE PENLTS
C
C        COMPUTES TOMLIN'S IMPROVED UP- AND DOWN-PENALTIES AND THE
C        DOMORY PENALTY FOR EACH NONINTEGER BASIC VARIABLE.  ALSO CHECKS
C        FOR FORCED BRANCHES ON BOTH BASIC AND NONBASIC VARIABLES.  IN
C        THE ABSENCE OF FORCED BRANCHES ON BASIC VARIABLES, THE
C        BRANCHING VARIABLE IS CHOSEN TO BE THE ONE WITH LARGEST
C        ASSOCIATED UP- OR DOWN-PENALTY.  THE FORWARD BRANCH IS MADE IN
C        THE DIRECTION OPPOSITE TO THE MAXIMUM PENALTY.  THE NODE
C        CORRESPONDING TO THE BRANCH IN THE SAME DIRECTION AS THE
C        MAXIMUM PENALTY IS ADDED TO THE LIST, TO BE EXAMINED LATER.
C        THE BRANCHING PROCESS ITSELF IS CARRIED OUT IN SUBROUTINE
C        BRANCH, WHICH IS CALLED FROM SUBROUTINE PENLTS.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCLMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(1000)
      REAL A(900)
      REAL PU(60),PD(60),PG(60)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOLST,ZTOLRJ,ZTOLSM,ERU,QMA,QBA,QFI,
```

```
      CALL UNPACK(JCOLP)
      CALL FTRAN(1)
      ICOL = JH(IROWP)
      DP = (X(IROWP) - XLB(ICOL))/Y(IROWP)
      GO TO 2000
C
C           LEAVING VARIABLE EXCEEDS ITS UPPER BOUND
C
 1000 DP = 1.E10
      DO 1500 J=1,NCOL
      IF (KINBAS(J) .GT. 0) GO TO 1500
      IF ((XUB(J) - XLB(J)) .LE. ZTOLZE) GO TO 1500
      CALL UNPACK(J)
      CALL FTRAN(1)
      IF (KINBAS(J) .EQ. -1) GO TO 1200
      IF (Y(IROWP) - ZTOLPV) 1500,1225,1225
 1200 IF (Y(IROWP) + ZTOLPV) 1225,1225,1500
C
 1225 DE = X(IOBJ)/Y(IROWP)
      IF (DE - DP) 1250,1500,1500
 1250 JCOLP = J
      DP = DE
 1500 CONTINUE
C
      IF (JCOLP .EQ. 0) RETURN
      CALL UNPACK(JCOLP)
      CALL FTRAN(1)
      ICOL = JH(IROWP)
      DP = (X(IROWP) - XUB(ICOL))/Y(IROWP)
C
 2000 IF (KINBAS(JCOLP) .EQ. 0) DE = DP + XLB(JCOLP)
      IF (KINBAS(JCOLP) .EQ. -1) DE = DP + XUB(JCOLP)
      NPIVOT = 1
      RETURN
      END
      SUBROUTINE TESTX
C        TESTS LP-OPTIMAL SOLUTION AT CURRENT NODE FOR FATHOMING.
C        FATHOMING OCCURS IF
C        (1) LP PROBLEM AT CURRENT NODE IS INFEASIBLE (MSTAT = 2); OR
C        (2) LP-OPTIMAL OBJECTIVE VALUE (IVAL) .LE. OBJECTIVE VALUE OF
C            CURRENT INCUMBENT SOLUTION (INCVAL); OR
C        (3) LP-OPTIMAL SOLUTION SATISFIES INTEGER RESTRICTIONS.
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JP,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCUMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOLST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBI,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QF,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ OFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCUMB(122),
     2               IVBND(500),IVID(500),ICBND(500)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAS,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      DP = X(IOBJ) + ZTOLZE
```

```
      C         BASIC VARIABLE IN ROW I FALLS BELOW ITS LOWER BOUND
  100 DE = XLB(ICOL) - X(I)
          IF (DE .LE. DP) GO TO 1000
          IPTYPE = 0
          GO TO 250
      C
      C         BASIC VARIABLE IN ROW I EXCEEDS ITS UPPER BOUND
  200 DE = X(I) - XUB(ICOL)
          IF (DE .LE. DP) GO TO 1000
          IPTYPE = -1
      C
  250 IROWP = I
 1000 CONTINUE
          RETURN
          END
          SUBROUTINE LCHOZO
      C
      C         SELECTS PIVOT COLUMN JCOLP FOR CURRENT DUAL-SIMPLEX ITERATION.
      C         SETS JCOLP = 0 IF LP-PROBLEM AT CURRENT NODE IS INFEASIBLE.
      C         OTHERWISE, CHOOSES JCOLP TO MAINTAIN PRIMAL-OPTIMALITY
      C
          IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
         1          INTEGER*4 (I-N,Q)
          INTEGER*2 JH,KINBAS,LA,LE,IA,IE
          INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
          DOUBLE PRECISION E(1000)
          REAL A(500)
          COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRD,QMA,QBA,QFI,
         1              QEQ,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
         2              QE,QH,QL,QD,QR,QM,QG,NTMAX
          COMMON/BLOCK2/ DPPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
         1               NPIVCT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
         2               IVBND(500),IVID(500),ICBND(500)
          COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
         1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
         2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAZ,
         3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
         4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      C
          JCOLP = 0
          IF (IPTYPE .EQ. -1) GO TO 1000
      C
      C              LEAVING VARIABLE FALLS BELOW ITS LOWER BOUND
      C
          DP = -1.E10
          DO 500 J=1,NCOL
          IF (KINBAS(J) .GT. 0) GO TO 500
          IF ((XUB(J) - XLB(J)) .LE. ZTOLZE) GO TO 500
          CALL UNPACK(J)
          CALL FTRAN(1)
          IF (KINBAS(J) .EQ. -1) GO TO 200
          IF (Y(IROWP) + ZTOLPV) 225,225,500
  200 IF (Y(IROWP) - ZTOLPV) 500,225,225
      C
  225 DE = Y(IOBJ)/Y(IROWP)
          IF (DE - DP) 500,500,250
  250 JCOLP = J
          DP = DE
  500 CONTINUE
      C
          IF (JCOLP .EQ. 0) RETURN
```

```
      100 CALL INVERT
          ITSINV = 0
C
C              DUAL SIMPLEX CYCLE
C
C         CHOOSE PIVOT ROW IROWP
      300 CALL DCHOZR
          IF (IROWP .GT. 0) GO TO 400
          MSTAT = OBL
          GO TO 100
C         CHOOSE PIVOT COLUMN JCOLP
      400 CALL DCHOZC
          IF (JCOLP .EQ. 0) GO TO 150
          IVIN = JCOLP
C         UPDATE RIGHT-HAND SIDES TO REFLECT NEW BASIS RESULTING FROM
C         CURRENT SIMPLEX PIVOT
          CALL UPDETA
          ITCNT = ITCNT + 1
          ITSINV = ITSINV + 1
C         DECIDE WHETHER TO REINVERT CURRENT BASIS
          IF ((NELEM .GT. 5000) .OR. (ITSINV .GE. INVFRQ)) GO TO 1000
C         REINVERSION NOT NECESSARY YET; WRITE OUT NEW ETA-VECTOR FOR
C         CURRENT SIMPLEX PIVOT
          CALL WRETA
          GO TO 300
          END
          SUBROUTINE DCHOZR
C
C         SELECTS PIVOT ROW IROWP FOR CURRENT DUAL-SIMPLEX ITERATION.
C         SETS IROWP = 0 IF CURRENT BASIS IS OPTIMAL.  OTHERWISE, IROWP
C         IS CHOSEN TO BE THE ROW WITH GREATEST PRIMAL-INFEASIBILITY
C
          IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
         1          INTEGER*4 (I-N,Q)
          INTEGER*2 JH,KINBAS,LA,LE,IA,IE
          INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
          DOUBLE PRECISION E(1000)
          REAL A(500)
          COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,ORG,OMA,OBA,OFI,
         1              OEO,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
         2              OE,OF,OL,OD,OR,OM,OG,NTMAX
          COMMON/BLOCK2/ DEPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
         1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
         2               IVBND(500),IVID(500),ICBND(500)
          COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
         1        DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
         2        IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
         3        JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
         4        IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
C              CHOOSE ROW WITH GREATEST INFEASIBILITY
C
          IROWP = 0
          DP = -1.E10
          DO 100 I=1,NROW
          IF (I .EQ. IOBJ) GO TO 1000
          ICOL = JH(I)
          IF (X(I) .LT. (XLB(ICOL) - ZTOLZE)) GO TO 100
          IF (X(I) .GT. (XUB(ICOL) + ZTOLZE)) GO TO 200
          GO TO 1000
C
```

