

AD-A062 012

NAVAL SURFACE WEAPONS CENTER DAHLGREN LAB VA
SOLUTION OF THE INTEGER CONCAVE PROGRAM USING THE IC PHI N ALGO--ETC(U)
NOV 78 H W LOOMIS
NSWC/DL-TR-3120-VOL-2

F/G 12/2

UNCLASSIFIED

NL

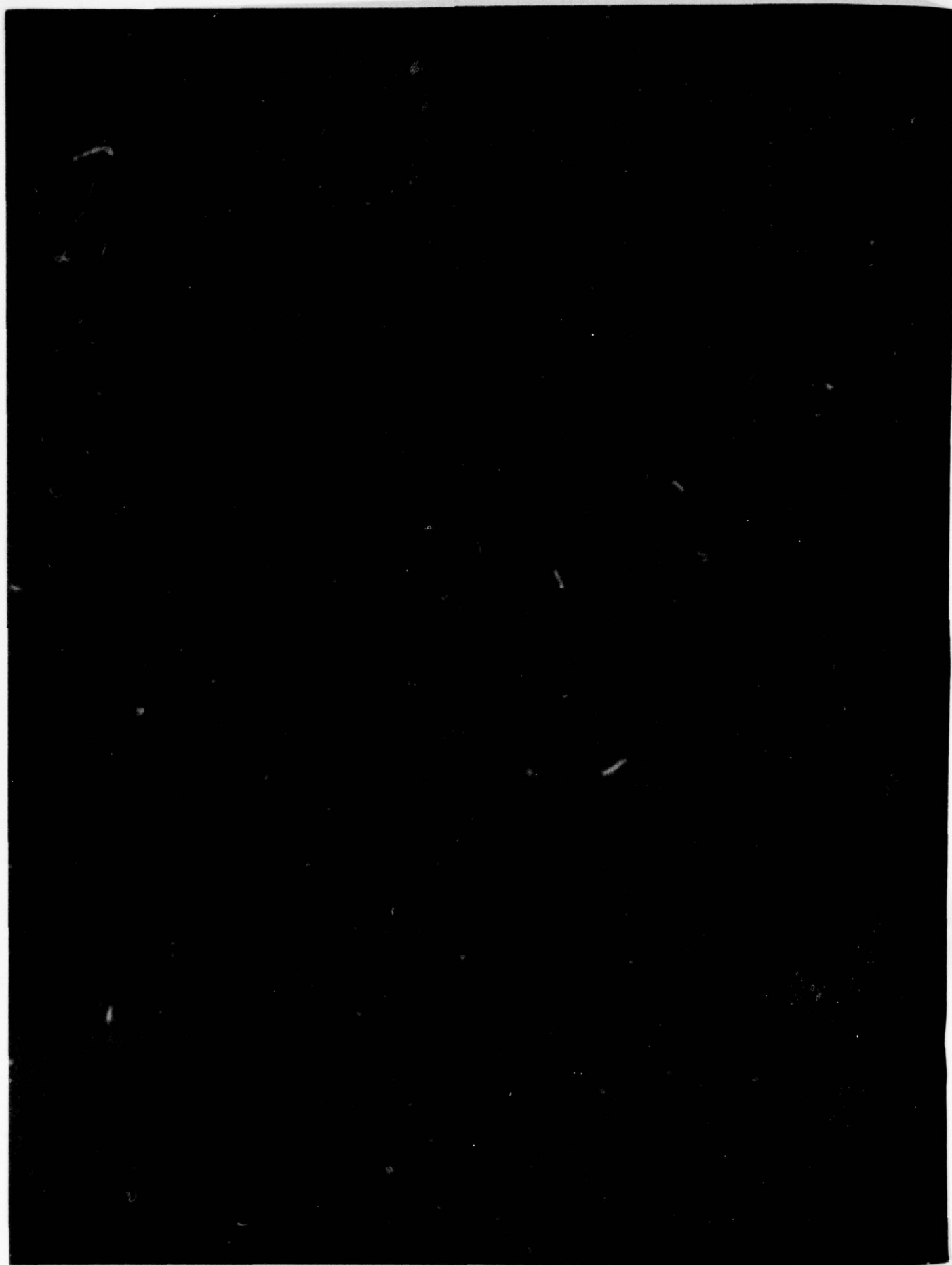
1 OF 2
AD
A062 012



DDC FILE COPY AD A062012

(12) LEVEL II

DDC
RECEIVED
DEC 18 1978
B



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NSWC/DL-TR-3128- Vol -2	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER 9
4. TITLE (and Subtitle) Solution of the Integer Concave Program Using the ICON Algorithm, Volume 2.	5. TYPE OF REPORT & PERIOD COVERED FINAL rept.	
7. AUTHOR(s) Harlan W. Loomis	6. CONTRACT OR GRANT NUMBER(s)	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Surface Weapons Center (K30) Dahlgren, Virginia 22448	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61152N, ZR00001 ZR01407, NWL/O1K07	
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Surface Weapons Center (K30) Dahlgren, Virginia 22448	12. REPORT DATE 1 November 1978	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 12) 135p.	13. NUMBER OF PAGES 149	
15. SECURITY CLASS. (of this report) UNCLASSIFIED		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) DDC DEC 18 1978 B		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) algorithm integer programming optimization theory branch-and-bound linear programming systems analysis computer program mathematical programming concave function nonlinear programming convex function operations research		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The branch-and-bound technique has been the basis for algorithms to solve both the mixed integer linear program (having linear objective function, linear constraints, and integrality restrictions on some variables) and the concave nonlinear program (having a separable concave objective function and linear constraints). The subprograms for each of these branch-and-bound algorithms are linear programs with simple upper bounds. → next page (Continued)		

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 68 IS OBSOLETE
S/N 0102-LF-014-660

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

391 598

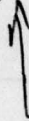
Handwritten initials/signature

~~UNCLASSIFIED~~

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

In Volume 1, a new branch-and-bound algorithm is presented for the composite mixed integer, concave nonlinear program. This integer concave (ICON) algorithm has been implemented in the form of a computer program coded in FORTRAN. A guide to the use of the computer program together with examples of its application are included in Volume 1. Documentation of the computer program is included in Volume 2.

0
zero



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT

The branch-and-bound technique has been the basis for algorithms to solve both the mixed integer linear program (having linear objective function, linear constraints, and integrality restrictions on some variables) and the concave nonlinear program (having a separable concave objective function and linear constraints). The subprograms for each of these branch-and-bound algorithms are linear programs with simple upper bounds.

In Volume 1, a new branch-and-bound algorithm is presented for the composite mixed integer, concave nonlinear program. This integer concave (ICON) algorithm has been implemented in the form of a computer program coded in FORTRAN. A guide to the use of the computer program together with examples of its application are included in Volume 1. Documentation of the computer program is included in Volume 2.

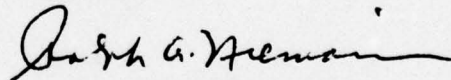
ACCESSION for		
NTIS	White Section	<input checked="" type="checkbox"/>
DDC	Buff Section	<input type="checkbox"/>
UNANNOUNCED		<input type="checkbox"/>
JUSTIFICATION _____		
BY _____		
DISTRIBUTION/AVAILABILITY CODES		
Dist.	AVAIL. and/or	SPECIAL
A		

FOREWORD

Optimization problems involving integer-valued decision variables occur frequently in operations research and in systems analysis. Many cost-effectiveness analyses performed within the Department of Defense are optimization problems of this type. Specific applications related to amphibious operations occur in mine warfare, logistics, and fire support. This report presents a new method for solving a large class of integer nonlinear optimization problems.

The material presented here implements ideas developed during the author's program of studies in the Department of Operations Research, School of Engineering and Applied Science, The George Washington University, Washington, D. C. Support for this research in integer nonlinear optimization and the development of computational algorithms has been provided by the Naval Surface Weapons Center's Independent Research Program. The author is presently involved in surface warfare applications of this optimization technique.

RELEASED BY:



RALPH A. NIEMANN, Head
Warfare Analysis Department

CONTENTS

	Page
Abstract	iii
Foreword	v
List of Figures	ix
List of Exhibits	x
1. Introduction	1
Appendices:	
A. Flow Schematics	3
B. Parameters and Arrays	15
Fixed Storage	18
Variable Storage	25
C. Branch-and-Bound List	33
D. Subprogram Representation and Solution	43
E. Subroutines	53
F. Listing of the ICON Algorithm	65
G. References	131
H. Distribution	135

LIST OF FIGURES

Figure	Page
1. Flow Schematic for the ICON Branch-and-Bound Algorithm	6
2. Flow Schematic for the ICON Branch-and-Bound Algorithm Using the Variable Best Upper Bound Method	10
3. Subprogram Representation	46
4. Calls to ICON Subroutines	56
5. Calls to System Subroutines	63

LIST OF EXHIBITS

Exhibit	Page
1. Labeled Common Storage	19
2. Example of Variable Dimensioning	26
3. Manipulation of the Branch-and-Bound List	40
4. Listing of the ICON Algorithm	68

SOLUTION OF THE
INTEGER CONCAVE PROGRAM
USING THE ICON ALGORITHM

VOLUME 2

1. INTRODUCTION

The ICON branch-and-bound algorithm has been implemented in the form of a computer program coded in FORTRAN for the CDC 6700 computer system. Appendices A and B of Volume 1 describe the input data and the user subroutines which are required by the program. Appendix C of Volume 1 provides descriptive examples of input data and user subroutines for three test problems. Volume 2 includes that additional information which is needed to fully document the computer program.

Although the computer program for the ICON algorithm was developed for use on the CDC 6700 computer, a special effort was made to assure that the programming techniques utilized would be compatible with a wide variety of computers. Minimal effort should be required to convert the program for use on other computers having a FORTRAN compiler.

The ICON algorithm, like many branch-and-bound algorithms, requires substantial amounts of computer storage for most efficient operation. In the CDC 6700 computer system, the availability of a random access mass storage device fulfills this requirement. The CDC 6700 system subroutines which are required in order to utilize this capability are described in this report. Other computers possess similar capabilities to which the computer program can be readily adapted.

Appendix A provides the general flow schematics on which the computer program is based. Appendix B gives the definitions of parameters and arrays utilized in the program. Aspects of programming

technique are discussed in Appendices C and D. Appendix C deals with the use of mass storage for the branch-and-bound list while Appendix D deals with the representation and solution of subprograms. The subroutines which comprise the ICON algorithm as well as the system subroutines which are used by the algorithm are described in Appendix E. Finally, Appendix F provides a listing of the computer code.

APPENDIX A

FLOW SCHEMATICS

The general flow schematics for the ICON branch-and-bound algorithm are presented in this appendix. Figure 1 shows the basic sequence of steps which are performed when Method 1 is used, and Figure 2 shows the sequence when Method 2 is used. Methods 1 and 2 are discussed in Volume 1.

The general flow schematics shown in Figures 1 and 2 agree exactly for phase 2 of the branch-and-bound algorithm. The schematic of Figure 1 can be regarded as being embedded in the schematic of Figure 2 if the test "Phase = 1?" in boxes 6, 18 and 28 of Figure 2 is replaced by the test "Phase = 1 and the variable best upper bound method is in use ?" With this replacement, Figure 2 represents a general flow schematic for the computer program implementing the ICON algorithm.

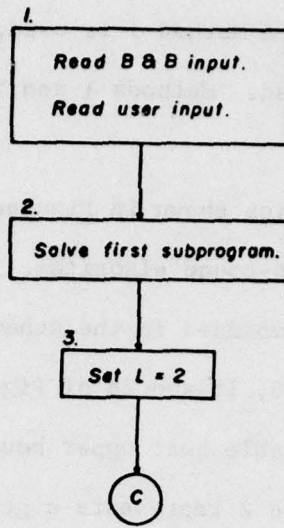


FIGURE 1

FLOW SCHEMATIC FOR THE ICON BRANCH-AND-BOUND ALGORITHM

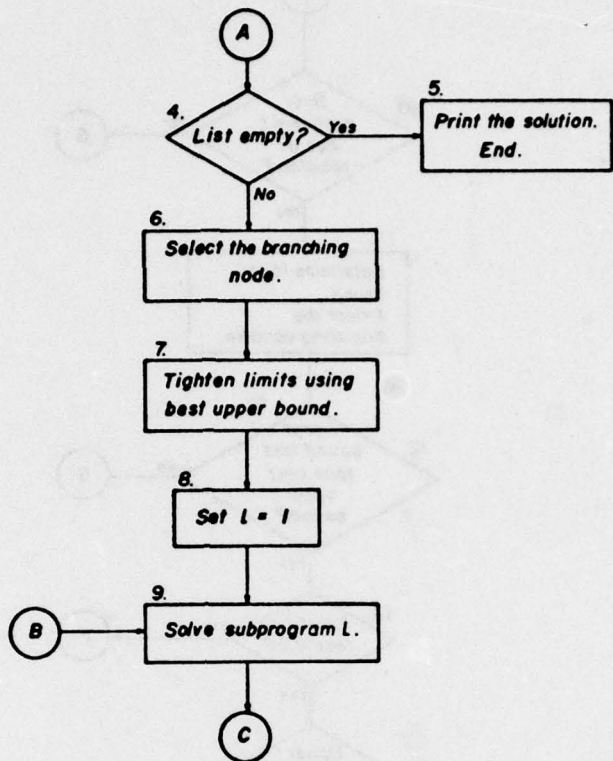


FIGURE 1 (Continued)

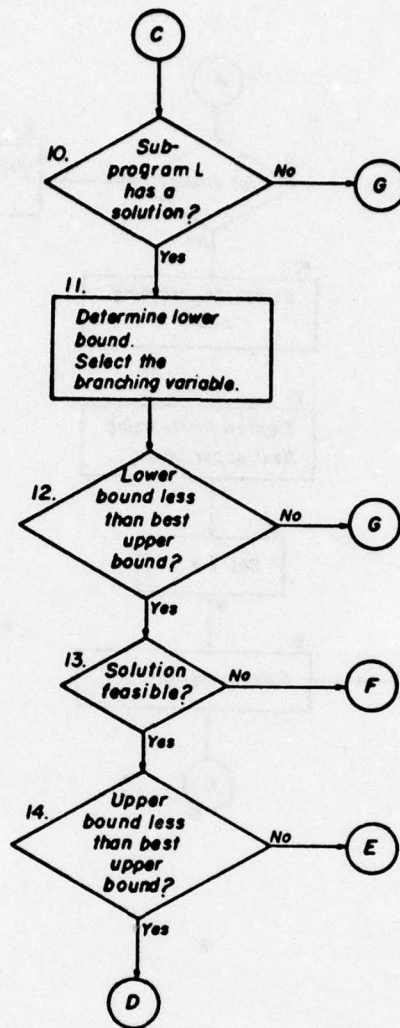


FIGURE 1 (Continued)

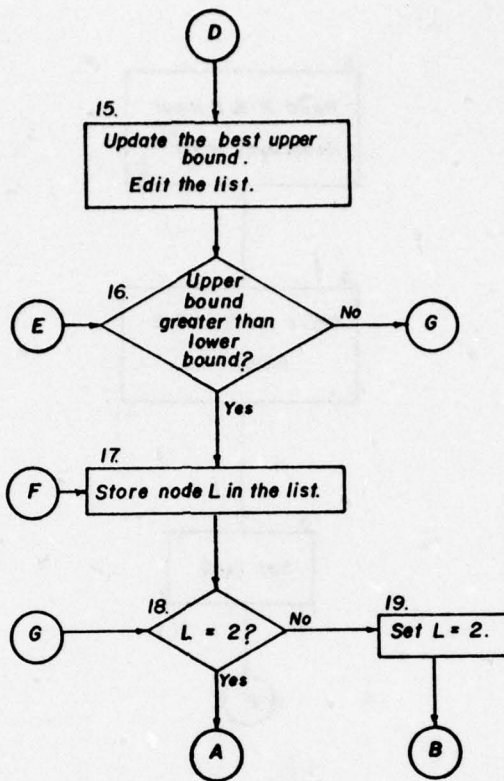


FIGURE 1 (Continued)