```
      GO TO 6000
 3000 IVIN = JCOLP
      CALL UNPACK(JCOLP)
      CALL FTRAN(1)
      CALL CHUZR
      CALL UPBETA
      ITCNT = ITCNT + 1
      ITSINV = ITSINV + 1
      IF (NPIVOT .EQ. 0) GO TO 4010
      IF (NELEM .GT. 5000) GO TO 1000
      CALL WRETA
 4010 IF (ITSINV .GE. INVFRQ) GO TO 1000
      IF (ITCNT .GE. ITRFRQ) GO TO 6000
      GO TO 1500
C
 6000 RETURN
      END
      SUBROUTINE BANDB(INITBD)
C
C        MASTER PROGRAM FOR BRANCH-AND-BOUND INTEGER PROGRAMMING ROUTINE
C        ALSO SERVES AS MASTER PROGRAM FOR REOPTIMIZATION VIA DUAL-
C        SIMPLEX METHOD AFTER A FORWARD BRANCH,
C********DESCRIPTION OF PARAMETERS********
C        INITBD = INITIAL LOWER BOUND ON MAXIMAL OBJECTIVE VALUE (INPUT)
C*****************************************
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QF,QL,QO,QR,QM,QG,NTMAX
      COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
     2               IVBND(500),IVID(500),ICBND(500)
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAS,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      LISTL = 0
      INCVAL = INITBD
C        TEST FOR FATHOMING
  100 CALL TESTA
      IF (MSTAT .EQ. QBL) GO TO 200
C        CURRENT NODE FATHOMED; BACKTRACK TO LAST PROMISING NODE ON LIST
  150 CALL BKTRAK
C        IF LIST IS EMPTY, RETURN TO MAIN (COMPUTATIONS COMPLETED)
      IF (LISTL .EQ. 0) RETURN
C        USE PRIMAL-SIMPLEX METHOD FOR REOPTIMIZATION AT NEW NODE
      CALL NORMAL
      GO TO 100
C        CURRENT NODE NOT FATHOMED; COMPUTE PENALTIES
C        BRANCHING AT CURRENT NODE IS DONE FROM SUBROUTINE PENLTS.
  200 CALL PENLTS
      IF (IDIR) 400,150,400
C
C        REINVERT CURRENT BASIS
```

```
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,LE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAS,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      DO 1000 I=1,NROW
 1000 X(I) = X(I) - Y(I)*DP
      IF (NPIVOT .EQ. 1) GO TO 2000
      KINBAS(JCOLP) = -(KINBAS(JCOLP) + 1)
      IVOUT = JCOLP
      RETURN
 2000 X(IROWP) = DE
      IVOUT = JH(IROWP)
      KINBAS(JCOLP) = IROWP
      KINBAS(IVOUT) = IPTYPE
      JH(IROWP) = JCOLP
      RETURN
      END
      SUBROUTINE NORMAL
C
C        SERVES AS MASTER PROGRAM FOR LINEAR PROGRAMMING COMPONENT
C        (REVISED PRIMAL-SIMPLEX METHOD).
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QF1,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QO,QR,QM,QG,NTMAX
C
      COMMON/BLOCK2/ DFPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
     2               IVBND(500),IVID(500),ICBND(500)
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,LE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAS,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      IF (ITSINV .LT. INVFRQ) GO TO 1500
 1000 CALL INVERT
      ITSINV = 0
C
C              SIMPLEX CYCLE
C
 1500 CALL FORMC
      CALL BTRAN
      CALL PRICE
      IF (JCOLP .GT. 0) GO TO 3000
      IF (MSTAT .EQ. QI    ) GO TO 2000
      MSTAT = QBL
      GO TO 6000
 2000 MSTAT = QN
```

```
      C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            DOUBLE PRECISION E(1000)
            REAL A(500)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,ORO,OMA,OBA,OFI,
           1              CEO,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
           2              OE,OH,OL,OO,OR,OM,OG,NTMAX
      C
            COMMON E,XLE(122),XOB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
           2       IOBJ,IRGAP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
           3       JOOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      C
      C               SHIFT IE AND E OF U ELEMENTS
      C
            NF = NEMAX - NUELEM + 1
            INCR = 0
            DO 1000 I = NF,NEMAX
            INCR = INCR + 1
            IE(NLELEM + INCR) = IE(I)
            E(NLELEM + INCR) = E(I)
       1000 CONTINUE
      C
            IDIF = NEMAX - NLELEM - NUELEM
            NF = NTMAX - NUETA + 1
            INCR = 0
            DO 2000 I = NF,NTMAX
            INCR = INCR + 1
            LE(NLETA + INCR) = LE(I) - IDIF
       2000 CONTINUE
            LE(NETA+1) = NELEM + 1
            RETURN
            END
            SUBROUTINE UPBETA
      C
      C         UPDATES RIGHT-HAND SIDES TO REFLECT NEW BASIS RESULTING FROM
      C         CURRENT SIMPLEX PIVOT
      C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
            DOUBLE PRECISION E(1000)
            REAL A(500)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,ORO,OMA,OBA,OFI,
           1              CEO,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
           2              OE,OH,OL,OO,OR,OM,OG,NTMAX
      C
            COMMON/BLOCK2/ DPPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
           1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
           2               IVBND(500),IVID(500),ICBND(500)
      C
```

```
1220.            CALL SHIFTR(1,3)
1221.            DO 8CCC J=1,NCLL
1222.            IF (KINBAS(J)) 86CC,87JC,9CCC
1223.       86CC DE = XUB(J)
1224.            GC TO 8750
1225.       87CC DE = XLB(J)
1226.       875C LL = LA(J)
1227.            KK = LA(J+1) - 1
1228.            DO 88CC I=LL,KK
1229.            IR = IA(I)
1230.       88CC Y(IR) = Y(IR) - A(I)*DE
1231.       9CCC CONTINUE
1232.            CALL FTRAN(1)
1233.            CALL SHIFTR(3,2)
1234.       C
1235.            NCPC = NELEM - NETA
1236.            NSTR = NROW - NSLCK
1237.       C
1238.            RETURN
1239.            END
1240.            SUBROUTINE UNPACK(IV)
1241.       C
1242.       C         EXPANDS COMPRESSED MATRIX COLUMN
1243.       C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
1244.       C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
1245.       C********DESCRIPTION OF PARAMETERS********
1246.       C         IV = PARAMETER INDEXING COLUMN TO BE EXPANDED
1247.       C**************************************
1248.       C
1249.            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
1250.           1         INTEGER*4 (I-N,Q)
1251.            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
1252.            DOUBLE PRECISION E(1000)
1253.            REAL A(500)
1254.       C
1255.            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
1256.           1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
1257.           2              QE,QH,QL,QD,QR,QM,QG,NTMAX
1258.       C
1259.            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
1260.           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
1261.           2       IOBJ,IREWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
1262.           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
1263.           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
1264.       C
1265.            DO 1CC I = 1,NROW
1266.            Y(I) = 0.
1267.         1CC CONTINUE
1268.       C
1269.            LL = LA(IV)
1270.            KK = LA(IV+1) - 1
1271.            DO 2CC I = LL,KK
1272.            IR = IA(I)
1273.            Y(IR) = A(I)
1274.         2CC CONTINUE
1275.       C
1276.            RETURN
1277.            END
1278.            SUBROUTINE SHFTC
1279.       C
1280.       C         SUBROUTINE FOR INVERT
```

```
      NELEM = NELEM + 1
      IE(NELEM) = I
      E(NELEM) = Y(I)
      GO TO 2300
C
C              U ETA ELEMENTS
C
2200  IE(NELAST) = I
      E(NELAST) = Y(I)
      NELAST = NELAST - 1
      NUELEM = NUELEM + 1
2300  CONTINUE
C
      JH(IROWP) = IV
      KINBAS(IV) = IROWP
      NUETA = NUETA + 1
      IE(NELAST) = IROWP
      IF (J .NE. KR1) GO TO 2330
      E(NELAST) = Y(IROWP)
      GO TO 2340
2330  E(NELAST) = 1.
      NETA = NETA + 1
      LE(NETA+1) = NELEM + 1
2340  NUELEM = NUELEM + 1
      LE(NTLAST) = NELAST
      NELAST = NELAST - 1
      NTLAST = NTLAST - 1
C
C           UPDATE ROW COUNTS
C
      DO 2350 I = 1,NROW
      IF (DABS(Y(I)) .LE. ZTOLZE) GO TO 2350
      IF (HREG(I) .GE. 0) GO TO 2350
      HREG(I) = HREG(I) - INCR
      IF (HREG(I) .GE. 0) HREG(I) = -1
2350  CONTINUE
      HREG(IROWP) = 0
3000  CONTINUE
C
C              MERGE L AND U ETAS
C
3500  NLETA = NETA
      NETA = NLETA + NUETA
      NLELEM = NELEM
      NELEM = NLELEM + NUELEM
      IF (NLELEM .EQ. 0) GO TO 3550
      CALL SHFTE
C
C                 INSERT SLACKS FOR DELETED COLUMNS
C
3550  DO 3600 I = 1,NROW
      IF (JH(I) .NE. 0) GO TO 3600
      JH(I) = I
      IROWP = I
      CALL UNPACK(I)
      CALL FTRAN(I)
      CALL PRETA
3600  CONTINUE
C
C              UPDATE X
C
```

```
1098.            NSLCK = NSLCK + 1
1099.       1200 LL = LA(IV)
1100.            KK = LA(IV+1) -1
1101.            IF (KK .GT. LL) GO TO 1300
1102.       1250 IF (ABS(A(LL) - 1.) .LE. ZTOLZE) GO TO 2000
1103.       C
1104.       1300 NUETA = NUETA + 1
1105.            DO 1400 J = LL,KK
1106.            IR = IA(J)
1107.            IF (IR .EQ. I)GO TO 1390
1108.            IE(NELAST) = IR
1109.            E(NELAST) = A(J)
1110.            NELAST = NELAST - 1
1111.            NUELEM = NUELEM + 1
1112.            GO TO 1400
1113.       1390 EP = A(J)
1114.       1400 CONTINUE
1115.            IE(NELAST) = I
1116.            E(NELAST) = EP
1117.            LE(NTLAST) = NELAST
1118.            NELAST = NELAST - 1
1119.            NTLAST = NTLAST - 1
1120.            NUELEM = NUELEM + 1
1121.       2000 CONTINUE
1122.       2050 IF(KR1 .EQ. 0) GO TO 3500
1123.       C
1124.       C          DO L-U DECOMPOSITION OF BUMP
1125.       C
1126.            DO 3000 J = LR1,KR1
1127.            IV = VREG(J)
1128.            CALL UNPACK(IV)
1129.            CALL FTRAN(2)
1130.            IRCWP = 0
1131.            IRCMIN = +999999
1132.            DO 2100 I = 1,NRCW
1133.            IF (DABS(Y(I)) .LE. ZTOLPV) GO TO 2100
1134.            IF (HREG(I) .GE.0) GO TO 2100
1135.            IF (HREG(I) .LE. IRCMIN) GO TO 2100
1136.            IRCMIN = HREG(I)
1137.            IRCWP = I
1138.       2100 CONTINUE
1139.            IF (IRCWP .GT. 0) GO TO 2150
1140.            WRITE(6,5000)
1141.            KINBAS(IV) = 0
1142.            GO TO 3000
1143.       C
1144.       2150 INCR = HREG(IRCWP) + 3
1145.       C
1146.       C             WRITE L AND U ETAS
1147.       C
1148.            IF (J .EQ. KR1) GO TO 2160
1149.            NELEM = NELEM + 1
1150.            IE(NELEM) = IRCWP
1151.            E(NELEM) = Y(IRCWP)
1152.       2160 DO 2300 I = 1,NRCW
1153.            IF (I .EQ. IRCWP) GO TO 2300
1154.            IF(DABS(Y(I)) .LE. ZTOLZE) GO TO 2300
1155.            IF (HREG(I) .GE. 0) GO TO 2200
1156.       C
1157.       C               U ETA ELEMENTS
1158.       C
```

```
1037.            LL = LA(IV)
1038.            KK = LA(IV+1) - 1
1039.            IMCNT = 0
1040.            DO 1050 I = LL,KK
1041.            IR = IA(I)
1042.            IF (MREG(IR) .GE. 0) GO TO 1050
1043.            IMCNT = IMCNT - (MREG(IR) +1)
1044.     1050 CONTINUE
1045.            MREG(J) = IMCNT
1046.     1100 CONTINUE
1047.     C
1048.     C                    SORT COLUMNS INTO MERIT ORDER
1049.     C                    USING SHELL SORT
1050.     C
1051.            ISD = 1
1052.     1106 IF (KR1 .LT. 2*ISD) GO TO 1108
1053.            ISD = 2*ISD
1054.            GO TO 1106
1055.     1108 ISD = ISD - 1
1056.     C                          END OF INITIALIZATION
1057.     1101 IF (ISD .EQ. 0) GO TO 1107
1058.            ISK = 1
1059.     1102 ISJ = ISK
1060.            ISL = ISK + ISD
1061.            ISY = MREG(ISL)
1062.            ISZ = VREG(ISL)
1063.     1103 IF (ISY .LT. MREG(ISJ)) GO TO 1104
1064.     1105 ISL = ISJ + ISD
1065.            MREG(ISL) = ISY
1066.            VREG(ISL) = ISZ
1067.            ISK = ISK + 1
1068.            IF ((ISK + ISD) .LE. KR1) GO TO 1102
1069.            ISD = (ISD - 1) / 2
1070.            GO TO 1101
1071.     1104 ISL = ISJ + ISD
1072.            MREG(ISL) = MREG(ISJ)
1073.            VREG(ISL) = VREG(ISJ)
1074.            ISJ = ISJ - ISD
1075.            IF (ISJ .GT. 0) GO TO 1103
1076.            GO TO 1105
1077.     1107 CONTINUE
1078.     C                              END OF SORT ROUTINE
1079.     C
1080.     C           PUT OUT BELOW BUMP ETAS (PART OF U)
1081.     C
1082.     1150 NSLCK = 0
1083.            NBELOW = 0
1084.            NELAST = NEMAX
1085.            NTLAST = NTMAX
1086.            LE(NTLAST + 1) = NELAST + 1
1087.     C
1088.            LR = LR3
1089.            IF (LR3 .GE. LR4) LR = LR4
1090.            IF (LR .GT. KR4) GO TO 2050
1091.            JK = KR4 + 1
1092.            DO 2000 JJ= LR,KR4
1093.            JK = JK - 1
1094.            IV = VREG(JK)
1095.            I = MREG(JK)
1096.            NBELOW = NBELOW + 1
1097.            IF (IV .GT. NROW) GO TO 1200
```

```
C
C              PIVOT ABOVE BUMP (PART OF L)
C
      NABOVE = NABOVE + 1
      IRCWP = IR
      CALL UNPACK(IV)
      CALL WRETA
      NLETA = NETA
      JH(IR) = IV
      KINBAS(IV) = IR
      VREG(J) = VREG(KR1)
      KR1 = KR1 - 1
      NVREM = NVREM + 1
      HREG(IR) = IV
      GO TO 940
C
  400 IF (HREG(IR) .GE. 0) GO TO 800
      IRCNT = IRCNT + 1
      IRP = IR
  800 CONTINUE
C
      IF (IRCNT - 1) 810,900,1000
  810 WRITE(6,8000)
      KINBAS(IV) = 0
      VREG(J) = VREG(KR1)
      NVREM = NVREM + 1
      KR1 = KR1 - 1
      IF (J .GT. KR1) GO TO 1010
      GO TO 320
C
C                  PUT VECTOR BELOW BUMP
C
  900 VREG(J) = VREG(KR1)
      NVREM = NVREM + 1
      KR1 = KR1 - 1
      LR3 = LR3 - 1
      VREG(LR3) = IV
      MREG(LR3) = IRP
      HREG(IRP) = 0
      JH(IRP) = IV
      KINBAS(IV) = IRP
C
C           CHANGE ROW COUNTS
C
  940 DO 950 II = LL,KK
      IIR = IA(II)
      IF (HREG(IIR) .GE. 0) GO TO 950
      HREG(IIR) = HREG(IIR) + 1
  950 CONTINUE
      IF (J .GT. KR1) GO TO 1010
      GO TO 320
 1000 IF (J .GE. KR1) GO TO 1010
      J = J+1
      GO TO 320
 1010 IF(NVREM .GT. 0) GO TO 310
C
C               GET MERIT COUNTS
C
 1020 IF (KR1 .EQ. 0) GO TO 1190
      DO 1100 J = LR1,KR1
      IV = VREG(J)
```