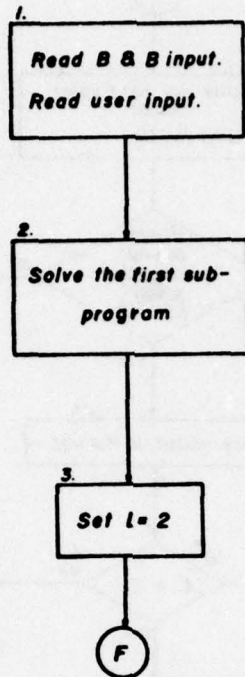


FIGURE 2

FLOW SCHEMATIC FOR THE ICON BRANCH-AND-BOUND ALGORITHM
USING THE VARIABLE BEST UPPER BOUND METHOD

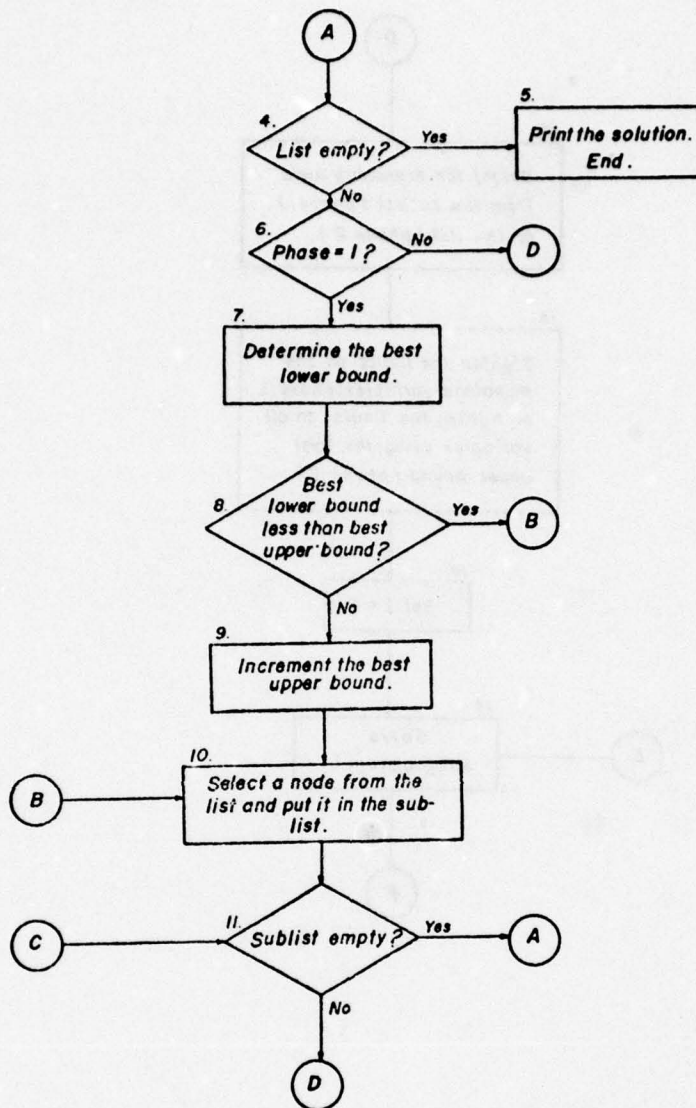


FIGURE 2 (Continued)

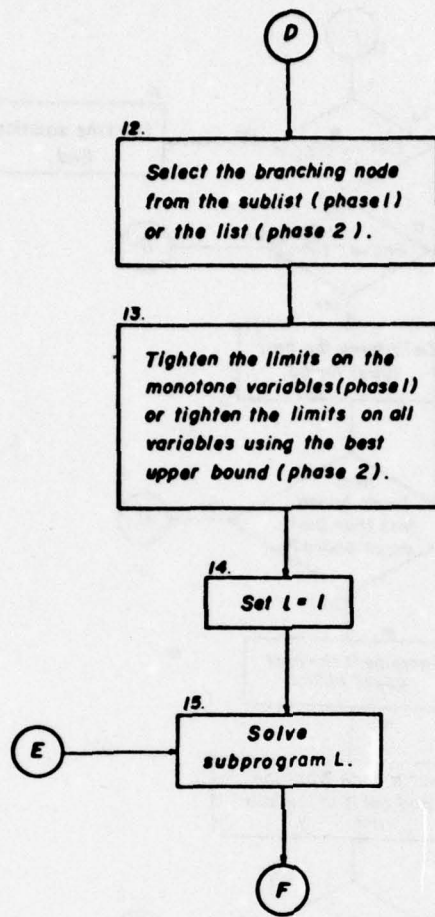


FIGURE 2 (Continued)

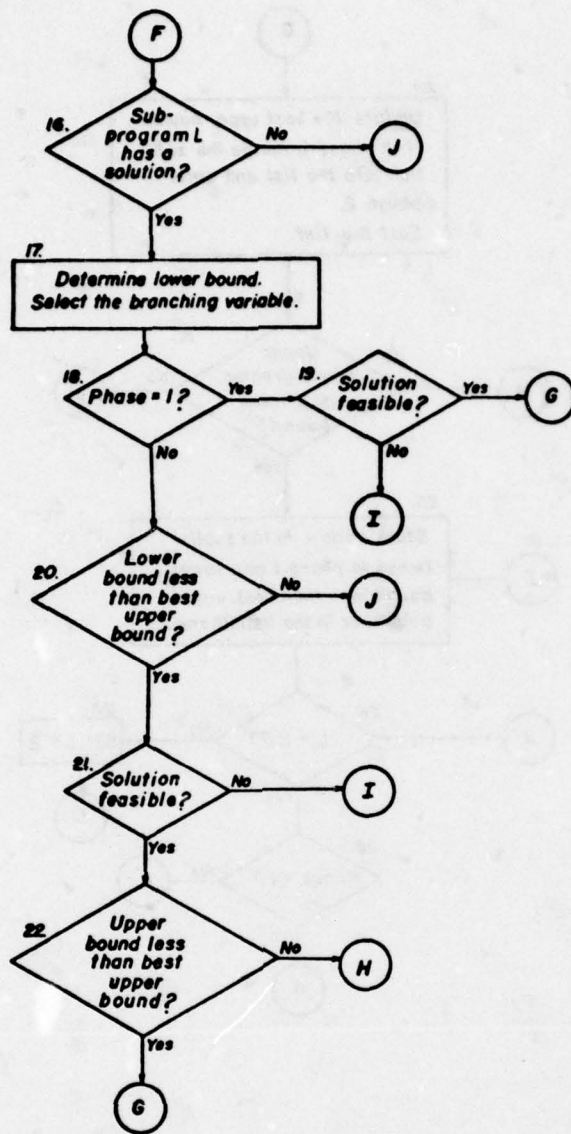


FIGURE 2 (Continued)

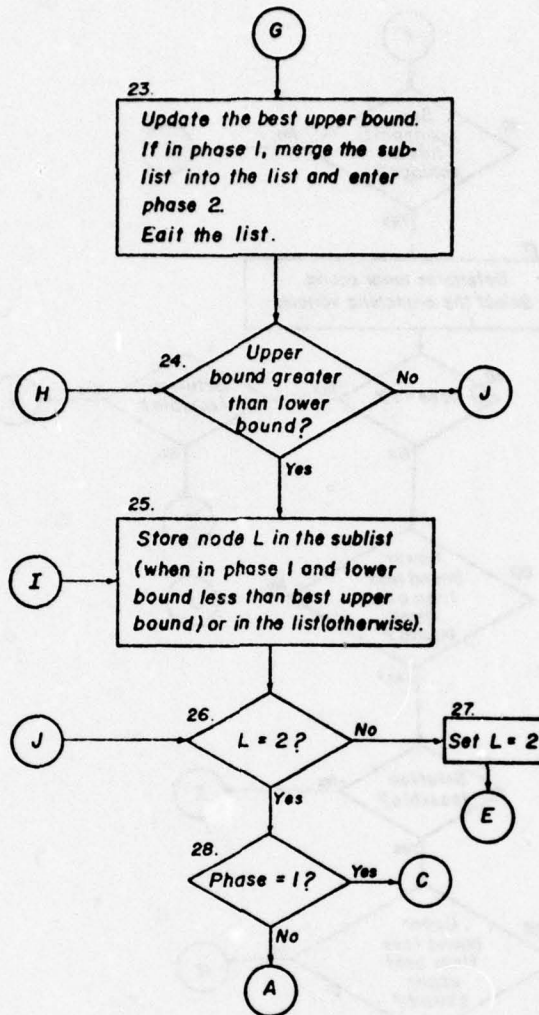


FIGURE 2 (Continued)

APPENDIX B
PARAMETERS AND ARRAYS

The parameters and arrays used in the ICØN branch-and-bound algorithm are described in this appendix. Fixed storage consists of all parameters and two arrays (arrays IMASS2 and IMASS3) which are located in labeled common storage. All other arrays are given variable dimensions, with the actual value of these dimensions being assigned during program execution.

The use of variable dimensions results in the minimization of the core storage required during a computer run, an important factor in the CDC 6700 computer system where the cost of a computer run is proportional to the amount of core storage used. A further benefit from variable dimensioning is the elimination of artificial limits on individual problem parameters (such as the number of variables, N, or the number of constraints, M) which typically result when fixed dimensions are assigned in a computer code. The use of variable dimensioning thus results both in reduced running cost and in program flexibility, where either small or large problems can be solved with the only changes being in the user supplied subroutines.

FIXED STORAGE

Exhibit 1 shows the labeled commons used in the ICØN branch-and-bound algorithm. CØMMØN/PØ/ consists of a set of pointers used in conjunction with the variable dimensioning of arrays (parameters NI1 through NI12 and parameters NF1 through NF24) and in conjunction with the branch-and-bound list (parameters NIMS2 and NFMS3). CØMMØN/P1/ consists of the control parameters discussed in Appendix A, Volume 1.

CØMMØN/P2/ consists of those parameters which are used repeatedly in the algorithm. These are:

Variable	Description
EPSI and EPSIM	Tolerances (10^{-11} and -10^{11}) used to test if a number is zero within roundoff error
BIGN	A large number (10^{100}) used in place of $+\infty$
BEGTM	The clock time at which the branch-and-bound algorithm commenced the processing of a given program
M1, M2 and M3	The number of less than or equal, equality, and greater than or equal constraints in the program ¹
M4	= $M2 + M3$, the number of artificial variables associated with the program
N1	= $N + (M3 + M1) + (M2 + M3)$, the total number of variables associated with the program including slack, surplus and artificial variables
MP1	= $M + 1$, the number of basic variables in a one phase linear program, and the position of the linear programming phase 2 objective function among the basic variables

¹See the discussion of the constraint matrix and right-hand-side vector in Appendix A, Volume 1.

EXHIBIT 1

LABELED COMMON STORAGE

```
COMMON/P0/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12,  
1      NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11,  
2      NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22,  
3      NF23,NF24,NFMS3  
COMMON/P1/N,M,ITYPE,NSTPAT,NODPL1,NBVR1,NTITE1,NODRL2,NBVR2,  
1      NTITE2,MXLIST,LISTOP,ITAPE,IFR,MXITER,NBINV,IOUTPT,  
2      ITRACE,MSTART,TIME1,TOL1,TOL2,PCRUB,ALPHA(10)  
COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,  
1      NM1M3,N1P2,NP1,NSUM,NTC,M10  
COMMON/P3/NODNOT,UNOT,IRUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,  
1      NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,RLB,NBRNOD,PBRNOD,  
2      NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,  
3      YSIG,IFEAS,IBPVR1,IUPDN1,XBPVR1,IBVR2,IUPDN2,XBRVR2,  
4      LI0,NITER,NBINV,M7,IPHASE,NPHASE,NM3N7,IALGO,IEOJ  
COMMON/P4/SAVE,KBRAN,X1  
COMMON/P5/IROUNO  
COMMON/A0/IMASS2(1001),IMASS3(1001)
```

<u>Variable</u>	<u>Description</u>
MP2	= $M + 2$, the number of basic variables in a two phase linear program, and the position of the linear programming phase 1 objective function among the basic variables
NM3	= $N + M3$, a parameter used in subroutines GETCØL and ØBJ1
NM1M2	= $N - M1 - M2$, a parameter used in subroutine GETCØL
NM1M3	= $N + M1 + M3$, the total number of variables excluding the artificial variables
N1P2	= $N1 + 2$, the total number of variables plus the linear programming phase 1 and phase 2 objective functions
NP1	= $N + 1$, a parameter used in subroutines BØX2, BØX15 and ØBJ1
NSUM	The total number of nonzero entries in the constraint matrix ¹
NTC	The number of constants in the table of constants for the constraint matrix ¹
M10	The position of the objective function (either MP1 or MP2) in the current phase of the linear program

COMMON/P3/ consists of those parameters which are required throughout the branch-and-bound algorithm. These are:

<u>Variable</u>	<u>Description</u>
NØDNØT	The node number corresponding to the current best upper bound
UNØT	The current best upper bound

¹See the discussion of the constraint matrix and right-hand-side vector in Appendix A, Volume 1.

<u>Variable</u>	<u>Description</u>
IBUBØP	Indicator for the variable best upper bound method (0 = do not use the method; 1 = use the method)
LPHASE	Current phase of the branch-and-bound algorithm (1 = no feasible point has yet been determined; 2 = a feasible point has been determined and the current best upper bound represents a feasible point)
NØDRUL	The current node selection rule according to the phase of the branch-and-bound algorithm
NBVRUL	The current branching variable selection rule according to the phase of the branch-and-bound algorithm
NIGHT	The current limit tightening rule according to the phase of the branch-and-bound algorithm
NLIST	Number of nodes currently in the branch-and-bound list
NLISTS	Number of nodes currently in the branch-and-bound sublist
NFEAS	The current total number of nodes for which the corresponding subprogram required complete solution
LSTMX	The current maximum size attained by the branch-and-bound list
ITRTØT	The current total number of linear programming iterations performed
ITRMAX	The current maximum number of linear programming iterations performed along any single branch of the branch-and-bound tree
BLB	The best lower bound
NBRNØD	The node number of the branching node
PBRNØD	The processing order number associated with the branching node

<u>Variable</u>	<u>Description</u>
NBRVAR	The branching variable associated with the branching node
NUPDWN	The direction for continued branching (when the node selection rule is the LIFO rule) for the branching node
XBRNØD	The value of the branching variable in the solution corresponding to the branching node
TBRNØD	The constant associated with the subprogram for the branching node
NØDE	The current node number
LNØDE	Indicator for the current node (1 = the lower node emanating from the branching node; 2 = the upper node emanating from the branching node)
Z	The optimal objective function value in a subprogram
BØUNDL	The lower bound for the current node
BØUNDU	The upper bound for the current node
TSIG	The constant associated with the subprogram for the current node
IFEAS	Indicator as to the feasibility with respect to the master problem of the solution to the current subprogram (0 = not feasible; 1 = feasible)
IBRVR1	Branching variable selection under the first branching variable selection strategy
IUPDN1	The direction for continued branching corresponding to branching variable IBRVR1
XBRVR1	The value of the branching variable IBRVR1
IBRVR2	Branching variable selection under the second branching variable selection strategy
IUPDN2	The direction for continued branching corresponding to branching variable IBRVR2

<u>Variable</u>	<u>Description</u>
XBRVR2	The value of the branching variable IBRVR2
L10	The number of nonbasic variables associated with the current subprogram tableau
NITER	A counter for the number of linear programming iterations which have been performed to reach the current subprogram tableau
NBINV	A counter, similar to NITER, for the number of linear programming interactions which have been performed since the last basis reinversion
M7	The number of basic variables associated with the current subprogram tableau (either $M + 1$ or $M + 2$)
IPHASE	Indicator of the method being applied to solve the current subprogram (1 = one phase method; 2 = two phase method)
NPHASE	The current phase of the method being applied to solve the current subprogram (when $IPHASE = 1$, $NPHASE$ is not needed and is set to 0; when $IPHASE = 2$, $NPHASE$ is set to 1 or 2)
NM3M7	$= N + M3 + M7$, the number of basic and nonbasic variables associated with the current subprogram tableau
IALGØ	The linear programming algorithm to be applied to move to an optimal tableau for the current subprogram (1 = primal algorithm; 2 = dual simplex algorithm)
IEØJ	Indicator for the tableau resulting from the application of a linear programming algorithm (0 = optimal; 1 = primal infeasible; 2 = primal unbounded; 3 = dual value exceeds the current best upper bound; 4 = maximum number of linear programming iterations exceeded)

COMMON/P4/ consists of three parameters which are used as temporary storage in subroutine BOX15. COMMON/P5/ consists of a single parameter

which is set in subroutine INPUT3 and used in subroutine BOX17. It is used to indicate whether or not the objective function is an integer valued function for a mixed integer linear program. COMMON/AO/ consists of two arrays used in conjunction with the branch-and-bound list.