```
      C
            DO 200 I = LR4,KR4
            IR = MREG(I)
            MREG(IR)= 0
            JH(IR) = IR
            KINBAS(IR) = IR
        200 CONTINUE
      C
      C              PULL OUT VECTORS BELOW BUMP AND GET ROW COUNTS
      C
            NBNCNZ = KR4 - LR4 + 1
            IF (KR1 .EQ. 0) GO TO 1190
            J = LR1
        210 IV = VREG(J)
            LL = LA(IV)
            KK = LA(IV+1) -1
            IRCNT = 0
            DO 220 I = LL,KK
            NBNCNZ = NBNCNZ + 1
            IR = IA(I)
            IF (MREG(IR) .GE. 0) GO TO 220
            IRCNT = IRCNT + 1
            MREG(IR) = MREG(IR) - 1
            IRP = IR
        220 CONTINUE
            IF (IRCNT - 1) 230,250,300
        230 WRITE(6,8000)
       8000 FORMAT(16HOMATRIX SINGULAR )
            KINBAS(IV) = 0
            VREG(J) = VREG(KR1)
            KR1 = KR1 - 1
            IF (J .GT. KR1) GO TO 310
            GO TO 210
      C
        250 VREG(J) = VREG(KR1)
            KR1 = KR1 - 1
            LR3 = LR3 - 1
            VREG(LR3) = IV
            MREG(LR3) = IRP
            MREG(IRP) = 0
            JH(IRP) = IV
            KINBAS(IV) = IRP
            IF (J .GT. KR1) GO TO 310
            GO TO 210
        300 IF (J .GE. KR1) GO TO 310
            J = J+1
            GO TO 210
      C
      C                    PULL OUT REMAINING VECTORS ABOVE AND BELOW THE
      C                    BUMP AND ESTABLISH MERIT COUNTS OF COLUMNS
      C
        310 NVREM = 0
            IF(KR1 .EQ. 0) GO TO 1190
            J = LR1
        320 IV = VREG(J)
            LL = LA(IV)
            KK = LA(IV+1) - 1
            IRCNT = 0
            DO 800 I = LL,KK
            IR = IA(I)
            IF(MREG(IR) .NE. -2) GO TO 400
```

```
      BARRAY(IFN + I) = BARRAY(IFO + I)
  100 CONTINUE
      RETURN
      END
      SUBROUTINE INVERT
C
C        COMPUTES INVERSE OF CURRENT BASIS BY LU DECOMPOSITION
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,ORD,OMA,OBA,OFI,
     1              OEO,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
     2              OE,OF,OL,OO,OR,OM,OG,NIMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IRLWP,IVIN,IVOUT,ITCNT,INVERQ,ITRERQ,ITSINV,IFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      INTEGER*2 MREG,HREG,VREG
      DIMENSION MREG(60),HREG(60),VREG(60)
      EQUIVALENCE (MREG(1),YTEMP(1)),(HREG(1),YTEMP(31)),(VREG(1),X(1))
C
C            SET PARAMETERS
C
      NETA = 0
      NLETA = 0
      NUETA = 0
      NELEM = 0
      NLELEM = 0
      NUELEM = 0
      NABOVE = 0
      LE(1) = 1
      LR1 = 1
      KR1 = 0
      LR4 = NROW + 1
      KR4 = NROW
C
C        PUT SLACKS AND ARTIFICIALS IN PART 4 AND REST IN PART 1
C
      DO 100 I = 1,NROW
      IF (JH(I) .GT. NROW) GO TO 50
      LR4 = LR4 - 1
      MREG(LR4) = JH(I)
      VREG(LR4) = JH(I)
      GO TO 90
   50 KR1 = KR1 + 1
      VREG(KR1) = JH(I)
   90 HREG(I) = -1
      JH(I) = 0
  100 CONTINUE
C
      KR3 = LR4 -1
      LR3 = LR4
```

```
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QG,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,CE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      NELEM = NELEM + 1
      IE(NELEM) = IROWP
      E(NELEM) = Y(IROWP)
C
      DO 1000 I = 1,NROW
      IF (I .EQ. IROWP) GO TO 1000
      IF (DABS(Y(I)) .LE. ZTOLZE) GO TO 1000
      NELEM = NELEM + 1
      IE(NELEM) = I
      E(NELEM) = Y(I)
 1000 CONTINUE
C
      NETA = NETA + 1
      LE(NETA+1) = NELEM + 1
      RETURN
      END
      SUBROUTINE SHIFTR(IOLD,INEW)
C
C        REARRANGES DATA STORAGE
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C********DESCRIPTION OF PARAMETERS********
C        IOLD, INEW = PARAMETERS INDICATING STORAGE LOCATIONS FROM AND
C        TO WHICH DATA IS TO BE TRANSFERRED, RESPECTIVELY
C*****************************************
C
      IMPLICIT REAL*4 (A,C,E-H,C,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QH,QL,QG,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,CE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      DIMENSION BARRAY(240)
      EQUIVALENCE (BARRAY(1),B(1))
      IFC = (IOLD - 1) * NRMAX
      IFN = (INEW - 1) * NRMAX
C
      DO 1000 I = 1,NROW
```

```
      100 IF (X(I) .LT. (XLB(ICOL) - ZTOLZE)) GO TO 500
          DE = (X(I) - XLB(ICOL))/Y(I)
          IF (DE .GE. DP) GO TO 500
          IPTYPE = 0
          GO TO 250
C                 NEGATIVE COEFFICIENT
      200 IF (X(I) .GT. (XUB(ICOL) + ZTOLZE)) GO TO 500
          DE = (X(I) - XUB(ICOL))/Y(I)
          IF (DE .GE. DP) GO TO 500
          IPTYPE = -1
      250 DP = DE
          IROWP = I
      500 CONTINUE
          DE = DP + XLB(JCOLP)
          IF (DE .LT. XUB(JCOLP)) GO TO 600
          DP = XUB(JCOLP) - XLB(JCOLP)
          NPIVOT = 0
          RETURN
      600 NPIVOT = 1
          RETURN
C
C                 INCOMING VARIABLE AT UPPER BOUND
C
     1000 DP = -1.E10
          DO 1500 I=1,NROW
          IF (I .EQ. IOBJ) GO TO 1500
          ICOL = JA(I)
          IF (Y(I) .GT. ZTOLPV) GO TO 1100
          IF (Y(I) .LT. -ZTOLPV) GO TO 1200
          GO TO 1500
C                 POSITIVE COEFFICIENT
     1100 IF (X(I) .GT. (XUB(ICOL) + ZTOLZE)) GO TO 1500
          DE = (X(I) - XUB(ICOL))/Y(I)
          IF (DE .LE. DP) GO TO 1500
          IPTYPE = -1
          GO TO 1250
C                 NEGATIVE COEFFICIENT
     1200 IF (X(I) .LT. (XLB(ICOL) - ZTOLZE)) GO TO 1500
          DE = (X(I) - XLB(ICOL))/Y(I)
          IF (DE .LE. DP) GO TO 1500
          IPTYPE = 0
     1250 DP = DE
          IROWP = I
     1500 CONTINUE
          DE = DP + XUB(JCOLP)
          IF (DE .GT. XLB(JCOLP)) GO TO 1600
          DP = XLB(JCOLP) - XLB(JCOLP)
          NPIVOT = 0
          RETURN
     1600 NPIVOT = 1
          RETURN
          END
          SUBROUTINE WRETA
C
C         FORMS NEW ETA-VECTOR CORRESPONDING TO CURRENT SIMPLEX PIVOT
C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
          IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
         1          INTEGER*4 (I-N,Q)
          INTEGER*2 JA,KINBAS,LA,LE,IA,IE
```