VARIABLE STORAGE

Arrays IF and F are assigned fixed dimensions in program MAIN compatible with the program or programs to be solved. These two arrays are in turn subdivided within the branch-and-bound algorithm into the various integer arrays and floating point arrays required to solve a program. This subdivision varies from program to program depending upon the program structure as specified in the input.

Exhibit 2 shows how the variable dimensioning of arrays is accomplished. Program MAIN transfers control to subroutine ICØN, at the same time passing the locations of arrays IF and F together with the corresponding dimensions NI and NF as shown at line 1 of subroutine ICØN. These two arrays are dimensioned at line 18 of subroutine ICØN. At lines 24-25, control is transferred to subroutine BØX1 which reads the input for the branch-and-bound algorithm and allocates the storage occupied by the arrays IF and F to the various integer and floating point arrays needed in the branch-and-bound algorithm. This allocation is accomplished by developing dimensions ND1 through ND11 for these arrays and by developing pointers NI1 through NI12 for the integer arrays and pointers NF1 through NF24 for the floating point arrays. The subsequent use of these variable dimensioned arrays is exemplified by the call to subroutine BØX7 at line 43 of subroutine ICØN. The first location of the eighth integer array is IF(NI8) and the first location of the thirteenth floating point array is F(NF13). It happens that both of these arrays have the same dimension, ND10. The nomenclature

EXHIBIT 2

EXAMPLE OF VARIABLE DIMENSIONING

SUBROUTINE ICON (IF,F,NI,NF)	ICON0001
C BRANCH-AND-BOUND ALGORITHM FOR THE INTEGER CONCAVE PROGRAM.	ICON0002
COMMON/P0/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12,	ICON0003
1 NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11.	ICON0004
2 NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22,	ICON0005
3 NF23,NF24,NFMS3	ICON0006
.	.
.	.
DIMENSION IF(NI),F(NF)	ICON0018
.	.
.	.
100 CALL BOX1 (IF,F,NI,NF,ND1,ND2,ND3,ND4,ND5,ND6,ND7,ND8,ND9,ND10,	ICON0024
1 ND11,NDMS2,NDMS3)	ICON0025
.	.
.	.
CALL BOX7 (IF(NI8),F(NF13),ND10)	ICON0043
.	.
.	.
END	ICON0098
SUBROUTINE BOX7 (INUSE,CAPP,ND10)	BOX70001
C DETERMINE THE BEST LOWER BOUND.	BOX70002
.	.
.	.
DIMENSION INUSE(ND10)	BOX70013
DIMENSION CAPP(ND10)	BOX70014
.	.
.	.
END	BOX70024

associated with these arrays is modified as shown at line 1 of subroutine BØX7. The labels INUSE and CAPP are mnemonics which reflect the functions performed by these arrays within the branch-and-bound algorithm. The arrays are dimensioned at lines 13-14 of subroutine BØX7 and are manipulated within this subroutine in a manner no different from that of an array having fixed dimensions.

The dimensions ND1 through ND11 are established in subroutine BØX1 from the control parameters discussed in Appendix A, Volume 1.

The values of these dimensions are as follows:

ND1	= N	
ND2	= NSUM	
ND3	= NTC	
ND4	= M + 2	
ND5	= N + M3	
ND6	= M + 2 + N + M3	
ND7	= M + 2	(= 1 if the basis reinversion feature of the program is not used, that is if IFB = MBINV = LISTØP = 0)
ND8	= M + 2	(= 1 if the sensitivity slopes are not needed, that is if NSTRAT = 1, NBVRL1 ≥ 3 and NTITE1 = 1, or if NSTRAT = 2, NBVRL1 ≥ 3, NBVRL2 ≥ 3 and NTITE1 = NTITE2 = 1)
ND9	= N	(=1 if the program to be solved is a mixed integer linear program or a linear program, that is if ITYPE = 1 or 3)
ND10	= MXLIST	

ND11 = MXLIST

(= 1 if the LIFO node selection rule is not used, that is if NSTRAT = 1 and NØDRL1 = 1 or if NSTRAT = 2 and NØDRL1 = NØDRL2 = 0).

Array IF is subdivided into the following twelve integer arrays:

<u>Array (Dimension)</u>	<u>Description</u>
1. NZ(ND1)	The number of nonzero entries in the constraint matrix by column
2. NP(ND1)	Pointers marking the beginning of each column for arrays IR and IA
3. IR(ND2)	Row index for a nonzero entry in the constraint matrix
4. IA(ND2)	Pointer to the appropriate constant (in the table of constants) for a nonzero entry in the constraint matrix
5. INT(ND1)	Markers for integer variables (0 = not integer; 1 = first integer variable; 2 = second integer variable; etc.)
6. ICC(ND1)	Markers for concave variables (0 = not concave; 1 = first concave variable; 2 = second concave variable; etc.)
7. IS(ND4)	Temporary storage used in the transfer of one column of the constraint matrix (corresponds to the data in array IR)
8. INUSE(ND10)	Indicator for an entry in the branch-and-bound list (0 = not in use; positive integer = node number of active node in the list; negative integer = node number of active node in the sublist)
9. NV(ND6)	Temporary storage
10. IBV(ND4)	Current list of basic variables in a subprogram
11. NBV(ND5)	Current list of nonbasic variables in a subprogram

<u>Array (Dimension)</u>	<u>Description</u>
12. IUPPER(ND5)	Upper bound indicator for nonbasic variables given in array NBV (0 = nonbasic variable is at lower bound; 1 = nonbasic variable is at upper bound)

Array F is subdivided into the following 24 floating point arrays:

<u>Array (Dimension)</u>	<u>Description</u>
1. TC(ND3)	The table of constants for the constraint matrix
2. BØRIG(ND4)	The original right-hand-side vector specified in the program input
3. RHS(ND4)	The initial right-hand-side vector for a subprogram
4. C2(ND1)	The coefficients of the program objective function for variables which enter linearly (i.e., objective function values for concave variables are provided through subroutine GETØBJ)
5. C1(ND5)	Objective function coefficients used in phase 1 of the linear programming solution of the first subprogram
6. BI(ND4)	The current values of basic variables in a subprogram
7. BN(ND5)	The current values of nonbasic variables in a subprogram
8. U(ND6)	The upper limits on the variables in a subprogram
9. PJ(ND4)	Temporary storage used (together with array IS) in the transfer of one column of the constraint matrix
10. BINV(ND7,ND7)	Temporary storage used to develop the basis inverse when the basis reinversion feature of the program is exercised
11. XJ(ND4)	The updated column of the entering variable in the subprogram

<u>Array (Dimension)</u>	<u>Description</u>
12. XNØT(ND1)	The current best solution in the branch-and-bound algorithm
13. CAPP(ND10)	The lower bounds associated with nodes saved in the branch-and-bound list
14. CAPL(ND11)	The processing order associated with nodes saved in the branch-and-bound list for use in the LIFO node selection rule
15. SIGMAL(ND6)	The lower limits on the variables for a subprogram
16. SIGMAU(ND6)	The upper limits on the variables for a subprogram
17. V(ND6)	Temporary storage (used in conjunction with array NV)
18. XZ(ND6)	The solution to a subprogram adjusted for lower bound constraints
19. SO(ND8)	The "left" sensitivity slopes associated with the basic variables in a subprogram solution
20. S1(ND8)	The "right" sensitivity slopes associated with the basic variables in a subprogram solution
21. SLØLD(ND6)	The lower limits on the variables for the branching node
22. SUØLD(ND6)	The upper limits on the variables for the branching node
23. C2ØLD(ND9)	The coefficients of the program objective function for the branching node
24. B(ND4,ND4)	The current basis inverse in a subprogram

The pointers NI1 through NI12 are determined in subroutine BØX1 so that the elements IF(NI1) through IF(NI12) of array IF correspond to the

first elements of arrays NZ through IUPPER. This is done, in a natural fashion, by setting $NI1 = 1$, $NI2 = NI1 + ND1$ where $ND1$ is the dimension of array NZ, $NI3 = NI2 + ND1$ where $ND1$ is the dimension of array NP, and so forth. The pointers NF1 through NF24 are similarly determined so that the elements $F(NF1)$ through $F(NF24)$ of array F correspond to the first elements of arrays TC through B.

APPENDIX C
BRANCH-AND-BOUND LIST

The branch-and-bound list is discussed in this appendix. The list contains a record of the active nodes in the branch-and-bound tree together with the data required to characterize each such node. Associated with the list is the data required by the branching node selection rule or rules to manipulate the list. The record of the active nodes and the branching node selection data are maintained in core storage, while the data required to characterize each node are maintained in random access mass storage.

The items of information which are maintained in core storage are the following:

- (i) The number of nodes in the list (parameter NLIST);
- (ii) The number of nodes in the sublist (parameter NLISTS);
- (iii) The node number for each active node (array INUSE);
- (iv) The lower bound for each active node, used by the priority node selection rule (array CAPP); and
- (v) The processing order for each active node, used by the LIFO node selection rule (array CAPL).

The items of information which are saved in random access mass storage for each active node are the following:

- (i) The lower and upper limits on the variables which serve to characterize a node (arrays SIGMAL and SIGMAU);
- (ii) Branching variable selection data (parameters IBVR1, IUPDN1, XBRVR1, IBVR2, IUPDN2, XBRVR2);
- (iii) Limit tightening information (parameter Z, arrays XZ, SO, S1); and
- (iv) The optimal tableau associated with the subprogram (parameters TSIG, L10, NITER, NBINV, M7, IPHASE, NPHASE, NM3M7, arrays IBV, NBV, IUPPER, C2, B).

Depending upon the program solution strategy, the program type, and the branch-and-bound list option, certain of these items need not be saved and are omitted from the list. If only one solution strategy is to be used in the solution of the program (NSTRAT = 1), the data associated with the second solution strategy (parameters IBRVR2, IUPDN2, XBRVR2) are omitted from the list. If limit tightening is not included in the program solution strategy (NSTRAT = 1 and NTITE1 = 1, or NSTRAT = 2 and NTITE1 = NTITE2 = 1), then the data associated only with limit tightening (arrays XZ, SO, S1) are omitted from the list. If the program is a mixed integer linear program (ITYPE = 1), the objective function coefficients (array C2) are the same for all nodes and are omitted from the list. If the branch-and-bound list option indicates that the basis inverse should not be included in the list (LISTOP = 1), then the basis inverse matrix associated with the optimal tableau for a node (array B) is not saved in the list. The basis reinversion feature is then used to regenerate the basis inverse matrix in the event that this node is selected for branching.

The balance of this appendix discusses the mechanics associated with manipulating that portion of the branch-and-bound list which is maintained in random access mass storage. Four integer arrays are associated with the data to be saved in the list. These are:

NV(ND6)

IBV(ND4)

NBV(ND5)

IUPPER(ND5).

The eleven integer parameters

IBRVR1

IUPDN1

IBRVR2

IUPDN2

L10

NITER

NBINV

M7

IPHASE

NPHASE

NM3M7

are transferred into the last eleven locations of array NV. The integer parameters and arrays to be saved in the list thus occupy

$$NDMS2 = ND4 + 2 \cdot ND5 + 11$$

consecutive locations in core storage, all within the basic integer array IF. An auxiliary array IMS, having variable dimension NDMS2, is used to refer to this integer data. The first location of array IMS within the array IF is positioned eleven locations before the first location of array IBV. Setting

$$NIMS2 = NI10 - 11,$$

IMS(1) corresponds to IF(NIMS2).

Eight floating point arrays are associated with the data to be saved in the list. These are:

V(ND6)
XZ(ND6)
SO(ND8)
S1(ND8)
SLØLD(ND6)
SUØLD(ND6)
C2ØLD(ND9)
B(ND4,ND4).

The contents of arrays SIGMAL, SIGMAU and C2 are transferred into the arrays SLØLD, SUØLD and C2ØLD (respectively). The four floating point parameters

Z
TSIG
XBRVR1
XBRVR2

are transferred into the last four locations of array V. The floating point parameters and arrays to be saved in the list thus occupy

$$NDMS3 = (ND4)^2 + 3 \cdot ND6 + 2 \cdot ND8 + ND9 + 4$$

consecutive locations in core storage, all within the basic floating point array F. An auxiliary array FMS, is used to refer to this floating point data. The first location of array FMS within the array F is positioned four locations before the first location of array XZ.

Setting

$$NFMS3 = NF18 - 3,$$

FMS(1) corresponds to F(NFMS3).

The definitions of the dimensions NDMS2, NDMS3 and the pointers NIMS2, NFMS3 just given correspond to the maximum possible list size. In the event that selected items of data are omitted from the branch-and-bound list, these dimensions and pointers are modified to reflect any such omissions.

Tape units 2 and 3 are the areas of mass storage occupied by the branch-and-bound list. Three CDC 6700 system subroutines are used to establish a random access structure for these areas of mass storage (subroutine OPENMS), to transfer the data for the branching node from mass storage into core storage (subroutine READMS), and to transfer the data for an active node from core storage into mass storage (subroutine WRITMS). Exhibit 3 illustrates the use of these subroutines in the manipulation of the branch-and-bound list. At line 22 of subroutine ICØN, a call to subroutine OPENMS establishes the random access structure for tape unit 2. Array IMASS2 is used to store subindices or pointers for 1000 "compartments" located within this area of mass storage. The call to subroutine OPENMS at line 23 of subroutine ICØN establishes a similar structure for tape unit 3 using array IMASS3 to store subindices. Subroutine BØX1, called at lines 24-25 of subroutine ICØN, sets the values for dimensions NDMS2, NDMS3 and pointers NIMS2, NFMS3 for the particular program being solved. Subroutine BØX12, called at lines 49-51 of subroutine ICØN, selects the branching node from the branch-and-bound list (or sublist). The compartment in which

EXHIBIT 3

MANIPULATION OF THE BRANCH-AND-BOUND LIST

```

SUBROUTINE ICON (IF,F,NI,NF)
C BRANCH-AND-BOUND ALGORITHM FOR THE INTEGER CONCAVE PROGRAM.
COMMON/PO/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12,
1 NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11,
2 NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22,
3 NF23,NF24,NFMS3
.
.
COMMON/AD/IMASS2(1001),IMASS3(1001)
DIMENSION IF(NI),F(NF)
.
.
CALL OPENMS (2,IMASS2,1001,0)
CALL OPENMS (3,IMASS3,1001,0)
100 CALL BOX1 (IF,F,NI,NF,ND1,ND2,ND3,ND4,ND5,ND6,ND7,ND8,ND9,ND10,
1 ND11,NDMS2,NDMS3)
.
.
160 CALL BOX12 (IF(NI8),IF(NIMS2),F(NF4),F(NF13),F(NF14),F(NF15),
1 F(NF16),F(NF21),F(NF22),F(NF23),F(NFMS3),ND1,ND6,
2 ND9,ND10,ND11,NDMS2,NDMS3)
.
.
240 CALL BOX25 (IF(NI8),IF(NIMS2),F(NF4),F(NF13),F(NF14),F(NF15),
1 F(NF16),F(NF21),F(NF22),F(NF23),F(NFMS3),ND1,ND6,
2 ND9,ND10,ND11,NDMS2,NDMS3)
.
.
END

SUBROUTINE BOX12 (INUSE,IMS,C2,CAPP,CAPL,SIGMAL,SIGMAU,SLOLD,
1 SUOLD,C2OLD,FMS,ND1,ND6,ND9,ND10,ND11,NDMS2,
2 NDMS3)
C SELECT THE BRANCHING NODE FROM THE SUBLIST (PHASE 1) OR THE LIST
C (PHASE 2).
.
.
DIMENSION INUSE(ND10),IMS(NDMS2)
DIMENSION C2(ND1),CAPP(ND10),CAPL(ND11),SIGMAL(ND6),SIGMAU(ND6),
1 SLOLD(ND6),SUOLD(ND6),C2OLD(ND9),FMS(NDMS3)
.
.
CALL READMS (2,IMS,NDMS2,I0)
CALL READMS (3,FMS,NDMS3,I0)
.
.
END

```

```

ICON0001
ICON0002
ICON0003
ICON0004
ICON0005
ICON0006
.
.
ICON0017
ICON0018
.
.
ICON0022
ICON0023
ICON0024
ICON0025
.
.
ICON0049
ICON0050
ICON0051
.
.
ICON0086
ICON0087
ICON0088
.
.
ICON0098
BOX10001
BOX10002
BOX10003
BOX10004
BOX10005
.
.
BOX10016
BOX10017
BOX10018
.
.
BOX10065
BOX10066
.
.
BOX10104

```

EXHIBIT 3 (Continued)

```

SUBROUTINE BOX25 (INUSE,IMS,C2,CAPP,CAPL,SIGMAL,SIGMAU,SLOLD,      BOX20001
1          SUOLD,C2OLD,FMS,ND1,ND6,ND9,ND10,ND11,NOMS2,          BOX20002
2          NOMS3)                                                BOX20003
C STORE NODE LNODE IN THE SUBLIST (WHEN IN PHASE 1 AND LOWER BOUND LESS BOX20004
C THAN BEST UPPER BOUND) OR IN THE LIST (OTHERWISE).          BOX20005
.
.
.
DIMENSION INUSE(ND10),IMS(NOMS2)                                BOX20016
DIMENSION C2(ND1),CAPP(ND10),CAPL(ND11),SIGMAL(ND6),SIGMAU(ND6), BOX20017
1          SLOLD(ND6),SUOLD(ND6),C2OLD(ND9),FMS(NOMS3)          BOX20018
.
.
.
360 CALL WRITMS (2,IMS,NOMS2,I0)                                BOX20145
CALL WRITMS (3,FMS,NOMS3,I0)                                    BOX20146
.
.
.
END                                                                BOX20195
```

the data for the branching node is stored (parameter IO) is determined. At line 65 of subroutine BØX12, subroutine READMS is called to transfer the data contained in compartment IO on tape unit 2 into the array IMS in core storage. At line 66 of subroutine BØX12, a similar call of subroutine READMS transfers the data contained in compartment IO on tape unit 3 into the array FMS. Subroutine BØX25, called at lines 86-88 of subroutine ICØN, stores the data for an active node in the branch-and-bound list (or sublist). An available compartment is found (parameter IO) and, at lines 145-146 of subroutine BØX25, two calls to subroutine WRITMS transfer the data contained in arrays IMS and FMS in core storage into compartment IO on tape unit 2 and compartment IO on tape unit 3 (respectively).