```
      GO TO 1000
  400 IF (QSUM .LE. QMAX) GO TO 1000
      QMAX = QSUM
      JCOL2 = J
 1000 CONTINUE
C
      IF (QMIN .LE. -ZTOLST) GO TO 1500
      IF (QMAX .GE. ZTOLST) GO TO 2000
      JCOLP = 0
      RCOST = 0.
      RETURN
 1500 IF (QMAX .GE. ZTOLST) GO TO 2500
 1600 JCOLP = JCOL1
      RCOST = QMIN
      RETURN
 2000 JCOLP = JCOL2
      RCOST = QMAX
      RETURN
 2500 IF (ABS(QMIN) - QMAX) 2000,2000,1600
      END
      SUBROUTINE CHUCK
C
C        PERFORMS MIN-RATIO TEST FOR PIVOT COLUMN JCOLP DETERMINED IN
C        SUBROUTINE PRICE.  SELECTS PIVOT ROW IROWP FOR CURRENT PRIMAL-
C        SIMPLEX ITERATION.
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JP,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,IOBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOLST,ZTOLRJ,ZTOLSM,ORG,OMA,OBA,OPI,
     1              OEO,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
     2              OE,OF,OL,OC,OR,OM,OG,NTMAX
C
      COMMON/BLOCK2/ DPPART(122),REVORD,INCVAL,IOOL,LISTL,IVAL,IDIR,
     1               NPIVLT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
     2               IVBND(500),IVID(500),IOBND(500)
C
      COMMON B,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DC,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IRLWP,IVIN,IVOUT,ITCNT,INVERQ,ITRERQ,ITSINV,IFEEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      IF (KINBAS(JCOLP) .EQ. -1) GO TO 1000
C
C              INCOMING VARIABLE AT LOWER BOUND
C
      DP = 1.E10
      DO 500 I=1,NROW
      IF (I .EQ. IOBJ) GO TO 500
      ICOL = JH(I)
      IF (Y(I) .GT. ZTOLPV) GO TO 100
      IF (Y(I) .LT. -ZTOLPV) GO TO 200
      GO TO 500
C              POSITIVE COEFFICIENT
```

```
      IF (X(I) .GE. (XUB(ICOL) + ZTOLZE)) GO TO 20
      Y(I) = 0.
      GO TO 30
   10 Y(I) = 1.
      SUMINF = SUMINF + XLB(ICOL) - X(I)
      GO TO 30
   20 Y(I) = -1.
      SUMINF = SUMINF + X(I) - XUB(ICOL)
   30 CONTINUE
      IF (SUMINF .GT. ZTOLSM) RETURN
      Y(ICBJ) = 1.
      IFFEZ = 1
      MSTAT = 4F
      RETURN
      END
      SUBROUTINE PRICE
C
C        PRICES OUT NONBASIC COLUMNS, CHOOSES PIVOT COLUMN JCOLP FOR
C        CURRENT PRIMAL-SIMPLEX ITERATION
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,C,P,R-W,Z), REAL*8 (B,D,X,Y),
     1          INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IPART,INCOMB,IVBND,IVID,ICBND
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QO,NRMAX,NEMAX,QB,QC,
     2              QE,QF,QL,QO,QR,QM,QG,NTMAX
C
      COMMON/BLOCK2/ DEPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
     1               NPIVOT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
     2               IVBND(500),IVID(500),ICBND(50C)
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       OY,DE,LP,A,ICNAM(122,2),NAME(2),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IRCWP,IVIN,IVOUT,ITCNT,INVERQ,ITREQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      CMIN = 1.E10
      CMAX = -1.E10
      DO 1000 J=1,NCOL
      IF (KINBAS(J) .GT. 0) GO TO 1000
      IF ((XUB(J) - XLB(J)) .LT. ZTOLZE) GO TO 1000
      DSUM = 0.
      LL = LA(J)
      KK = LA(J+1) - 1
      DO 500 I = LL,KK
      IR = IA(I)
      DE = A(I)
      DPROD = DE * Y(IR)
      DSUM = DSUM + DPROD
  500 CONTINUE
      IF (KINBAS(J) .EQ. -1) GO TO 600
      IF (DSUM .GE. CMIN) GO TO 1000
      CMIN = DSUM
      JCOL1 = J
```

```
      2              QE,QF,QL,QO,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      IF (NETA .LE. 0) GO TO 9000
      DO 1000 I = 1,NETA
      IK = NETA - I + 1
      LL = LE(IK)
      KK = LE(IK+1) - 1
      IPIV = IE(LL)
      DP = E(LL)
      DY = Y(IPIV)
      DSUM = 0.
      IF (KK .LE. LL) GO TO 600
      LL = LL + 1
      DO 500 J = LL,KK
      IR = IE(J)
      DE = E(J)
      DPROD = DE * Y(IR)
      DSUM = DSUM + DPROD
  500 CONTINUE
C
  600 Y(IPIV) = (DY - DSUM) / DP
 1000 CONTINUE
 9000 RETURN
      END
      SUBROUTINE FORMC
C
C        FORMS COST VECTOR FOR CURRENT PRIMAL-SIMPLEX ITERATION; USED IN
C        CONJUNCTION WITH SUBROUTINE PRICE
C        SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C        BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
      IMPLICIT REAL*4 (A,C,E-H,C,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      DOUBLE PRECISION E(1000)
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
     1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QU,NRMAX,NEMAX,QB,QC,
     2              QE,QF,QL,QO,QR,QM,QG,NTMAX
C
      COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      MSTAT = QI
      IFFEZ = 0
      SUMINF = 0.
      Y(ICBJ) = C.
      DO 30 I=1,NROW
      IF (I .EQ. ICBJ) GO TO 30
      ICOL = JH(I)
      IF (X(I) .LE. (XLB(ICOL) - ZTOLZE)) GO TO 10
```

```
      C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C********DESCRIPTION OF PARAMETERS********
      C         IPAR = PARAMETER INDICATING BY WHICH ETA-VECTORS THE MATRIX
      C         COLUMN IS TO BE UPDATED
      C****************************************
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            DOUBLE PRECISION E(1000)
            REAL A(500)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,ORO,OMA,OBA,OFI,
           1              CEO,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
           2              OE,OH,OL,OO,OR,OM,OG,NTMAX
      C
            COMMON E,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
           2       IOBJ,IRCWP,IVIN,IVOJT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      C
            GO TO (100,110), IPAR
        100 NFE = 1
            NLE = NETA
            GO TO 200
        110 NFE = NLETA + 1
            NLE = NETA
        200 IF (NFE .GT. NLE) GO TO 9000
            DO 1000 IK = NFE,NLE
            LL = LE(IK)
            KK = LE(IK+1) - 1
            IPIV = IE(LL)
            DY = Y(IPIV)
            DY = DY/E(LL)
            Y(IPIV) = DY
            IF (KK .LE. LL) GO TO 1000
            LL = LL + 1
            DO 500 J = LL,KK
            IR = IE(J)
            Y(IR) = Y(IR) - E(J) * DY
        500 CONTINUE
       1000 CONTINUE
       9000 CONTINUE
            RETURN
            END
            SUBROUTINE BTRAN
      C
      C         PERFORMS BACKWARD TRANSFORMATION ON COLUMN STORED IN VECTOR Y
      C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
      C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            DOUBLE PRECISION E(1000)
            REAL A(500)
      C
            COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,ORO,OMA,OBA,OFI,
           1              CEO,OBL,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
```