APPENDIX D

SUBPROGRAM REPRESENTATION AND SOLUTION

The representation and solution of the subprograms generated by the branch-and-bound algorithm are discussed in this appendix. The data associated with a subprogram (a linear program) are:

- (i) A, the $M \times N$ matrix of constraint coefficients;
- (ii) RHS, the right-hand-side vector;
- (iii) C1, the vector of coefficients for the (linear programming) phase 1 objective function;
- (iv) C2, the vector of coefficients for the (linear programming) phase 2 objective function; and
- (v) U, the vector of simple upper bounds on the program variables.

Figure 3 displays this data in the form in which it is used in the computer program.

Specific labeling conventions are adopted for the rows and columns of the linear program. The $(M + 2)$ rows are ordered as follows:

Less than or equal constraints	(M1 rows)
Equality constraints	(M2 rows)
Greater than or equal constraints	(M3 rows)
Phase 2 objective function	(1 row)
Phase 1 objective function	(1 row).

The $(N + M + M3 + 2)$ columns of the linear program are ordered as follows:

Program variables	(N columns)
Surplus variables	(M3 columns)
Slack variables	(M1 columns)
Artificial variables (equality constraints)	(M2 columns)

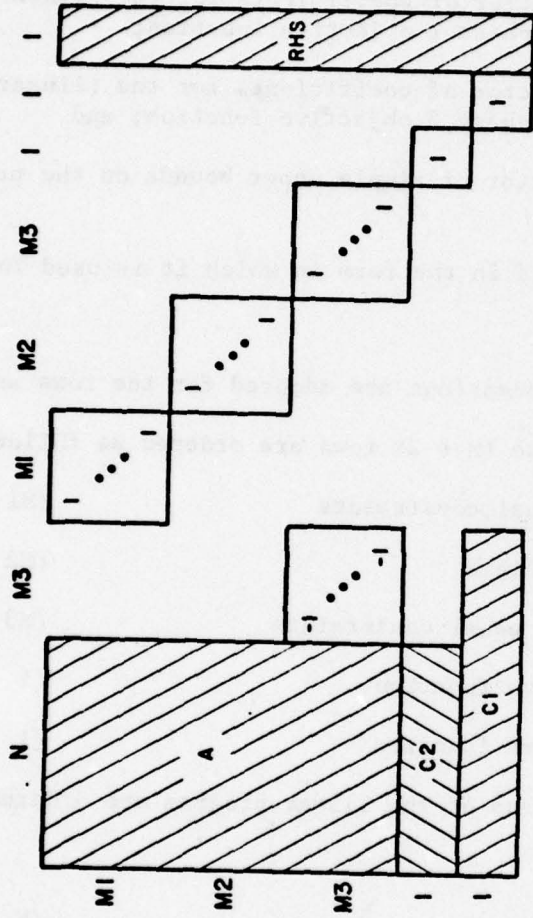


FIGURE 3
SUBPROGRAM REPRESENTATION

Artificial variables (greater than or equal constraints)	(M3 columns)
Phase 2 objective function	(1 column)
Phase 1 objective function	(1 column)
Right-hand-side	(1 column).

The constraint matrix A is represented within the computer program in a form which is commonly used for large linear programs. The density of nonzero entries in the constraint matrix is frequently low for such programs. Moreover, among the nonzero entries, particular values (such as 1 or - 1) may recur often. Maintaining the entire M x N dimensional constraint matrix in core storage is wasteful for matrices of this type. An alternative means of representing the constraint matrix begins by establishing a table of constants which lists the distinct values which occur in the matrix. For each column of the matrix, the following information is then recorded:

- (i) The number of nonzero entries;
- (ii) For each nonzero entry, the row number in which the entry occurs; and
- (iii) For each nonzero entry, a pointer which indicates the appropriate value in the table of constants.

In the computer program, the array TC corresponds to the table of constants, the array NZ corresponds to the number of nonzero entries by column, the array IR contains the row numbers in which nonzero entries occur, and the array IA contains the corresponding pointers to the table of constants. An auxiliary array NP is defined by the relations $NP(1) = 0$ and

$$NP(J) = \sum_{I=1}^{J-1} NZ(I)$$

for $J = 2, N$. This array provides a set of pointers for use in conjunction with arrays IR and IA. The data corresponding to the J-th column of the constraint matrix is then stored in the locations $NP(J) + 1$ through $NP(J) + NZ(J)$ of arrays IR and IA. For extremely large programs, the integer arrays NZ, NP, IR and IA could be maintained in core storage by packing them.

The linear programming code used in this branch-and-bound algorithm utilizes the revised simplex method with simple upper bounds on the variables. This version of the simplex method is discussed in Lasdon (1970) and in Garfinkel and Nemhauser (1972). Both the primal and the dual simplex algorithms are included. The primal algorithm operates either as a one phase method or as a two phase method depending upon the requirements of the program being solved. Associated with this linear programming code is a basis inversion feature which can be used to combat the accumulation of round-off error associated with the simplex pivot operations.

The data associated with the current tableau in a subprogram are:

- (i) IBV, the vector of indices of basic variables;
- (ii) NBV, the vector of indices of nonbasic variables;
- (iii) IUPPER, the vector of indicators of nonbasic variables at upper bound;
- (iv) BI, the vector of values of basic variables;
- (v) BN, the vector of values of nonbasic variables; and

(vi) B, the basis inverse.

These arrays are updated at each simplex iteration. Note that

$$BI = B(RHS - \sum_j U_j \cdot P_j)$$

where the summation is taken over all nonbasic variables j which are at upper bound and where P_j denotes the corresponding column in Figure 3.

Also, BN can be obtained directly from IUPPER and U. As a consequence, BI and BN need not be saved in the branch-and-bound list.

The initial feasible basis for the first subprogram consists of slack and artificial variables unless the user elects to provide a basis as a part of the branch-and-bound input. A phase 1 objective function is established if there are artificials in the initial basis. The basis is primal feasible if the initial values of the basic variables are nonnegative and do not exceed their upper bounds. In this case, the primal algorithm is applied to solve subprogram 1, using either a one phase method (if there are no artificials in the basis) or a two phase method (otherwise). If the initial basis is not primal feasible, the reduced costs are examined to see if they are nonnegative, in which case the initial basis is dual feasible. In this case, the dual simplex algorithm is applied to solve subprogram 1. In the event that there are artificials in the initial basis, the phase 1 objective function is treated as a constraint so that the values of these artificials are forced to be zero in the solution. When the solution to subprogram 1 is reached, the optimal basis is tested to see if it contains artificial variables. If no artificials are in this basis, the row and column

corresponding to the phase 1 objective function are deleted from the tableau. If there are artificials remaining in the basis, these necessarily have value zero. However, the phase 1 objective function must be maintained in the tableau as a constraint to assure that these artificials cannot assume positive values in the solutions to subsequent subprograms.

Linear programming sensitivity analysis is applied to solve the subprograms after the first subprogram. This allows the solution of a subprogram to proceed from the optimal tableau associated with the branching node. If the basis inverse is not included in the branch-and-bound list, then it is regenerated using the basis inversion feature of the linear programming code. The simple upper bounds U are computed to be the difference between the lower and upper limits $SIGMAL$ and $SIGMAU$ (respectively) on the variables. The initial values BI of the basic variables are computed by adjusting RHS for variables at upper bound and then multiplying by the basis inverse. As a result of the new constraint added for the branching variable and of the tightening of lower and upper limits for the basic variables, this tableau may be primal infeasible. Since the initial feasible point was the optimal solution for the branching node, this tableau is dual feasible. The dual simplex algorithm is applied to generate an optimal solution under the new constraints. The cost coefficients $C2$ are next updated to reflect any changes made to the lower and upper limits for concave variables. When the cost coefficients change, the reduced costs appearing in the tableau

also change resulting in a tableau which is not in canonical form. The basis inverse and values of the basic variables are modified to re-establish the canonical form of the tableau, which then is primal feasible but not optimal. The primal algorithm is applied to generate an optimal solution for the new cost data. An adequate presentation of the linear programming sensitivity analysis techniques used here may be found in Hillier and Lieberman (1967).

APPENDIX E
SUBROUTINES

The subroutines used in the ICØN branch-and-bound algorithm are discussed in this appendix. The algorithm consists of 29 subroutines. Three of these (MAIN, READIN and GETØBJ are provided by the user for each program to be solved. These are discussed in Appendices B and C of Volume 1. Following is a description of each of the remaining 26 subroutines, indicating the function performed by the subroutine in the algorithm and any notable features of the subroutine. The nomenclature selected for subroutines BØX1 through BØX25 corresponds to the labels which appear in Figure 2, the general flow schematic for the algorithm. Figure 4 shows the number of times one of these subroutines calls or is called by another subroutine.

<u>Subroutine</u>	<u>Description</u>
ICØN	This is the master subroutine which calls the various subroutines in the sequence specified in the general flow schematic for the branch-and-bound algorithm. The initiation of a program solution from previously prepared restart tapes and the preparation of restart tapes when the program time limit is reached are accomplished by calling subroutine RSTART.
BØX1	The branch-and-bound input and the user input are accepted by this subroutine by means of calls to subroutines INPUT1, INPUT2, INPUT3 and READIN. The dimensioning of arrays and the structuring of the branch-and-bound list are done in BØX1.
BØX2	This subroutine initializes the data used in the branch-and-bound algorithm. The initial basis, basis inverse and values of the basic variables are established by a call to subroutine INPUT4. The algorithm to be used to solve the first subprogram is determined by subroutine INPUT5. The first subprogram is then solved.

CALLING SUBROUTINE

	MAIN	ICON	BØX1	BØX2	BØX5	BØX7	BØX10	BØX12	BØX13	BØX15	BØX17	BINVRT	INPUT4	INPUT5	SIMPLE	SLOPES
ICON	1															
BØX1		1														
BØX2		1														
BØX5		2														
BØX7		1														
BØX10		1														
BØX12		1														
BØX13		1														
BØX15		1														
BØX17		1														
BØX23		1														
BØX25		1														
ADJUST							1									
BINVRT												1		1	1	
GETCØL				1			3		3	1	1	3	1			
INPUT1			1													
INPUT2			1													
INPUT3			1													
INPUT4				1												
INPUT5				1												
INVERT												1				
ØBJ1													2			
RSTART		2														
SIMPLE				1			2									
SLOPES				1			1									
TIMEC		1	2		1	1										
READIN			1													
GETØBJ				2		5	2	7								

SUBROUTINE CALLED

FIGURE 4

CALLS TO ICON SUBROUTINES

<u>Subroutine</u>	<u>Description</u>
BØX5	The solution to the problem at the end of the branch-and-bound computation is printed by this subroutine. It also prints the current best solution in the event that the program time limit is reached and re-start tapes are prepared.
BØX7	This subroutine computes the current best lower bound as required by the variable best upper bound method.
BØX10	This subroutine selects a node from the branch-and-bound list and places it in the sublist as required by the variable best upper bound method.
BØX12	The branching node is selected from the sublist (in phase 1 of the variable best upper bound method) or from the list (in phase 1 of any other solution method and in phase 2). The data for the branching node is read in from random access mass storage (using subroutine READMS) if this data is not already in core. The branching variable selection is adjusted according to the current phase of the branch-and-bound algorithm.
BØX13	This subroutine tightens the limits on monotone variables during phase 1 and tightens the limits on all basic variables using the best upper bound during phase 2. The latter function is accomplished by calling the auxiliary subroutine ADJUST.
BØX15	The subprograms associated with the two nodes obtained from the branching node are solved in this subroutine. The tableau associated with the branching node is modified to reflect any new lower and upper limits on the variables. The modified tableau is the starting point for the solution of both subprograms and is saved on tape unit 4 for use in the second subprogram solution. The first subprogram is solved using the dual simplex and primal algorithms, as required. The information on tape unit 4 is read in before the second subprogram is solved.
BØX17	This subroutine determines the lower bound for the node and selects the branching variable. If one of the most noninteger branching variable selection rules (rules 3 or 4) is being used for a mixed

SubroutineDescription

integer linear program (ITYPE = 1), or if the conventional branching variable selection rule (rule 5) is being used for a concave program (ITYPE = 2), the lower bound is taken to be the objective function value Z. If the maxmin or maxmax branching variable selection rules (rules 0, 1 or 2) are being used, the stronger lower bound corresponding to the maxmin penalty is used. If it was determined in subroutine INPUT3 that the objective function is integer valued, then this lower bound is rounded up. When a two strategy method is being used, two branch-variable selections are made.

BØX23

The best upper bound is updated in this subroutine. The branch-and-bound list is edited to delete any nodes for which the lower bound exceeds this new best upper bound. If the algorithm was in phase 1, then phase 2 is entered and the sublist is merged into the list when the variable best upper bound method is being used.

BØX25

The current node is saved in the sublist (in phase 1 of the variable best upper bound method when the lower bound is less than the best upper bound) or in the list (otherwise). The data for the node is placed in random access mass storage (using subroutine WRITMS). In the event that the maximum list size is exceeded, the node having the greatest lower bound is erased from the list to make room for the new node.

ADJUST

This subroutine is called by subroutine BØX13 to adjust the lower and upper limits on a variable using sensitivity analysis information together with the best upper bound.

BINVRT

Basis inversion or reinversion is accomplished by this subroutine. The basis matrix consists of those columns P_j from Figure 3 for the basic variables j (specified in array IBV). The corresponding right-hand-side vector is

$$\text{RHS} = \sum_j U_j \cdot P_j$$

SubroutineDescription

where the summation is taken over all nonbasic variables j which are at upper bound (specified in arrays NBV and IUPPER). Subroutine INVERT is called to determine the basis inverse matrix and the corresponding values of the basic variables.

GETCØL	This subroutine places the J-th column from the tableau shown in Figure 3 into the array PJ which is assumed to have been set to zero prior to calling the subroutine. The number of nonzero entries in the column and indicators as to which entries are nonzero are returned using parameter NZERØS and array IS.
INPUT1	The number of less than or equal, equality, and greater than or equal constraints are read and the parameters in CØMMØN/P2/ are initialized. From the number of nonzero entries by column (array NZ), the column pointers are developed (array NP).
INPUT2	The arrays IR and IA associated with the constraint matrix representation are read, along with the number of entries in the table of constants (parameter NTC).
INPUT3	The remainder of the data associated with the constraint matrix, the table of constants and the right-hand-side vector, are read. The lower and upper limits on variables are read in when values other than the default values of 0 and $\div \infty$ (respectively) are desired. The cost data and lists of integer and/or concave variables are read in next. For a mixed integer linear program (ITYPE = 1), a determination is made if the objective function is integer valued. This information is used in subroutine BØX17 to round lower bounds on the nodes.
INPUT4	Called by subroutine BØX2, this subroutine establishes the initial basis, basis inverse and right-hand-side for the first subprogram. If the initial feasible basis is a part of the program input (IFB = 1 on the first control card), then this basis is read in and the basis inversion feature is used to establish the corresponding basis inverse and right-hand-side. Otherwise, the initial feasible basis consists of slack and arti-

SubroutineDescription

	ificial variables with the basis inverse being an identity matrix and the initial right-hand-side agrees with the input right-hand-side vector. In either case, if there are artificials in the initial basis, subroutine ØBJ1 is called to establish the phase 1 objective function.
INPUT5	This subroutine selects the linear programming algorithm (primal or dual simplex) to be used to solve the first subprogram. The initial basis is tested for primal feasibility (nonnegative right-hand-side) and, if it is primal feasible, the primal algorithm is selected. If the initial basis is not primal feasible, it is tested for dual feasibility (nonnegative reduced cost vector) and, if it is dual feasible, the dual simplex algorithm is selected.
INVERT	The Gauss-Jordan method of matrix inversion is used to invert the basis matrix and determine the corresponding vector of values of basic variables. An ill-conditioned basis matrix, if encountered, results in the termination of computations.
ØBJ1	This subroutine computes and stores the (linear programming) phase 1 objective function associated with the tableau for the first subprogram in its canonical form.
RSTART	<p>The manipulation of job restart tapes (tape units 7 through 10) is done in this subroutine.</p> <p>If a job is to commence from previously prepared restart tapes (indicated by the setting MSTART = 1 on the second control card), subroutine ICØN calls subroutine RSTART with IENTRY = 0. Tape units 7 and 8 are read, restoring the branch-and-bound list, labeled commons and variable dimensioned arrays to their condition at the time computations were last terminated. In subroutine ICØN, the flow of the branch-and-bound computation is then reentered at the point where computations were terminated.</p> <p>The determination that job restart tapes are to be created is made in subroutine ICØN when the elapsed execution time exceeds the input maximum execution</p>

SubroutineDescription

time (parameter TIME1 on the third control card). The current values of labeled commons (COMMON/P2/, /P3/ and /P4/) and variable dimensioned arrays are saved on tape unit 9. The branch-and-bound list together with intermediate data on tape unit 4 (if any) are saved on tape unit 10. Note that the data saved on tape unit 9 corresponds to the data read from tape unit 7 on a subsequent restart run. Similarly, tape unit 10 corresponds to tape unit 8.

SIMPLE

This subroutine contains the primal and dual simplex algorithms. Basis reinversion is used as a technique for suppressing round-off error in the simplex computations. The frequency of basis reinversion is specified by parameter MBINV on the first control card. In the primal algorithm, the entering variable is taken as the first nonbasic variable encountered having negative reduced cost. As compared with the usual rule in which all reduced costs are computed with the entering variable corresponding to the minimum reduced cost, this frequently results in less total computation in reaching the optimum. A common pivot logic is used for both the primal and dual simplex algorithms. Note that the phase 1 objective function is deleted when no artificials remain in the basis.

The computations in this subroutine can terminate when any one of the following determinations is made:

- (i) The current solution is optimal;
- (ii) The primal program is infeasible;
- (iii) The primal program is unbounded; or
- (iv) The dual value exceeds the best upper bound (for the dual simplex algorithm).

Unboundedness of the primal program is impossible since the lower and upper bounds define a compact set; however, this form of termination can occur due to the accumulation of round-off errors. Because the dual value in the dual simplex algorithm is monotone nondecreasing, comparison against the

<u>Subroutine</u>	<u>Description</u>
	best upper bound is a useful way of saving unnecessary computation.
SLØPES	This subroutine computes the sensitivity analysis slopes which are required by the branch-and-bound algorithm for penalty analyses. These are computed from the optimal tableau using the dual simplex entering variable selection criteria.
TIMEC	A printout of the elapsed job execution time is obtained by calling this subroutine.

There are six CDC 6700 system subroutines which are used in conjunction with the ICØN algorithm. Only three of these (ØPENMS, READMS and WRITMS) are nonstandard. These are described in Appendix C.

Figure 5 shows the number of times each system routine is called by some subroutine in the algorithm.

		CALLING SUBROUTINE							
		MAIN	ICON	BOX1	BOX12	BOX25	INVERT	RSTART	TIMEC
SUBROUTINE CALLED	OPENMS	2							
	READMS			2			2		
	WRITMS				2		2		
	SECND		1	1					1
	EOF			1					
	EXIT		1	3			1	1	

FIGURE 5
CALLS TO SYSTEM SUBROUTINES

APPENDIX F

LISTING OF THE ICON ALGORITHM

This appendix presents a listing of the 26 basic subroutines which comprise the ICON branch-and-bound algorithm. Not shown in Exhibit 4 are the three user provided subroutines (MAIN, READIN and GETOBJ) discussed in Appendix B, Volume 1.

EXHIBIT 4

LISTING OF THE ICON ALGORITHM

```

SUBROUTINE ICON (IF,F,NI,NF)
C BRANCH-AND-BOUND ALGORITHM FOR THE INTEGER CONCAVE PROGRAM.
COMMON/PC/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12,
1 NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11,
2 NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22,
3 NF23,NF24,NFMS3
COMMON/P1/N,M,ITYPE,NSTRAT,NOCRL1,NBVR1,NTITE1,NODRL2,NBVR2,
1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
COMMON/P2/EPSI,EPSTM,BIGN,BEGTH,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
1 NM1M3,NIP2,NP1,NSUM,NTC,M10
COMMON/P3/NCONOT,LNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
1 NLISTS,NFEAS,LSTMX,IIRT(T,ITRMA),BLB,NBRNOD,PBRNOD,
2 NBRVAR,NUPDOWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BCUNDU,
3 TSIG,IFEAS,IBRVR1,IUPCN1,XBRVR1,IBRVR2,IUPCN2,XBRVR2,
4 L10,NITER,NBINV,M7,I PHASE,NPHASE,NM3M7,IALGO,IECJ
COMMON/A0/IMASS2(1001),IMASS3(1001)
DIMENSION IF(NI),F(NF)
EPSI=1.0E-11
EPSTM=-EPSI
BIGN=1.0E+100
CALL OPENMS (2,IMASS2,1001,0)
CALL OPENMS (3,IMASS3,1001,0)
100 CALL BOX1 (IF,F,NI,NF,ND1,ND2,ND3,ND4,ND5,ND6,ND7,ND8,ND9,ND10,
1 ND11,NCMS2,NDMS3)
IF(MSTART.EQ.0)GOTO110
CALL RSTART (IF,IF(NI8),IF(NI10),F(NI11),IF(NI12),IF(NIMS2),
1 F,F(NF6),F(NF7),F(NF24),F(NFMS3),NI,NF,ND4,ND5,
2 ND10,NCMS2,NDMS3,0)
GOTO170
110 CALL BOX2 (IF(NI1),IF(NI2),IF(NI3),IF(NI4),IF(NI5),IF(NI6),
1 IF(NI7),IF(NI8),IF(NI9),IF(NI10),IF(NI11),IF(NI12),
2 F(NF1),F(NF2),F(NF3),F(NF4),F(NF5),F(NF6),F(NF7),
3 F(NF8),F(NF9),F(NF10),F(NF11),F(NF12),F(NF15),F(NF16),
4 F(NF17),F(NF18),F(NF19),F(NF20),F(NF24),ND1,ND2,ND3,
5 ND4,ND5,ND6,ND7,ND8,ND10)
LNODE=2
GOTO190
120 IF(NLIST.NE.0)GOTO130
CALL BOX5 (IF(NI8),F(NF12),F(NF13),F(NF14),ND1,ND10,ND11,0)
GOTO100
130 IF(IBUBOP.EQ.0)GOTO160
CALL BOX7 (IF(NI8),F(NF13),ND10)
IF(BLB.LT.UNOT)GOTO140
UNOT=BLB + PCBUB
IF(IOUTPT.NE.0)WRITE(6,1000)UNOT
140 CALL BOX10 (IF(NI8),F(NF13),F(NF14),ND10,ND11)
150 IF(NLISTS.EQ.0)GOTO120
160 CALL BOX12 (IF(NI8),IF(NIMS2),F(NF4),F(NF13),F(NF14),F(NF15),
1 F(NF16),F(NF21),F(NF22),F(NF23),F(NFMS3),ND1,ND6,
2 ND9,ND10,ND11,NDMS2,NDMS3)
CALL BOX13 (IF(NI5),IF(NI6),IF(NI10),F(NF4),F(NF15),F(NF16),
1 F(NF18),F(NF19),F(NF20),F(NF21),F(NF22),ND1,ND4,ND6,

```

EXHIBIT 4 (Continued)

2	NO8)	ICON0054
	LNODE=1	ICON0055
170	CALL SECOND (TIME2)	ICON0056
	IF (TIME2.LT.BEGTM+TIME1)GOTO180	ICON0057
	CALL BOX5 (IF(NI8),F(NF12),F(NF13),F(NF14),ND1,ND10,ND11,1)	ICON0058
	CALL RSTART (IF,IF(NI8),IF(NI10),IF(NI11),IF(NI12),IF(NIMS2),	ICON0059
1	F,F(NF6),F(NF7),F(NF24),F(NFMS3),NI,NF,ND4,ND5,	ICON0060
2	ND10,NDMS2,NCMS3,1)	ICON0061
	GOTO100	ICON0062
180	CALL BOX15 (IF(NI1),IF(NI2),IF(NI3),IF(NI4),IF(NI5),IF(NI6),	ICON0063
1	IF(NI7),IF(NI9),IF(NI10),IF(NI11),IF(NI12),F(NF1),	ICON0064
2	F(NF2),F(NF3),F(NF4),F(NF5),F(NF6),F(NF7),F(NF8),	ICON0065
3	F(NF9),F(NF10),F(NF11),F(NF15),F(NF16),F(NF17),	ICON0066
4	F(NF18),F(NF19),F(NF20),F(NF21),F(NF23),F(NF24),	ICON0067
5	ND1,ND2,ND3,ND4,ND5,NC6,ND7,ND8,ND9)	ICON0068
190	IF (IEOJ.NE.0)GOTO250	ICON0069
	CALL BOX17 (IF(NI5),IF(NI6),IF(NI10),F(NF4),F(NF15),F(NF16),	ICON0070
1	F(NF18),F(NF19),F(NF20),ND1,ND4,ND6,NC8)	ICON0071
	IF (IBUBOP.EQ.1)GOTO210	ICON0072
	IF (BOUNDL.LT.(1.0-TOL1)*UNOT)GOTO200	ICON0073
	IF (IOUTPT.EQ.0)GOTO250	ICON0074
	WRITE(6,1001)	ICON0075
	GOTO250	ICON0076
200	IF (IFEAS.EQ.0)GOTO240	ICON0077
	IF (BOUNDU.GE.UNOT)GOTO230	ICON0078
	GOTO220	ICON0079
210	IF (IFEAS.EQ.0)GOTO240	ICON0080
220	CALL BOX23 (IF(NI8),F(NF12),F(NF13),F(NF18),ND1,ND6,ND10)	ICON0081
230	IF (BOUNDU-BOUNDL.GT.TOL1*ABS(BOUNDU))GOTO240	ICON0082
	IF (IOUTPT.EQ.0 .OR. ITYPE.EQ.1)GOTO250	ICON0083
	WRITE(6,1002)	ICON0084
	GOTO250	ICON0085
240	CALL BOX25 (IF(NI8),IF(NIMS2),F(NF4),F(NF13),F(NF14),F(NF15),	ICON0086
1	F(NF16),F(NF21),F(NF22),F(NF23),F(NFMS3),ND1,ND6,	ICON0087
2	ND9,ND10,ND11,NCMS2,NDMS3)	ICON0088
250	IF (IOUTPT.NE.0)CALL TIMEC	ICON0089
	IF (LNODE.NE.2)GOTO260	ICON0090
	IF (IBUBOP.EQ.1)GOTO150	ICON0091
	GOTO120	ICON0092
260	LNODE=2	ICON0093
	GOTO170	ICON0094
1000	FORMAT(31H0THE PHASE 1 BEST UPPER BOUND =,E15.6)	ICON0095
1001	FORMAT(46H0THE LOWER BOUND EXCEEDS THE BEST UPPER BOUND.)	ICON0096
1002	FORMAT(38H0THE LOWER AND UPPER BOUNDS ARE EQUAL.)	ICON0097
	END	ICON0098

EXHIBIT 4 (Continued)

```

SUBROUTINE BOX1 (IF,F,NI,NF,ND1,N(2,ND3,ND4,ND5,ND6,ND7,ND8,ND9,
1 ND10,ND11,NDMS2,NDMS3) BOX10001
C READ BRANCH-AND-BOUND INPUT. READ USER INPUT. BOX10002
COMMON/P0/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12, BOX10003
1 NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11, BOX10004
2 NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22, BOX10005
3 NF23,NF24,NFMS3 BOX10006
COMMON/P1/N,M,I,TYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2, BOX10008
1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT, BOX10009
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10) BOX10010
COMMON/P2/EPSI,EPSTM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2, BOX10011
1 NM1M3,N1P2,NP1,NSUM,NTC,M10 BOX10012
COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST, BOX10013
1 NLISTS,NFEAS,LSIMX,ITRTCT,ITRMAX,BL8,NBRNOD,PBRNOC, BOX10014
2 NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU, BOX10015
3 TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2, BOX10016
4 L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ BOX10017
DIMENSION IF(NI),F(NF) BOX10018
CALL SECOND (BEGTM) BOX10019
C READ THE CONTROL PARAMETERS. BOX10020
READ(5,1000)N,M,I,TYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2, BOX10021
1 NBVRL2,NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER, BOX10022
2 MBINV BOX10023
IF(EOF(5).NE.0.C)CALL EXIT BOX10024
IF(N.LE.0)CALL EXIT BOX10025
READ(5,1000)IOUTPT,ITRACE,MSTART BOX10026
IF(MXLIST.EQ.0)MXLIST=1000 BOX10027
IF(MXLIST.GT.1000)MXLIST=1000 BOX10028
IF(ITAPE.EQ.0)ITAPE=5 BOX10029
IF(MXITER.EQ.0)MXITER=1000 BOX10030
INDEX=0 BOX10031
IF(MBINV.EQ.0 .AND. IFB.EQ.0 .AND. LISTOP.EQ.0)INDEX=1 BOX10032
READ(5,1001)TIME1,TOL1,TOL2,UNOT,PCBUB BOX10033
IF(TIME1.EQ.0.C)TIME1=180. BOX10034
IF(TOL1.EQ.0.C)TOL1=EPSI BOX10035
IF(TOL2.EQ.0.C)TOL2=EPSI BOX10036
IF(UNOT.EQ.0.C .OR. PCBUB.NE.0.C)UNOT=BIGN BOX10037
IBUBOP=0 BOX10038
IF(PCBUB.NE.0.C)IBUBOP=1 BOX10039
READ(5,1002)ALPHA BOX10040
WRITE(6,1003)N,M,I,TYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2, BOX10041
1 NBVRL2,NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER, BOX10042
2 MBINV,IOUTPT,ITRACE,MSTART BOX10043
WRITE(6,1004)TIME1,TOL1,TOL2,UNOT,PCBUB BOX10044
WRITE(6,1005)ALPHA BOX10045
IF(ITRACE.GE.1)WRITE(6,1009) BOX10046
ND1=N BOX10047
NI1=1 BOX10048
NI2=NI1 + ND1 BOX10049
NI3=NI2 + NI1 BOX10050
C READ THE NUMBER OF CONSTRAINTS, THE NUMBER OF NONZERO ENTRIES IN THE BOX10051
C CONSTRAINT MATRIX (BY COLUMN), AND DEVELOP THE COLUMN POINTERS. BOX10052
CALL INPUT1 (IF(NI1),IF(NI2),ND1) BOX10053
ND2=NSUM BOX10054
NI4=NI3 + ND2 BOX10055
NI5=NI4 + ND2 BOX10056
C READ THE CONSTRAINT MATRIX COLUMN-BY-COLUMN. BOX10057

```

EXHIBIT 4 (Continued)