```
      ICS1 = NAME(1)
      ICS2 = NAME(2)
C
C        TEST FOR ROW MATCH
C
  530 DO 540 I = 1,NROW
      IF(NAME(J) .NE. ICNAM(I,1) .OR. NAME(K) .NE. ICNAM(I,2))GO TO 540
      B(I) = ATEMP1
  535 IF(K .GT. 5)  GO TO 5
      IF(DABS(ATEMP2) .LE. ZTOLZE)  GO TO 5
      J = 5
      K = 6
      ATEMP1 = ATEMP2
      GO TO 530
  540 CONTINUE
      WRITE(6,650) NAME(J),NAME(K)
      STOP
C
C           END OF INPUT
C
  600 NSCOL = NCOL - NROW
      K = NROW + 1
C        INPUT LOWER AND UPPER BOUNDS ON DECISION VARIABLES FOR SUBPROB-
C        LEM INPROB
      READ (5,650) (XLB(J), J=K,NCOL)
      READ (5,650) (XUB(J), J=K,NCOL)
  650 FORMAT (15F5.0)
      XLB(ICBJ) = -10000.
      XUB(ICBJ) = 10000.
C        INPUT LOWER AND UPPER BOUNDS ON ALLOCATION VARIABLE FOR SUB-
C        PROBLEM INPROB
      READ (5,651) (IBLB(I,INPROB), I=1,NLINK)
      READ (5,651) (IBUB(I,INPROB), I=1,NLINK)
  651 FORMAT (8I10)
      INVERC = NROW
C              STORE A,IA,LA,XUB,OLB,B,NCOL,NROW FOR SUBPROBLEM INPROB
      LL = JFIRST(INPROB) - 1
      DO 101C I=1,NELEM
      LLPI = LL + I
      AMATRS(LLPI) = A(I)
 1010 IAS(LLPI) = IA(I)
      DO 102C K=1,NCOL
      LAS(K,INPROB) = LA(K)
      XLBS(K,INPROB) = XLB(K)
 1020 XUBS(K,INPROB) = XUB(K)
      K = NLINK + 1
      LL = NROW - K
      DO 103C I=1,LL
 1030 RHS(I,INPROB) = SNGL(B(K+I))
      NCOLS(INPROB) = NCOL
      NROWS(INPROB) = NROW
      JFIRST(INPROB+1) = JFIRST(INPROB) + NELEM
C
      NELEM = NELEM - NROW
      RELEM = NELEM
      RDENS = RELEM/(NROW*NSCOL)
      RETURN
      END
      SUBROUTINE FTRAN(IPAR)
C
C        PERFORMS  FORWARD TRANSFORMATION ON COLUMN STORED IN VECTOR Y
```

```
C          TEST FOR ROW MATCH
C
  330 DO 340 I = 1,NROW
      IF(NAME(J) .NE. ICNAM(I,1) .OR. NAME(K) .NE. ICNAM(I,2))GO TO 340
      NELEM = NELEM + 1
      IA(NELEM) = I
      A(NELEM) = ATEMP1
      LA(NCOL+1)=NELEM+1
  335 IF(K .GT. 5)  GO TO 5
      IF(DABS(ATEMP2) .LE. ZTOLZE)  GO TO 5
      J = 5
      K = 6
      ATEMP1 = ATEMP2
      GO TO 330
  340 CONTINUE
      WRITE(6,8500) NAME(J),NAME(K),NAME(1),NAME(2)
 8500 FORMAT(18HNO MATCH FOR ROW ,2A4,10HAT COLUMN ,2A4)
      STOP
C
C          BASIS CARDS
C
  410 DO 420 I = 1,NCOL
      IF(NAME(1) .NE. ICNAM(I,1) .OR. NAME(2) .NE. ICNAM(I,2)) GO TO 420
      IBVEC = I
      GO TO 425
  420 CONTINUE
      WRITE(6,8400) NAME(1),NAME(2)
 8400 FORMAT(21HNO MATCH FOR VECTOR ,2A4)
      GO TO 5
  425 DO 430 I = 1,NROW
      IF (NAME(3).NE.ICNAM(I,1).OR.NAME(4).NE.ICNAM(I,2))  GO TO 430
      IBROW=I
      GO TO 440
  430 CONTINUE
      WRITE(6,8500) NAME(3),NAME(4)
      GO TO 5
  440 JA(IBROW) = IBVEC
      KINBAS(IBROW) = 0
      KINBAS(IBVEC) = IBROW
      GO TO 5
C
C          RHS
C
  500 J = 3
      K = 4
      IF (DABS(ATEMP1) .LE. ZTOLZE)  GO TO 521
      GO TO 524
  521 J=5
      K=6
      ATEMP1=ATEMP2
      GO TO 530
  524 CONTINUE
      IF (NAME(1) .EQ. ICS1 .AND. NAME(2) .EQ. ICS2)  GO TO 530
C
C          TEST FOR SPLIT VECTOR
C
      DO 525 I = 1,NCOL
      IF (NAME(1) .NE. ICNAM(I,1))  GO TO 525
      IF (NAME(2) .NE. ICNAM(I,2))  GO TO 525
      WRITE(6,8250) NAME(1),NAME(2)
  525 CONTINUE
```

```
      150 CONTINUE
          GO TO 5
C
      210 NROW=NROW+1
          NCOL=NROW
          ICNAM(NROW,1) = NAME(1)
          ICNAM(NROW,2) = NAME(2)
C
C         TEST ROW TYPE
C
          IF(K2.EQ.GE .OR. K3.EQ.GE)  GO TO 220
          IF(K2.EQ.LE .OR. K3.EQ.LE)  GO TO 230
          IF(K2.EQ.GG .OR. K3.EQ.GG)  GO TO 240
          IF(K2.EQ.EN .OR. K3.EQ.EN)  GO TO 250
          GO TO 260
      220 XLB(NROW) = 0.
          XUB(NROW) = 1.E4
          GO TO 250
      230 XLB(NROW) = 0.
          XUB(NROW) = 0.
          GO TO 250
      240 XLB(NROW) = 0.
          XUB(NROW) = 1.E4
          A(NROW) = -1.
          GO TO 260
      250 A(NROW) = 1.
      260 IA(NROW) = NROW
          LA(NROW) = NROW
          JH(NROW) = NROW
          KINBAS(NROW) = NROW
          NELEM=NROW
          GO TO 5
C
C         MATRIX ELEMENTS
C
      320 J = 3
          K = 4
          IF (DABS(ATEMP1) .LE. ZTOLZE)  GO TO 321
          GO TO 324
      321 J=5
          K=6
          ATEMP1=ATEMP2
          GO TO 330
      324 CONTINUE
          IF (NAME(1) .EQ. ICS1 .AND. NAME(2) .EQ. ICS2)  GO TO 330
C
C         TEST FOR SPLIT VECTOR
C
          DO 325 I = 1,NCOL
          IF (NAME(1) .NE. ICNAM(I,1))  GO TO 325
          IF (NAME(2) .NE. ICNAM(I,2))  GO TO 325
          WRITE(6,3250) NAME(1),NAME(2)
     3250 FORMAT(14HOSPLIT VECTOR ,2A4)
      325 CONTINUE
          NCOL = NCOL + 1
          ICS1 = NAME(1)
          ICS2 = NAME(2)
          ICNAM(NCOL,1) = ICS1
          ICNAM(NCOL,2) = ICS2
          LA(NCOL) = NELEM + 1
C
```

```
C         SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C         BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C********DESCRIPTION OF PARAMETERS********
C         IPPROB = INPROB = IDENTIFICATION NUMBER OF SUBPROBLEM BEING
C         INPUT (USER MUST INPUT SUBPROBLEMS IN SEQUENCE 1,2,...,NSUBS)
C*************************************
C
      IMPLICIT REAL*4 (A-C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
     1         INTEGER*4 (I-N,Q)
      INTEGER*2 JH,KINBAS,LA,LE,IA,IE
      INTEGER*2 IZSTAR,IEIC,IBBND,IZBND
      INTEGER*2 IAS,LAS
      DOUBLE PRECISION E(1000),ATEMP1,ATEMP2
      REAL A(500)
C
      COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTCOST,ZTOLRJ,ZTOLSM,ORO,OMA,OBA,OFI,
     1              OEC,OBC,OA,OPL,OMI,OZ,OI,OF,ON,OU,NRMAX,NEMAX,OB,OC,
     2              OE,OF,OL,OO,OR,OM,OG,ATMAX
C
      COMMON/BLOCK3/ RLAMDA(5,20,10),ZNAUT(20,10),ZJBAR(10),NLINK,NSUBS,
     1IZINC,IZBAR,IDUM,IBLL,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
     2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
     3IZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
      COMMON/BLOCK4/ XLBS(20,10),XUBS(20,10),AMATRS(1000),RHS(10,10),
     1NROWS(10),NCOLS(10),JFIRST(11),IAS(1000),LAS(20,10)
      COMMON B,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
     1        DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
     2        IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFFEZ,
     3        JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
     4        IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
C
      DO 10 I=1,60
   10 B(I) = 0.
      DO 20 J=1,122
   20 KINBAS(J) = 0
      NROW = 0
      ITCNT = 0
      ICS1=0
      ICS2=0
      READ (5,7000) INPROB
 7000 FORMAT (5I4)
      IPPROB = INPROB
      IF (IPPROB .EQ. 0) RETURN
C
    5 READ(5,101) K1,K2,K3,K4,(NAME(I),I=1,4),ATEMP1,NAME(5),NAME(6),
     1ATEMP2
  101 FORMAT(4A1,2A4,2X,2A4,2X,F12.4,3X,2A4,2X,F12.4)
      IF(K1 .EQ. OE) GO TO 600
      IF(K1 .EQ. OEL) GO TO 50
      IF(K1 .EQ. ON) GO TO 100
      IF(K1 .EQ. OR .AND. K2 .EQ. OO) L=1
      IF(K1 .EQ. OR .AND. K2 .EQ. OO) GO TO 150
      IF(K1 .EQ. OC) L=2
      IF(K1 .EQ. OC) GO TO 150
      IF(K1 .EQ. OB .AND. K2 .EQ. OA) L=3
      IF(K1 .EQ. OB .AND. K2 .EQ. OA) GO TO 150
      IF(K1 .EQ. OR .AND. K2 .EQ. OH) L=4
      IF(K1 .EQ. OR .AND. K2 .EQ. OH) GO TO 150
   50 GO TO(210,320,410,500),L
  100 NTEMP(1) = NAME(3)
      NTEMP(2) = NAME(4)
```