CALL INPUT2 (IF(NI1),IF(NI2),IF(NI3),IF(NI4),ND1,ND2)	80X10058
ND3=NTC	80X10059
ND4=M + 2	80X10060
ND5=N + M3	80X10061
ND6=N + M3 + M + 2	80X10062
IF(ND6.LT.11)ND6=11	80X10063
ND7=ND4	80X10064
IF(INDEX.EQ.1)ND7=1	80X10065
ND8=ND4	80X10066
IF(NBVRL1.LE.2 .OR. NTITE1.EQ.0)GOTO110	80X10067
IF(NSTRAT.EQ.1)GOTO100	80X10068
IF(NBVRL2.LE.2 .OR. NTITE2.EQ.0)GOTO110	80X10069
100 ND9=1	80X10070
110 ND9=ND1	80X10071
IF(ITYPE.EQ.1 .OR. ITYPE.EQ.3)ND9=1	80X10072
ND10=MXLIST	80X10073
ND11=ND10	80X10074
IF(NODRL1.EQ.1)GOTO130	80X10075
IF(NSTRAT.EQ.1)GOTO120	80X10076
IF(NODRL2.EQ.1)GOTO130	80X10077
120 ND11=1	80X10078
130 NI6=NI5 + ND1	80X10079
NI7=NI6 + ND1	80X10080
NI8=NI7 + ND4	80X10081
NI9=NI8 + ND10	80X10082
NI10=NI9 + ND6	80X10083
NI11=NI10 + ND4	80X10084
NI12=NI11 + ND5	80X10085
NITOT=NI12 + ND5	80X10086
NF1=1	80X10087
NF2=Nf1 + ND3	80X10088
NF3=Nf2 + ND4	80X10089
NF4=Nf3 + ND4	80X10090
NF5=Nf4 + ND1	80X10091
NF6=Nf5 + ND5	80X10092
NF7=Nf6 + ND4	80X10093
NF8=Nf7 + ND5	80X10094
NF9=Nf8 + ND6	80X10095
NF10=Nf9 + ND4	80X10096
NF11=Nf10 + ND7*ND7	80X10097
NF12=Nf11 + ND4	80X10098
NF13=Nf12 + ND1	80X10099
NF14=Nf13 + ND10	80X10100
NF15=Nf14 + ND11	80X10101
NF16=Nf15 + ND6	80X10102
NF17=Nf16 + ND6	80X10103
NF18=Nf17 + ND6	80X10104
NF19=Nf18 + ND6	80X10105
NF20=Nf19 + ND8	80X10106
NF21=Nf20 + ND8	80X10107
NF22=Nf21 + ND6	80X10108
NF23=Nf22 + ND6	80X10109
NF24=Nf23 + ND9	80X10110
NFTOT=Nf24 + ND4*ND4	80X10111
IF(NI.LT.NITOT .OR. NF.LT.NFTOT)GOTO230	80X10112
C READ THE TABLE OF CONSTANTS, THE RIGHT-HAND-SIDE, THE LOWER AND	80X10113
C UPPER BOUNDS, THE COST DATA, AND THE LISTS OF INTEGER AND CONCAVE	80X10114

EXHIBIT 4 (Continued)

C VARIABLES.	
CALL INPUT3 (IF(NI5),IF(NI6),IF(NI9),F(NF1),F(NF2),F(NF4),	80X10115
1 F(NF15),F(NF16),F(NF17),ND1,ND3,ND4,ND6)	80X10116
WRITE(6,1006)NITOT,NFTOT	80X10117
C ESTABLISH THE STRUCTURE OF THE BRANCH-AND-BOUND LIST.	80X10118
IF(NSTRAT.EQ.2)GOTO140	80X10119
NIMS2=NI10 - 9	80X10120
NDMS2=ND4 + 2*ND5 + 9	80X10121
GOTO150	80X10122
140 NIMS2=NI10 - 11	80X10123
NDMS2=ND4 + 2*ND5 + 11	80X10124
150 NDMS3=2*ND6 + 3	80X10125
IF(NTITE1.EQ.0)GOTO160	80X10126
IF(NSTRAT.EQ.1)GOTO170	80X10127
IF(NTITE2.EQ.1)GOTO170	80X10128
160 NFMS3=NF18 - 3	80X10129
NDMS3=NDMS3 + ND6 + 2*ND8	80X10130
GOTO180	80X10131
170 NFMS3=NF21 - 3	80X10132
180 IF(NSTRAT.EQ.1)GOTO190	80X10133
NFMS3=NFMS3 - 1	80X10134
NDMS3=NDMS3 + 1	80X10135
190 IF(LISTOP.EQ.1)GOTO200	80X10136
NDMS3=NDMS3 + ND9 + ND4*ND4	80X10137
GOTO210	80X10138
200 IF(ITYPE.EQ.1)GOTO210	80X10139
NDMS3=NDMS3 + ND9	80X10140
210 NMSTOT=NDMS2 + NDMS3	80X10141
WRITE(6,1007)NMSTOT,NDMS2,NDMS3	80X10142
IF(ITYPE.EQ.1)GOTO220	80X10143
C READ USER INPUT.	80X10144
CALL READIN	80X10145
220 CALL TIMEC	80X10146
RETURN	80X10147
230 WRITE(6,1006)NITOT,NFTOT	80X10148
WRITE(6,1008)NI,NF	80X10149
CALL TIMEC	80X10150
CALL EXIT	80X10151
RETURN	80X10152
1000 FORMAT(16I5)	80X10153
1001 FORMAT(5E12.0)	80X10154
1002 FORMAT(10A8)	80X10155
1003 FORMAT(37H1INPUT FOR BRANCH-AND-BOUND ALGORITHM/ 1 21H0INTEGER PARAMETERS =,1(I7/21X,3I7)	80X10156
1004 FORMAT(21H0REAL PARAMETERS =,5E15.6)	80X10157
1005 FORMAT(21H0PROGRAM IDENTIFIER =,4X,10A8)	80X10158
1006 FORMAT(44H0ADEQUATE DIMENSIONS FOR ARRAYS IF AND F ARE,I10, 1 4H AND,I10,16H (RESPECTIVELY).)	80X10159
1007 FORMAT(39H0THE BRANCH-AND-BOUND LIST CONSISTS OF ,I10,21H LOCATION 15 PER NODE (,I10,25H LOCATIONS ON UNIT 2 AND ,I10,22H LOCATIONS ON 2 UNIT 3).)	80X10160
1008 FORMAT(15H0THE DIMENSIONS,I10,4H AND,I10,18H ARE INSUFFICIENT.)	80X10161
1009 FORMAT(10H *****BCX1)	80X10162
END	80X10163
	80X10164
	80X10165
	80X10166
	80X10167
	80X10168

EXHIBIT 4 (Continued)

```

SUBROUTINE BOX2 (NZ, NP, IR, IA, INT, ICC, IS, INUSE, NV, IBV, NBV, IUPPER, 80X20001
1 TC, BORIG, RMS, C2, C1, BI, BN, U, PJ, BINV, XJ, XNOT, 80X20002
2 SIGNAL, SIGMAU, V, XZ, S0, S1, B, ND1, ND2, ND3, ND4, ND5, 80X20003
3 ND6, ND7, ND8, ND10) 80X20004
C SOLVE THE FIRST SUBPROGRAM. 80X20005
COMMON/P1/N, H, ITYPE, NSTRAT, NOORL1, NBVRL1, NTITE1, NOORL2, NBVRL2, 80X20006
1 NTITE2, MXLIST, LISTOP, ITAPE, IFB, MXITER, MBINV, IOUTPT, 80X20007
2 ITRACE, MSTART, TIME1, TOL1, TOL2, PCBUB, ALPHA (10) 80X20008
COMMON/P2/EPS1, EPSIM, ZIGN, BEGTM, M1, M2, M3, M4, N1, MP1, MP2, NM3, NM1M2, 80X20009
1 NM1M3, N1P2, NP1, NSUM, NTC, M10 80X20010
COMMON/P3/NOONOT, UNOT, IBUBOP, LPHASE, NOORUL, NBVRUL, NTIGHT, NLIST, 80X20011
1 NLISTS, NFEAS, LSTMX, ITRTCT, ITRMAX, BLB, NBRNOD, PBRNOD, 80X20012
2 NBRVAR, NUPOWN, XBRNOD, TBRNOD, NODE, LNODE, Z, EOUNDL, BCUNDU, 80X20013
3 TSIG, IFEAS, IBRVR1, IUPCN1, XBRVR1, IBRVR2, IUPDN2, XBRVR2, 80X20014
4 L10, NITER, NBINV, M7, IPHASE, NPHASE, NM3M7, IALGO, IECJ 80X20015
DIMENSION INT (NC1), ICC (NC1), IS (ND4), INUSE (ND10) 80X20016
DIMENSION BCRIG (NC4), RMS (ND4), C2 (ND1), U (ND6), PJ (NC4), XNOT (ND1), 80X20017
1 SIGNAL (ND6), SIGMAU (ND6), XZ (ND6), S0 (ND8), S1 (ND8) 80X20018
IF (ITRACE.GE.1) WRITE (6, 1007) 80X20019
C INITIALIZE THE BEST UPPER BOUND AND LIST. 80X20020
LPHASE=1 80X20021
NODNOT=0 80X20022
DO100J=1, N 80X20023
100 XNOT (J)=0.0 80X20024
NLIST=0 80X20025
NLISTS=0 80X20026
NFEAS=0 80X20027
LSTMX=0 80X20028
ITRTOT=0 80X20029
ITRMAX=0 80X20030
DO110I=1, MXLIST 80X20031
110 INUSE (I)=0 80X20032
NOORUL=NOORL1 80X20033
NBVRUL=NBVRL1 80X20034
NTIGHT=NTITE1 80X20035
PBRNOD=1.0 80X20036
NUPOWN=1 80X20037
NODE=1 80X20038
NBRNOD=0 80X20039
IF (IOUTPT.NE.C) WRITE (6, 1000) NODE, NBRNOD 80X20040
C ESTABLISH THE UPPER BOUNDS FOR THE FIRST SUBPROGRAM. 80X20041
DO120J=1, N1P2 80X20042
120 U (J)=SIGMAU (J)-SIGNAL (J) 80X20043
C ESTABLISH THE COST DATA FOR THE FIRST SUBPROGRAM. 80X20044
TSIG=0.0 80X20045
DO150J=1, N 80X20046
K=ICC (J) 80X20047
IF (K.EQ.0) GOTO140 80X20048
CALL GETOBJ (K, SIGNAL (J), F0) 80X20049
TSIG=TSIG + F0 80X20050
IF (ABS (U (J)).LE.TOL2) GOTO130 80X20051
CALL GETOBJ (K, SIGMAU (J), F1) 80X20052
C2 (J)=(F1-F0)/U (J) 80X20053
GOTO150 80X20054
130 C2 (J)=0.0 80X20055
GOTO150 80X20056
140 TSIG=TSIG + C2 (J)*SIGNAL (J) 80X20057

```

EXHIBIT 4 (Continued)

150 CONTINUE	BOX20058
IF (IOUTPT.LE.1)GOTO190	BOX20059
WRITE(6,1001)TSIG	BOX20060
WRITE(6,1002)	BOX20061
DO170J=1,N	BOX20062
K=INT(J)	BOX20063
L=ICC(J)	BOX20064
IF (IOUTPT.GE.3)GOTO160	BOX20065
IF (K.EQ.0 .AND. L.EQ.0)GOTO170	BOX20066
160 WRITE(6,1003)J,K,L,SIGMAL(J),SIGMAU(J),C2(J)	BOX20067
170 CONTINUE	BOX20068
IF (IOUTPT.LE.2)GOTO190	BOX20069
IF (NM1M3.EQ.N)GOTO190	BOX20070
DO180J=NP1,NM1M3	BOX20071
180 WRITE(6,1004)J,SIGMAL(J),SIGMAU(J)	BOX20072
190 CONTINUE	BOX20073
C ESTABLISH THE RIGHT-HAND-SIDE FOR THE FIRST SUBPROGRAM.	BOX20074
DO200I=1,MP1	BOX20075
200 RHS(I)=BORIG(I)	BOX20076
IPHASE=1	BOX20077
DO230J=1,N	BOX20078
IF (ABS(SIGMAL(J)).LE.EPSI)GOTO230	BOX20079
DO210I=1,MP1	BOX20080
210 PJ(I)=0.0	BOX20081
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,	BOX20082
1 J,NZEROS)	BOX20083
DO220I1=1,NZEROS	BOX20084
I=IS(I1)	BOX20085
220 RHS(I)=RHS(I) - PJ(I)*SIGMAL(J)	BOX20086
230 CONTINUE	BOX20087
C ESTABLISH THE INITIAL BASIS, BASIS INVERSE, AND VALUES OF THE BASIC	BOX20088
C VARIABLES FOR THE FIRST SUBPROGRAM.	BOX20089
CALL INPUT4 (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,BORIG,RHS,C2,C1,	BOX20090
1 BI,BN,U,PJ,BINV,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)	BOX20091
C DETERMINE THE APPLICABLE LP ALGORITHM.	BOX20092
CALL INPUT5 (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,L,PJ,B,	BOX20093
1 ND1,ND2,ND3,ND4,ND5,ND6)	BOX20094
IF (IALGO.NE.0)GOTO240	BOX20095
IEOJ=1	BOX20096
RETURN	BOX20097
C SOLVE THE FIRST SUBPROGRAM.	BOX20098
240 CALL SIMPLE (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,BN,	BOX20099
1 U,PJ,BINV,XJ,V,XZ,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)	BOX20100
IF (IEOJ.NE.0)RETURN	BOX20101
NFEAS=NFEAS + 1	BOX20102
C PRINT THE SOLUTION.	BOX20103
Z=Z+TSIG	BOX20104
DO250J=1,NIP2	BOX20105
250 XZ(J)=XZ(J) + SIGMAL(J)	BOX20106
IF (IOUTPT.EQ.0)GOTO260	BOX20107
NUP=N	BOX20108
IF (IOUTPT.GE.3)NUP=NM1M3	BOX20109
WRITE(6,1005)Z	BOX20110
WRITE(6,1006)(XZ(J),J=1,NUP)	BOX20111
260 IF (NDB.EQ.1)RETURN	BOX20112
IF (NBVRUL.LE.2 .OR. NTIGHT.EQ.0)GOTO280	BOX20113
C INITIALIZE THE SLOPES.	BOX20114

EXHIBIT 4 (Continued)

DO270I=1,M7	BOX20115
S0(I)=0.0	BOX20116
270 S1(I)=0.0	BOX20117
RETURN	BOX20118
C DETERMINE THE SLOPES ASSOCIATED WITH THE OPTIMAL OBJECTIVE VALUE.	BOX20119
280 CALL SLOPES (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,PJ,XJ,S0,	BOX20120
1 S1,8,ND1,ND2,ND3,ND4,ND5,ND8)	BOX20121
RETURN	BOX20122
1000 FORMAT(1H1,50(1H*)/6H0NODE ,I5/26H0BRANCHED FROM NODE ,I5)	BOX20123
1001 FORMAT(7HJTSIG =,E15.6)	BOX20124
1002 FORMAT(9HCVARIABLE,4X,3HINT,8X,2HCC,12X,5HLOWER,12X,5HUPPER,13X,	BOX20125
1 4HCOST/2X,6HNUMBER,3X,8HVARIABLE,2X,8HVARIABLE,9X,5HLIMIT,	BOX20126
2 12X,5HLIMIT,9X,11HCOEFFICIENT/12X,6HNUMBER,4X,6HNUMBER//)	BOX20127
1003 FORMAT(3X,I5,2(5X,I5),3X,3(E15.6,2X))	BOX20128
1004 FORMAT(3X,I5,23X,E15.6,2X,E15.6)	BOX20129
1005 FORMAT(17H0SOLUTION VALUE =,E15.6)	BOX20130
1006 FORMAT(17H0VARIABLES =,6E15.6/(17X,6E15.6))	BOX20131
1007 FORMAT(13H *****BOX2)	BOX20132
END	BOX20133

EXHIBIT 4 (Continued)