```
      3310 AV(J)=((NTSOL(J)-1)*AV(J)+(ZJBAR(J)-INCVAL))/NTSOL(J)
           GO TO 3400
      3320 ISLACK = 0
           NP1 = NLINK + 1
           DO 3325 I=2,NP1
      3325 ISLACK = ISLACK + INCOMB(I)
           AV(J)=((NTSOL(J)-1)*AV(J)+ISLACK)/NTSOL(J)
           GO TO 3400
      3330 AV(J)=((NTSOL(J)-1)*AV(J)-TIME)/NTSOL(J)
C          SAVE ALLOCATION AND SOLUTION INFORMATION FOR SUBPROBLEM J
      3400 CALL SAVER(J)
           K = NTEMP(2)
C          OUTPUT SOLUTION INFORMATION FOR SUBPROBLEM J
           WRITE (6,2) J,K,(INCOMB(KK), KK=1,NCOL)
         2 FORMAT (' SUB',I2,' SOL',I3,':  ',20I5)
C          TEST IF ALL SUBPROBLEMS HAVE BEEN SOLVED; IF SO, NO BRANCHING
C          NECESSARY
           IF (NJFIX - NSUBS + 1) 2200,2300,9999
C
C          COMPUTATIONS COMPLETED; OUTPUT OPTIMAL SOLUTION INFORMATION
      5000 CALL CLTSOL(INC)
           CALL RCLOCK(JTIME,ITCT)
           TTCT = JTIME/100.
C          OUTPUT ALGORITHMIC PERFORMANCE INFORMATION
           WRITE (6,7) TTCT
         7 FORMAT (/' TOTAL SOLUTION TIME =',F7.2,'SECONDS')
           WRITE (6,92) TPH1,TALG,TALG,(NTSOL(J), J=1,NSUBS)
        92 FORMAT (' TIME IN PHASE1 =',F6.2,' SECONDS'/' MASTER PROBLEM SOLVE
          1D',I10,' TIMES:  TOTAL TIME SOLVING MASTER PROBLEM =',F6.2,' SECON
          2DS'/' NUMBER OF TIMES EACH SUBPROBLEM SOLVED =',18X,10I5)
           WRITE (6,9) (NSOL(J), J=1,NSUBS)
         9 FORMAT (' NUMBER OF DISTINCT SOLUTIONS OBTAINED IN EACH SUBPROBLEM
          1 =',10I5)
           WRITE (6,91) TSUBS
        91 FORMAT (' TOTAL TIME SOLVING SUBPROBLEMS =',F6.2,' SECONDS')
           GO TO 10
      9999 STOP
           END
           BLOCK DATA
C
C          INITIALIZES GLOBAL PROGRAM CONSTANTS
C          SUBROUTINE ADAPTED FROM LINEAR PROGRAMMING CODE LPM-1, WRITTEN
C          BY J.A. TOMLIN (OPERATIONS RESEARCH, STANFORD UNIVERSITY)
C
           IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
          1          INTEGER*4 (I-N,Q)
           COMMON/BLOCK/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,CRO,QMA,QBA,QFI,
          1              QEO,QBL,QA,QPL,QMI,QZ,QI,QF,QN,QO,NRMAX,NEMAX,QB,QC,
          2              QE,QH,QL,QO,QR,QM,QG,NTMAX
           REAL ZTOLZE/1.E-5/,ZTOLPV/1.E-4/,ZTOCST/1.E-3/,ZTOLRJ/1.E-4/,
          1     ZTOLSM/1.E-10/
           INTEGER*4 NRMAX/60/,NTMAX/250/,NEMAX/1000/
           INTEGER*4 CRO/'ROW '/,QMA/'MATR'/,QBA/'BASI'/,QFI/'FIRS'/,
          1QEO/'EOF '/,QBL/'    '/,QA/'A   '/,QPL/' +  '/,QMI/' -  '/,
          2QZ/' Z  '/,QI/'I   '/,QF/'F   '/,QN/'N   '/,QO/'O   '/,
          3QB/'B   '/,QC/'C   '/,QE/'E   '/,QH/'H   '/,QL/'L   '/,
          4QO/'O   '/,QR/'R   '/,QM/'M   '/,QG/'G   '/
           END
           SUBROUTINE INPUT(IFPROB)
C
C          INPUTS PROBLEM DATA
```

```
C        CHECK IF ANY SUBPROBLEM HAS PREVIOUSLY BEEN SOLVED UNDER ITS
C        CURRENT ALLOCATION
      CALL CCMCHK(J,K)
      IF (IICM .EQ. 0) GO TO 3000
C        BRANCH ON ALLOCATION VARIABLE IN SUBPROBLEM J
 2200 CALL CMBRAN(J,K)
      IF (MSTAT .EQ. 0) GO TO 2000
C        FIX ALLOCATION TO SUBPROBLEM J AT LEVEL K
 2300 JFIX(J) = K
      NJFIX = NJFIX + 1
      JFIXSM = JFIXSM + IZSTAR(K,J)
      IF (NJFIX .LT. NSUBS) GO TO 2100
C              ALL SUBS FIXED: COMPARE SOLUTION WITH INCUMBENT
      IF (JFIXSM .LE. IZINC) GO TO 2500
      DO 2400 I=1,NSUBS
 2400 INC(I) = JFIX(I)
      IZINC = JFIXSM
      CALL PCLOCK(JTIME,ITOT)
      TOPT = JTIME/100.
      WRITE (6,93) TOPT,IZINC
   93 FORMAT (' TIME =',F7.2,' SECONDS:  IZINC =',I10)
 2500 JFIX(J) = 0
      JFIXSM = JFIXSM - IZSTAR(K,J)
      NJFIX = NJFIX - 1
      GO TO 2000
C
C        SELECT NEXT SUBPROBLEM TO SOLVE
 3000 AVMAX = -10000.
      J = 0
      DO 3100 I=1,NSUBS
      IF (JFIX(I) .GT. 0) GO TO 3100
      IF (AVMAX .GE. AV(I)) GO TO 3100
      J = I
      AVMAX = AV(I)
 3100 CONTINUE
 3200 NTSOL(J) = NTSOL(J) + 1
      CALL PCLOCK(JTIME,ITOT)
C        LOAD PROBLEM DATA FOR SUBPROBLEM J
      CALL LDDATA(J)
      IPAR = 0
      ITSIM = 99999
      IFEAS = 0
C        GET OPTIMAL LP-SOLUTION TO SUBPROBLEM J
      CALL NORMAL
C        TEST IF SUBPROBLEM J IS LP-FEASIBLE
      IF (MSTAT .EQ. 6A) GO TO 3300
C        STORE OPTIMAL DUAL MULTIPLIERS IF DISTINCT FROM PREVIOUS MULTS.
      CALL FISAVR(J)
C        SUBPROBLEM J LP-FEASIBLE; GET INITIAL LOWER BOUND ESTIMATE ON
C        MAX. IP-OBJ. VAL.
      CALL GETLBC(J)
      INITBD = INCOME(1)
C        USE BRANCH-AND-BOUND ROUTINE TO COMPUTE OPT. IP-SOLUTION TO
C        SUBPROBLEM J
      CALL BANDB(INITBD)
 3300 CALL PCLOCK(JTIME1,ITOT)
      TIME = (JTIME1 - JTIME)/100.
      TSUBS = TSUBS + TIME
C        UPDATE SUBPROBLEM SELECTION RULE INFORMATION FOR SUBPROBLEM J,
C        IN ACCORDANCE WITH VALUE SPECIFIED FOR IPARM
      GO TO (3310,3320,3330,3400),IPARM
```