SUBROUTINE BOX5 (INLSE, XNOT, CAPP, CAPL, ND1, ND10, ND11, IENTRY)	BOX50001
C PRINT THE SOLUTION.	BOX50002
COMMON/P1/N, M, ITYPE, NSTRAT, NOORL1, NBVRL1, NTITE1, NOORL2, NBVRL2,	BOX50003
1 NTITE2, MXLIST, LISTOP, ITAPE, IF8, MXITER, MBINV, IOUTPT,	BOX50004
2 ITRACE, MSTART, TIME1, TOL1, TOL2, PCBUS, ALPHA (10)	BOX50005
COMMON/P3/NOENOT, UNOT, I8UBOP, LPHASE, NOORUL, NBVRUL, NTIGHT, NLIST,	BOX50006
1 NLISTS, NFEAS, LSTMX, ITRTCT, ITRMAX, BL2, NBRNOD, PBRNOD,	BOX50007
2 NBRVAR, NUPDOWN, XBRNOD, TBRNOD, NODE, LNODE, Z, BOUNDL, BCUNDU,	BOX50008
3 TSIG, IFEAS, I8VR1, IUPDN1, X8VR1, I8VR2, IUPDN2, X8VR2,	BOX50009
4 L10, NITER, NBINV, M7, IPHASE, NPHASE, NM3M7, IALGO, IE0J	BOX50010
DIMENSION INUSE (ND1C)	BOX50011
DIMENSION XNOT (ND1), CAPP (ND10), CAPL (ND11)	BOX50012
IF (ITRACE.GE.1) WRITE (6, 1017)	BOX50013
C PRINT OUT PROGRAM IDENTIFICATION.	BOX50014
WRITE (6, 1000) N, M, ITYPE, NSTRAT, NOORL1, NBVRL1, NTITE1, NOORL2,	BOX50015
1 NBVRL2, NTITE2, MXLIST, LISTOP, ITAPE, IF8, MXITER,	BOX50016
2 MBINV, IOUTPT, ITRACE, MSTART	BOX50017
WRITE (6, 1001) TIME1, TOL1, TOL2, UNOT, PCBUS	BOX50018
WRITE (6, 1002) ALPHA	BOX50019
C PRINT OUT THE SOLUTION OR THE CURRENT BEST SOLUTION.	BOX50020
IF (IENTRY.EQ.1) GOTO 100	BOX50021
WRITE (6, 1003) NOENOT	BOX50022
REWIND 9	BOX50023
REWIND 10	BOX50024
WRITE (9) NOENCT	BOX50025
WRITE (10) NOENOT	BOX50026
END FILE 9	BOX50027
END FILE 10	BOX50028
GOTO 110	BOX50029
100 WRITE (6, 1004) NOENOT	BOX50030
110 WRITE (6, 1005) UNCT	BOX50031
WRITE (6, 1006) (XNOT (I), I=1, N)	BOX50032
WRITE (6, 1007) NODE	BOX50033
WRITE (6, 1008) NFEAS	BOX50034
WRITE (6, 1009) LSTMX	BOX50035
WRITE (6, 1010) ITRTCT	BOX50036
WRITE (6, 1011) ITRMAX	BOX50037
IF (IENTRY.NE.1) GOTO 180	BOX50038
C PRINT OUT THE LIST.	BOX50039
WRITE (6, 1012) NLIST	BOX50040
INDEX=0	BOX50041
IF (NSTRAT.EQ.2) GOTO 120	BOX50042
IF (NOORL1.EQ.1) GOTO 140	BOX50043
GOTO 130	BOX50044
120 IF (NOORL2.EQ.1) GOTO 140	BOX50045
IF (LPHASE.EQ.1 .AND. NOORL1.EQ.1) GOTO 140	BOX50046
130 INDEX=1	BOX50047
WRITE (6, 1013)	BOX50048
GOTO 150	BOX50049
140 WRITE (6, 1014)	BOX50050
150 DO 170 I=1, MXLIST	BOX50051
IF (INUSE (I).EQ.0) GOTO 170	BOX50052
IF (INDEX.EQ.0) GOT (100	BOX50053
WRITE (6, 1015) INUSE (I), CAPP (I)	BOX50054
GOTO 170	BOX50055
160 WRITE (6, 1016) INUSE (I), CAPP (I), CAPL (I)	BOX50056
170 CONTINUE	BOX50057

EXHIBIT 4 (Continued)

100	CALL TIMEC	80X50058
	RETURN	80X50059
1000	FORMAT(21)INTEGER PARAMETERS =,1(I7/21X,3I7)	80X50060
1001	FORMAT(21)REAL PARAMETERS =,5E15.6)	80X50061
1002	FORMAT(21)PROGRAM IDENTIFIER =,4X,10A8)	80X50062
1003	FORMAT(66)THE SOLUTION TO THE INTEGER CONCAVE PROGRAM WAS PROVIDED	80X50063
	10 BY NODE ,I5)	80X50064
1004	FORMAT(79)THE CURRENT BEST SOLUTION TO THE INTEGER CONCAVE PROGRAM	80X50065
	1M WAS PROVIDED BY NODE ,I5)	80X50066
1005	FORMAT(17)SOLUTION VALUE =,E15.6)	80X50067
1006	FORMAT(17)VARIABLES =,6E15.6/(17X,6E15.6))	80X50068
1007	FORMAT(34)THE NUMBER OF NODES EXAMINED WAS ,I5)	80X50069
1008	FORMAT(32)THE NUMBER OF NODES SOLVED WAS ,2X,I5)	80X50070
1009	FORMAT(27)THE MAXIMUM LIST SIZE WAS ,I5)	80X50071
1010	FORMAT(54)THE TOTAL NUMBER OF SIMPLEX ITERATIONS PERFORMED WAS ,	80X50072
	1 I7)	80X50073
1011	FORMAT(51)THE MAXIMUM NUMBER PERFORMED ALONG ANY BRANCH WAS ,	80X50074
	1 3X,I7)	80X50075
1012	FORMAT(26)THE CURRENT LIST SIZE IS ,I5)	80X50076
1013	FORMAT(1)0,3X,4HNCDE,9X,5HLOWER/17X,5HBCUND//)	80X50077
1014	FORMAT(1)0,3X,4HNCDE,9X,5HLOWER,8X,10HPROCESSING/17X,5HBOUND,	80X50078
	1 11X,5HORCER//)	80X50079
1015	FORMAT(3X,I5,2X,E15.6)	80X50080
1016	FORMAT(3X,I5,2X,E15.6,2X,F11.0)	80X50081
1017	FORMAT(10) *****BCX5)	80X50082
	END	80X50083

EXHIBIT 4 (Continued)

SUBROUTINE BOX7 (INUSE,CAPP,ND10)	BOX70001
C DETERMINE THE BEST LOWER BOUND.	BOX70002
COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVR1,NTITE1,NODRL2,NBVR2,	BOX70003
1 NTITE2,MXLIST,LISTOP,ITAPE,IFE,MXITER,MEINV,IOUPT,	BOX70004
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUS,ALPHA(10)	BOX70005
COMMON/P2/EPSI,EP SIM,BIGN,BEGTH,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,	BOX70006
1 NM1M3,N1P2,NP1,NSUM,NTC,M10	BOX70007
COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,	BOX70008
1 NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,	BOX70009
2 NBRVAR,NUPDOWN,XBRNOD,TBRNOD,NOGE,LNODE,Z,BOUNDL,BCUNDU,	BOX70010
3 TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUFON2,XBRVR2,	BOX70011
4 L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ	BOX70012
DIMENSION INUSE(N010)	BOX70013
DIMENSION CAPP(ND10)	BOX70014
IF (ITRACE.GE.1) WRITE(6,1000)	BOX70015
BLB=BIGN	BOX70016
DO100I=1,MXLIST	BOX70017
IF (INUSE(I).EQ.0) GOTO100	BOX70018
IF (CAPP(I).GE.BLB) GOTO100	BOX70019
BLB=CAPP(I)	BOX70020
100 CONTINUE	BOX70021
RETURN	BOX70022
1000 FORMAT(10H *****BOX7)	BOX70023
END	BOX70024

EXHIBIT 4 (Continued)

SUBROUTINE BOX10 (INUSE,CAPP,CAPL,ND10,ND11)	BOX10001
C SELECT A NODE FROM THE LIST AND PUT IT IN THE SUBLIST.	BOX10002
COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVR1,NTITE1,NODRL2,NBVR2,	BOX10003
1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,	BOX10004
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)	BOX10005
COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,	BOX10006
1 NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,DLB,NBRNOD,PBRNOD,	BOX10007
2 NBRVAR,NUPDN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,	BOX10008
3 TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,	BOX10009
4 L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ	BOX10010
DIMENSION INUSE(ND10)	BOX10011
DIMENSION CAPP(ND10),CAPL(ND11)	BOX10012
IF(ITRACE.GE.1)WRITE(6,1000)	BOX10013
C SELECT THE NODE.	BOX10014
PBRNOD=J.0	BOX10015
DO100I=1,MXLIST	BOX10016
IF(INUSE(I).EQ.0)GOTO100	BOX10017
IF(CAPP(I).GE.UNOT)GOTO100	BOX10018
IF(CAPL(I).LE.PBRNOD)GOTO100	BOX10019
PBRNOD=CAPL(I)	BOX10020
IJ=I	BOX10021
100 CONTINUE	BOX10022
C PUT IT IN THE SUBLIST.	BOX10023
INUSE(I0)=-INUSE(I0)	BOX10024
NLISTS=NLISTS+1	BOX10025
RETURN	BOX10026
1000 FORMAT(11H *****BOX10)	BOX10027
END	BOX10028

EXHIBIT 4 (Continued)

```

SUBROUTINE BOX12 (INUSE,IMS,C2,CAPP,CAPL,SIGMAL,SIGMAU,SLOLD,      BOX10001
1          SUOLD,CZOLD,FMS,ND1,ND6,ND9,ND10,ND11,NOMS2,        BOX10002
2          NOMS3)                                             BOX10003
C SELECT THE BRANCHING NODE FROM THE SUBLIST (PHASE 1) OR THE LIST
C (PHASE 2).
COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVR1,NTITE1,NODRL2,NBVR2,    BOX10004
1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,HBINV,IOUTPT,   BOX10005
2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)        BOX10006
COMMON/P2/EPSI,EPSIM,BIGN,BEGTH,M1,M2,M3,M4,N1,MP1,MP2,NM1M2,   BOX10009
1          NM1M3,N1P2,NP1,NSUM,NTC,M10                          BOX10010
COMMON/P3/NODNOT,LNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST, BOX10011
1          NLISTS,MFEAS,LSTMX,ITRTCT,ITRMAX,ELB,NBRNOD,PBRNOD,   BOX10012
2          NERVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BCUNDU, BOX10013
3          TSIG,IFEAS,IBRVR1,IUPCN1,XBRVR1,IBRVR2,IUPON2,XBRVR2, BOX10014
4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ     BOX10015
DIMENSION INUSE(ND10),IMS(NOMS2)                               BOX10016
DIMENSION C2(ND1),CAPP(ND10),CAPL(ND11),SIGMAL(ND6),SIGMAU(ND6), BOX10017
1          SLOLD(ND6),SUOLD(ND6),CZOLD(ND9),FMS(NOMS3)          BOX10018
IF (ITRACE.GE.1) WRITE(6,1000)                                BOX10019
IF (IBUBOP.EQ.0) GOTO110                                       BOX10020
C*****
C SELECT THE BRANCHING NODE FROM THE SUBLIST.                   BOX10022
C*****
PBRNOD=0.0                                                       BOX10024
DO100I=1,MXLIST                                                 BOX10025
IF (INUSE(I).GE.0) GOTO100                                       BOX10026
IF (CAPL(I).LE.PBRNOD) GOTO100                                    BOX10027
PBRNOD=CAPL(I)                                                  BOX10028
NBRNOD=-INUSE(I)                                               BOX10029
I0=I                                                            BOX10030
100 CONTINUE                                                    BOX10031
C DECREMENT THE SUBLIST COUNTER.                                BOX10032
NLISTS=NLISTS-1                                               BOX10033
GOTO150                                                         BOX10034
C*****
C SELECT THE BRANCHING NODE FROM THE LIST.                       BOX10036
C*****
110 IF (NODRUL.EQ.1) GOTO130                                     BOX10038
C PRIORITY NODE SELECTION RULE.                                  BOX10039
BLB=BIGN                                                       BOX10040
DO120I=1,MXLIST                                                 BOX10041
IF (INUSE(I).EQ.0) GOTO120                                       BOX10042
IF (CAPP(I).GE.BLB) GOTO120                                       BOX10043
BLB=CAPP(I)                                                    BOX10044
NBRNOD=INUSE(I)                                               BOX10045
I0=I                                                            BOX10046
120 CONTINUE                                                    BOX10047
GOTO150                                                         BOX10048
C LIFO NODE SELECTION RULE.                                     BOX10049
130 PBRNOD=0.0                                                 BOX10050
DO140I=1,MXLIST                                                 BOX10051
IF (INUSE(I).EQ.0) GOTO140                                       BOX10052
IF (CAPL(I).LE.PBRNOD) GOTO140                                       BOX10053
PBRNOD=CAPL(I)                                               BOX10054
NBRNOD=INUSE(I)                                               BOX10055
I0=I                                                            BOX10056
140 CONTINUE                                                    BOX10057

```

EXHIBIT 4 (Continued)

C DECREMENT THE LIST COUNTER.	80X10058
150 INUSE(I0)=0	80X10059
NLIST=NLIST-1	80X10060
C*****	80X10061
C READ IN THE DATA FOR THE BRANCHING NOISE.	80X10062
C*****	80X10063
IF (NODE.EQ.NBRNOD)GOTO200	80X10064
CALL READMS (2,IMS,NDMS2,I0)	80X10065
CALL READMS (3,FMS,NDMS3,I0)	80X10066
IBRVR1=IMS(1)	80X10067
IUPDN1=IMS(2)	80X10068
L10=IMS(3)	80X10069
NITER=IMS(4)	80X10070
NBINV=IMS(5)	80X10071
M7=IMS(6)	80X10072
IPHASE=IMS(7)	80X10073
NPHASE=IMS(8)	80X10074
NM3M7=IMS(9)	80X10075
IF (INSTRAT.EQ.1)GOTO160	80X10076
IBRVR2=IMS(10)	80X10077
IUPDN2=IMS(11)	80X10078
160 Z=FMS(1)	80X10079
TSIG=FMS(2)	80X10080
XBRVR1=FMS(3)	80X10081
IF (INSTRAT.EQ.1)GOTO170	80X10082
XBRVR2=FMS(4)	80X10083
170 DO180J=1,N1P2	80X10084
SIGMA(J)=SLOLD(J)	80X10085
180 SIGMAU(J)=SUOLD(J)	80X10086
IF (ITYPE.EQ.1)GOTO200	80X10087
DO190J=1,N	80X10088
190 C2(J)=C2OLD(J)	80X10089
200 TBRNOD=TSIG	80X10090
C ADJUST THE BRANCHING VARIABLE SELECTION ACCORDING TO THE CURRENT	80X10091
C PHASE OF THE ALGORITHM.	80X10092
IF (INSTRAT.EQ.1)GOTO210	80X10093
IF (LPHASE.EQ.1)GOTO210	80X10094
NBRVAR=IBRVR2	80X10095
NUPDOWN=IUPDN2	80X10096
XBRNOD=XBRVR2	80X10097
RETURN	80X10098
210 NBRVAR=IBRVR1	80X10099
NUPDOWN=IUPDN1	80X10100
XBRNOD=XBRVR1	80X10101
RETURN	80X10102
1000 FORMAT(11H *****80X12)	80X10103
END	80X10104