```
      C
      C              PHASE 1
      C
            DO 100 J=1,NSUBS
            NLAMDA(J) = 0
            NTSOL(J) = 0
            AV(J) = 0.
            NSOL(J) = 1
            JFIX(J) = 0
            CALL INPUT(J)
            DO 50 I=1,NLINK
         50 IBSTAR(I,1,J) = IDINT(B(I+1))
            ITSINV = 99999
            CALL NORMAL
        100 CALL PISAVR(J)
      C
        200 CALL ALOCTE(IPAR)
            NNEWPI = 0
            DO 300 J=1,NSUBS
            LL = NSOL(J)
            LISTL = LL + 1
            DO 250 K=1,LL
            DO 240 I=1,NLINK
            IF (IBSTAR(I,K,J) .NE. IBSTAR(I,LISTL,J)) GO TO 250
        240 CONTINUE
            GO TO 300
        250 CONTINUE
            CALL LOCATA(J)
            ITSINV = 99999
            CALL NORMAL
            NEWPI = NLAMDA(J)
            CALL PISAVR(J)
            NEWPI = NLAMDA(J) - NEWPI
            NNEWPI = NNEWPI + NEWPI
            NSOL(J) = LISTL
        300 CONTINUE
            IF (NNEWPI) 400,400,200
        400 DO 500 J=1,NSUBS
        500 NSOL(J) = 0
            CALL PCLOCK(JTIME,ITOT)
            TPH1 = JTIME/100.
            GO TO 2100
      C
      C              PHASE 2
      C
      C         BACKTRACK TO LAST PROMISING NODE IN ALLOCATION-VARIABLES SEARCH
      C         TREE
      2000 CALL BBKTRK
      C         IF LIST OF STORED NODES IS EMPTY, COMPUTATIONS ARE COMPLETED
            IF (IFLL .EQ. 0) GO TO 9000
      2100 NALC = NALC + 1
            CALL PCLOCK(JTIME,ITOT)
      C         SOLVE MASTER PROBLEM TO OBTAIN PROMISING NEW ALLOCATION OF
      C         LINKING RESOURCE TO SUBPROBLEMS
            CALL ALOCTE(IPAR)
            CALL PCLOCK(JTIME1,ITOT)
            TALC = TALC + (JTIME1 - JTIME)/100.
            IPAR = 1
      C         TEST FOR FATHOMING OF CURRENT NODE
            IF (INCVAL .LE. IZINC) GO TO 2000
            IZBAR = INCVAL
```

```
      C
      C         DECOMPOSITION ROUTINE, DMLC
      C         DECOMPOSITION AND SOLUTION OF PURE-INTEGER LINEAR PROGRAMS
      C         (WITH GENERAL INTEGER VARIABLES) WITH BLOCK ANGULAR CONSTRAINT
      C         MATRICES.
      C********MULTIPLE LINKING CONSTRAINTS CASE*******
      C         GARY A. KOCHMAN (OPERATIONS RESEARCH), SEPT. 1976, MOD 1
      C
            IMPLICIT REAL*4 (A,C,E-H,O,P,R-W,Z), REAL*8 (B,D,X,Y),
           1          INTEGER*4 (I-N,Q)
            INTEGER*2 JH,KINBAS,LA,LE,IA,IE
            INTEGER*2 IPART,INCOMB,IVBND,IVIB,ICBND
            INTEGER*2 IZSTAR,IBID,IBBND,IZBND
            INTEGER*2 IAS,LAS
            DOUBLE PRECISION E(1000)
            DIMENSION A(500),INC(10),AV(10),NTSOL(10)
            COMMON/BLOCK1/ ZTOLZE,ZTOLPV,ZTOCST,ZTOLRJ,ZTOLSM,QRO,QMA,QBA,QFI,
           1               QEO,QBL,QA,QPL,QM1,QZ,QI,QF,QN,QU,NRMAX,NCMAX,QB,QC,
           2               QE,QF,QL,QC,QR,QM,QG,NTMAX
            COMMON/BLOCK2/ DPART(122),REVBND,INCVAL,ICOL,LISTL,IVAL,IDIR,
           1               NPIVCT,IPTYPE,RCOST,IFEAS,IPART(122),INCOMB(122),
           2               IVBND(500),IVIB(500),ICBND(500)
            COMMON/BLOCK3/ RLAMDA(5,20,10),ZNAUX(20,10),ZJBAR(10),NLINK,NSUBS,
           1IZINC,IZBAR,IDEM,IBLL,NJFIX,JFIXSM,IB(5),JFIX(10),NSOL(10),
           2NLAMDA(10),IBSTAR(5,40,10),IBUSED(5,40,10),IBUB(5,10),IBLB(5,10),
           3IZSTAR(40,10),IBID(2,200),IBBND(200),IZBND(200)
            COMMON/BLOCK4/ XLBS(20,10),XUBS(20,10),AMATRS(1000),RHS(10,10),
           1NROWS(10),NCOLS(10),JFIRST(11),IAS(1000),LAS(20,10)
            COMMON B,XLB(122),XUB(122),B(60),X(60),Y(60),YTEMP(60),DSUM,DPROD,
           1       DY,DE,DP,A,ICNAM(122,2),NAME(20),NTEMP(20),SUMINF,MSTAT,
           2       IOBJ,IROWP,IVIN,IVOUT,ITCNT,INVFRQ,ITRFRQ,ITSINV,IFEAS,
           3       JCOLP,NROW,NCOL,NELEM,NETA,NLELEM,NLETA,NUELEM,NUETA,
           4       IE(1000),IA(500),LE(252),LA(122),KINBAS(122),JH(60)
      C
      C         MAIN PROGRAM
      C         COORDINATES APPLICATION OF THE VARIOUS STEPS OF THE DECOMPOSI-
      C         TION ALGORITHM.
      C
      C         INITIALIZE PARAMETERS
        10 CALL RCLOCK(ITCT)
            READ(5,1,END=9999) NSUBS,NLINK,IPARM,IZINC
         1 FORMAT (5I10)
            READ (5,1) (IB(I), I=1,NLINK)
            DO 20 J=1,122
            ICNAM(J,1) = 0
        20 ICNAM(J,2) = 0
            ICBJ = 1
            ITRFRQ = 99999
            INITBC = 0
            IBLL = 0
            IZBAR = 10000
            JFIX(NSUBS) = 0
            IPAR = 0
            NJFIX = 0
            JFIXSM = 0
            JFIRST(1) = 1
            NALC = 0
            TALC = 0.
            TSUBS = 0.
            NP1 = NLINK + 1
```

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: SOL-76-20
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): Computer Programs for Decomposition in Integer Programming.

5. TYPE OF REPORT & PERIOD COVERED: Technical Report.

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): Gary A. Kochman

8. CONTRACT OR GRANT NUMBER(s): N00014-76-C-0418, DAAC 29-74-C-0079, F44620-74-C-0079

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Department of Operations Research
Stanford University
Stanford, CA 95305

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Operations Research Program, Code 434
Office of Naval Research
Arlington, VA 22217

Mathematics Division
U.S. Army Research Office
Box CM, Duke Station
Durham, N.C. 27706

Directorate of Mathematical and Information Sciences
Department of the Air Force
Air Force Office of Scientific Research
Bolling Air Force Base, D.C. 20332

12. REPORT DATE: September 1976

13. NUMBER OF PAGES: 162 (165 p.)

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report): UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release and sale; distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Mathematical Programming
Integer Programming
Decomposition

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report gives documentations for two computer codes, DSLC and DMLC for solving block angular integer programming problems. The first code, DSLC, is for the single linking constraint case and the second DMLC is for the multiple linking constraint case.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)