EXHIBIT 4 (Continued)

```

SUBROUTINE BOX13 (INT,ICC,IEV,C2,SIGMAL,SIGMAU,XZ,S0,S1,SLOLD,      BOX10001
1 SUOLD,NC1,ND4,N(6,ND8)                                         BOX10002
C TIGHTEN THE LIMITS ON THE MONOTONE VARIABLES (PHASE 1) OR TIGHTEN      BOX10003
C THE LIMITS ON ALL VARIABLES USING THE BEST UPPER BOUND (PHASE 2).    BOX10004
COMMON/P1/N,M,I,TYPE,NSTRAT,NODRL1,NBVR1,NTITE1,NODRL2,NBVR2,      BOX10005
1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,FXITER,MBINV,IOUTPT,           BOX10006
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBL8,ALPHA(10)                 BOX10007
COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,  BOX10008
1 NM1M3,N1P2,NP1,NSUM,NTC,M10                                    BOX10009
COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,   BOX10010
1 NLISTS,NFEAS,LSTMX,IIRT(IT,ITRMAX,BLE,NBRNOD,PBRNOD,          BOX10011
2 NBRVAR,NUPDOWN,XERNOC,TBRNOD,NODE,LNODE,Z,BOUNDL,BCUNDU,      BOX10012
3 TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,        BOX10013
4 LIC,NITER,NBINV,M7,I,PHASE,NPHASE,NM3:I7,IALGO,IEOJ           BOX10014
DIMENSION INT(ND1),ICC(ND1),I3V(ND4)                             BOX10015
DIMENSION C2(ND1),SIGMAL(ND6),SIGMAU(ND6),XZ(ND6),S0(ND8),S1(ND8),BOX10016
1 SLOLD(ND6),SUOLE(ND6)                                          BOX10017
IF (ITRACE.EQ.1)WRITE(6,1005)                                     BOX10018
IF (NTIGHT.EQ.1)RETURN                                           BOX10019
NCOUNT=0                                                         BOX10020
IF (UNOT.EQ.8IGN)GOTO310                                          BOX10021
IF (IBUBOP.EQ.1)GOTO310                                          BOX10022
C*****                                                           BOX10023
C TIGHTEN THE LIMITS ON THE BASIC VARIABLES.                       BOX10024
C*****                                                           BOX10025
DO300I=1,M7                                                       BOX10026
IF (I.EQ.MP1)GOTO300                                             BOX10027
J=I+V(I)                                                          BOX10028
IF (ABS(SIGMAU(J)-SIGMAL(J)).LE.TOL2)GOTO300                    BOX10029
XZJJJ=XZ(J)                                                       BOX10030
IF (XZJJJ.GT.SIGMAL(J)+TOL2)GOTO100                              BOX10031
S0(I)=-BIGN                                                       BOX10032
XZJJJ=SIGMAL(J)                                                  BOX10033
GOTO110                                                            BOX10034
100 IF (XZJJJ.LT.SIGMAU(J)-TOL2)GOTO110                          BOX10035
S1(I)=BIGN                                                        BOX10036
XZJJJ=SIGMAU(J)                                                  BOX10037
110 CONTINUE                                                       BOX10038
C ALL VARIABLE TYPES.                                             BOX10039
V1=SIGMAL(J)                                                       BOX10040
IF (S0(I).EQ.-BIGN)GOTO120                                        BOX10041
T1=Z + S0(I)*(V1 - XZJJJ)                                         BOX10042
GOTO130                                                            BOX10043
120 T1=BIGN                                                       BOX10044
130 IF (J.GT.N)GOTO230                                            BOX10045
IF (INT(J).EQ.0)GOTO230                                          BOX10046
C INTEGER LINEAR OR INTEGER CONCAVE VARIABLE.                    BOX10047
K=XZJJJ                                                            BOX10048
V2=K                                                               BOX10049
V3=K+1                                                            BOX10050
K=ICC(J)                                                           BOX10051
IF (K.EQ.0)GOTO150                                               BOX10052
CALL GETOBJ (K,V1,F1)                                             BOX10053
CALL GETOBJ (K,V2,F2)                                             BOX10054
CALL GETOBJ (K,V3,F3)                                             BOX10055
IF (S0(I).EQ.-BIGN .AND. ABS(XZJJJ-V2).GT.TOL2)GOTO170         BOX10056
IF (S0(I).EQ.-BIGN)GOTO140                                       BOX10057

```

EXHIBIT 4 (Continued)

	T2=Z + S0(I)*(V2 - XZJJJ) + F2 - (F1 + C2(J)*(V2-SIGMAL(J)))	80X10058
	GOTO180	80X10059
140	T2=Z + F2 - (F1 + C2(J)*(V2-SIGMAL(J)))	80X10060
	GOTO180	80X10061
150	IF(S0(I).EQ.-BIGN .AND. ABS(XZJJJ-V2).GT.TOL2)GOTO170	80X10062
	IF(S0(I).EQ.-BIGN)GOTO160	80X10063
	T2=Z + S0(I)*(V2 - XZJJJ)	80X10064
	GOTO180	80X10065
160	T2=Z	80X10066
	GOTO180	80X10067
170	T2=BIGN	80X10068
180	IF(S1(I).EQ.BIGN .AND. ABS(V3-XZJJJ).GT.TOL2)GOTO220	80X10069
	IF(K.EQ.0)GOTO200	80X10070
	IF(S1(I).EQ.BIGN)GOTO190	80X10071
	T3=Z + S1(I)*(V3 - XZJJJ) + F3 - (F1 + C2(J)*(V3-SIGMAL(J)))	80X10072
	GOTO260	80X10073
190	T3=Z + F3 - (F1 + C2(J)*(V3-SIGMAL(J)))	80X10074
	GOTO260	80X10075
200	IF(S1(I).EQ.BIGN)GOTO210	80X10076
	T3=Z + S1(I)*(V3 - XZJJJ)	80X10077
	GOTO260	80X10078
210	T3=Z	80X10079
	GOTO260	80X10080
220	T3=BIGN	80X10081
	GOTO260	80X10082
C	LINEAR OR CONCAVE VARIABLE.	80X10083
230	V2=XZJJJ	80X10084
	V3=V2	80X10085
	IF(J.GT.N)GOTO240	80X10086
	K=ICC(J)	80X10087
	IF(K.EQ.0)GOTO240	80X10088
	CALL GETOBJ (K,V1,F1)	80X10089
	CALL GETOBJ (K,V2,F2)	80X10090
	T2=Z + F2 - (F1 + C2(J)*(V2-SIGMAL(J)))	80X10091
	GOTO250	80X10092
240	T2=Z	80X10093
250	T3=T2	80X10094
C	ALL VARIABLE TYPES.	80X10095
260	V4=SIGMAU(J)	80X10096
	IF(S1(I).EQ.BIGN)GOTO270	80X10097
	T4=Z + S1(I)*(V4 - XZJJJ)	80X10098
	GOTO280	80X10099
270	T4=BIGN	80X10100
280	CALL ADJUST (V1,V2,V3,V4,T1,T2,T3,T4,SL,SU)	80X10101
	SIGMAL(J)=SL	80X10102
	SIGMAU(J)=SU	80X10103
	IF(J.GT.N)GOTO290	80X10104
	IF(INT(J).EQ.0)GOTO290	80X10105
C	FOR INTEGER VARIABLES, CHECK THAT THE LOWER AND UPPER LIMITS ARE	80X10106
C	INTEGERS.	80X10107
	K=SL	80X10108
	SIGMAL(J)=K	80X10109
	IF(ABS(SL - SIGMAL(J)).GT.TOL2)SIGMAL(J)=K+1	80X10110
	K=SU	80X10111
	SIGMAU(J)=K+1	80X10112
	IF(ABS(SU - SIGMAU(J)).GT.TOL2)SIGMAU(J)=K	80X10113
C	INCREMENT THE COUNTER IF THE LIMITS HAVE CHANGED.	80X10114

EXHIBIT 4 (Continued)

```

290 IF (ABS(SIGMA(J)-SLOLD(J)).GT.TOL2) NCOUNT=NCOUNT+1      BOX10115
    IF (ABS(SIGMAU(J)-SUOLD(J)).GT.TOL2) NCOUNT=NCOUNT+1      BOX10116
300 CONTINUE                                                    BOX10117
    GOTO370                                                       BOX10118
C*****
C ADJUST THE LIMITS ON THE MONOTONE VARIABLES.                  BOX10119
C*****
310 DO360 I=1,M7
    IF (I.EQ.MP1) GOTO360                                         BOX10122
    J=IBV(I)                                                       BOX10123
    IF (ABS(SIGMAU(J)-SIGMA(J)).LE.TOL2) GOTO360                 BOX10124
    XZJJJ=XZ(J)                                                   BOX10125
    IF (XZJJJ.LE.SIGMA(J)+TOL2) GOTO330                           BOX10127
    IF (S0(I).NE.-SIGN) GOTO330                                    BOX10128
C ADJUST THE LOWER LIMIT.                                        BOX10129
    IF (J.GT.N) GOTO320                                           BOX10130
    IF (INT(J).EQ.C) GOTO320                                       BOX10131
    K=XZJJJ                                                         BOX10132
    SIGMA(J)=K                                                     BOX10133
    IF (ABS(XZJJJ-SIGMA(J)).GT.TOL2) SIGMA(J)=K+1                BOX10134
    GOTO330                                                         BOX10135
320 SIGMA(J)=XZJJJ                                               BOX10136
330 IF (XZJJJ.GE.SIGMAU(J)-TOL2) GOTO350                          BOX10137
    IF (S1(I).NE.BIGN) GOTO350                                     BOX10138
C ADJUST THE UPPER LIMIT.                                       BOX10139
    IF (J.GT.N) GOTO340                                           BOX10140
    IF (INT(J).EQ.C) GOTO340                                       BOX10141
    K=XZJJJ                                                         BOX10142
    SIGMAU(J)=K+1                                                  BOX10143
    IF (ABS(XZJJJ-SIGMAU(J)).GT.TOL2) SIGMAU(J)=K                BOX10144
    GOTO350                                                         BOX10145
340 SIGMAU(J)=XZJJJ                                               BOX10146
C INCREMENT THE COUNTER IF THE LIMITS HAVE CHANGED.            BOX10147
350 IF (ABS(SIGMA(J)-SLOLD(J)).GT.TOL2) NCOUNT=NCOUNT+1      BOX10148
    IF (ABS(SIGMAU(J)-SUOLD(J)).GT.TOL2) NCOUNT=NCOUNT+1      BOX10149
360 CONTINUE                                                    BOX10150
C*****
C PRINT THE OLD AND NEW LIMITS.                                  BOX10152
C*****
370 IF (IOUTPT.LE.2) RETURN                                       BOX10154
    IF (UNOT.EQ.BIGN) GOTO380                                       BOX10155
    IF (IBUBOP.EQ.1) GOTO380                                       BOX10156
    WRITE(6,1000) NBRNUD                                           BOX10157
    GOTO390                                                         BOX10158
380 WRITE(6,1001) NBRNCD                                           BOX10159
390 IF (NCOUNT.EQ.C) GOTO410                                       BOX10160
    WRITE(6,1002) NCOU T                                           BOX10161
    DO400 I=1,M7                                                   BOX10162
    IF (I.EQ.MP1) GOTO400                                           BOX10163
    J=IBV(I)                                                       BOX10164
    IF (ABS(SIGMA(J)-SLOLD(J)).LE.TOL2 .AND.                      BOX10165
        ABS(SIGMAU(J)-SUOLD(J)).LE.TOL2) GOTO400                 BOX10166
    WRITE(6,1003) J, SLOLD(J), SIGMA(J), XZ(J), SIGMAU(J), SUOLD(J) BOX10167
400 CONTINUE                                                    BOX10168
    GOTO420                                                         BOX10169
410 WRITE(6,1004)                                                 BOX10170
420 CALL TIMEC                                                    BOX10171

```

EXHIBIT 4 (Continued)

RETURN	BOX10172
1000 FORMAT(1H0,50(1H*)/52H0TIGHTEN THE LIMITS ON THE BASIC VARIABLES FOR NODE ,I5)	BOX10173
1001 FORMAT(1H0,50(1H*)/54H0ADJUST THE LIMITS ON THE MONOTONE VARIABLES FOR NODE ,I5)	BOX10174
1002 FORMAT(12H0THERE WERE ,I5,23H CHANGES TO THE LIMITS./	BOX10175
1 1H0,2X,5HBASIC,11X,3HOLD,14X,3HNEW,11X,8HVARIABLE,12X,	BOX10176
2 3HNEW,14X,3HOLD/1X,8HVARIABLE,2(9X,5HLOWER,3X),9X,	BOX10177
3 5HVALUE,2(12X,5HUPPER)/6X,2(12X,5HLIMIT),17X,	BOX10178
4 2(12X,5HLIMIT)//)	BOX10179
1003 FORMAT(3X,I5,1X,5(2X,E15.6))	BOX10180
1004 FORMAT(37H0THERE WERE NO CHANGES TO THE LIMITS.)	BOX10181
1005 FORMAT(11H *****BOX13)	BOX10182
END	BOX10184
	BOX10185

EXHIBIT 4 (Continued)

```

SUBROUTINE BOX15 (NZ, NP, IR, IA, INT, ICC, IS, NV, IBV, NBV, IUPPER, TC,      BOX10001
1      BORIG, RHS, C2, C1, BI, BN, U, PJ, BINV, XJ, SIGMAL,                BOX10002
2      SIGMAU, V, XZ, S0, S1, SLOLD, C2OLD, B, ND1, ND2, ND3,            BOX10003
3      ND4, ND5, ND6, ND7, ND8, ND9)                                     BOX10004
C-SOLVE SUBPROGRAM LNODE.                                             BOX10005
COMMON/P1/N, M, ITYPE, NSTRAT, NCORL1, NBVRL1, NTITE1, NCORL2, NBVRL2,   BOX10006
1      NTITE2, MXLIST, LISTOP, ITAPE, IFB, MXITER, MBINV, IOUTPT,        BOX10007
2      ITRACE, MSTART, TIME1, TOL1, TOL2, PCBUB, ALPHA(10)             BOX10008
COMMON/P2/EPSI, EPSIM, BIGN, BEGTM, M1, M2, M3, M4, N1, MP1, MP2, NM1, NM2, BOX10009
1      NM1M3, N1P2, NP1, NSUM, NTC, M10                                 BOX10010
COMMON/P3/NODN0T, UNOT, IBUEOF, LPHASE, NODRUL, NBVRUL, NTIGHT, NLIST,   BOX10011
1      NLISTS, NFEAS, LSTMX, ITRTCT, ITRMAX, BLB, NBRNOD, PBRNOD,       BOX10012
2      NBRVAR, NUPDWN, XBRNOD, TBRNOD, NODE, LNODE, Z, BOUNDL, BCUNDO,   BOX10013
3      TSIG, IFEAS, IBRVR1, IUPDN1, XBRVR1, IBRVR2, IUPDN2, XBRVR2,    BOX10014
4      L10, NITER, NBINV, M7, IPHASE, NPHASE, NM3M7, IALGO, IE0J        BOX10015
COMMON/P4/SAVE, KBRAN, X1                                             BOX10016
DIMENSION INT(ND1), ICC(ND1), IS(ND4), IBV(ND4), NBV(ND5), IUPPER(ND5)   BOX10017
DIMENSION BORIG(ND4), RHS(ND4), C2(ND1), BI(ND4), BN(ND5), U(ND6),     BOX10018
1      PJ(ND4), SIGMAL(ND6), SIGMAU(ND6), XZ(ND6), SLOLD(ND6),          BOX10019
2      C2OLD(ND9), B(ND4, ND4)                                         BOX10020
IF (ITRACE.GE.1) WRITE(6, 1015)                                       BOX10021
NODE=NODE + 1                                                         BOX10022
IF (IOUTPT.NE.0) WRITE(6, 1000) NODE, NBRNOD                          BOX10023
IF (LNODE.EQ.2) GOTO290                                               BOX10024
C*****                                                                BOX10025
C INITIALIZE THE DATA REQUIRED FOR SUBPROGRAM 1.                      BOX10026
C*****                                                                BOX10027
IF (XBRNOD.LT.SIGMAL(NBRVAR)) XBRNOC=SIGMAL(NBRVAR)                   BOX10028
IF (XBRNOD.GT.SIGMAU(NBRVAR)) XBRNOC=SIGMAU(NBRVAR)                   BOX10029
C MODIFY THE UPPER LIMIT FOR THE BRANCHING VARIABLE.                 BOX10030
SAVE=SIGMAU(NBRVAR)                                                  BOX10031
IF (INT(NBRVAR).NE.C) GOTO110                                         BOX10032
C BRANCHING ON A CONCAVE VARIABLE.                                    BOX10033
100 IF (IOUTPT.NE.0) WRITE(6, 1001) NBRVAR, ICC(NBRVAR)              BOX10034
SIGMAU(NBRVAR)=XBRNOD                                               BOX10035
KBRAN=0                                                              BOX10036
GOTO120                                                                BOX10037
C BRANCHING ON AN INTEGER VARIABLE.                                   BOX10038
110 K=XBRNOD                                                         BOX10039
XC=K                                                                  BOX10040
X1=K+1                                                                BOX10041
IF (ABS(X0-XBRNOD).LE.TOL2 .AND. ICC(NBRVAR).NE.0) GOTO100          BOX10042
IF (ABS(X1-XBRNOD).LE.TOL2 .AND. ICC(NBRVAR).NE.0) GOTO100          BOX10043
IF (IOUTPT.NE.0) WRITE(6, 1002) NBRVAR, INT(NBRVAR)                 BOX10044
SIGMAU(NBRVAR)=XC                                                  BOX10045
KBRAN=1                                                              BOX10046
C ESTABLISH THE UPPER LIMITS FOR SUBPROGRAM 1.                      BOX10047
120 DO130 J=1, N1P2                                                  BOX10048
130 U(J)=SIGMAU(J) - SIGMAL(J)                                       BOX10049
C ESTABLISH THE COST DATA FOR SUBPROGRAM 1.                        BOX10050
DO140 J=1, N                                                         BOX10051
140 TBRNOD=TBRNOD + C2(J)*(SIGMAL(J) - SLOLD(J))                   BOX10052
TSIG=TBRNOD                                                         BOX10053
C ESTABLISH THE RIGHT-HAND-SIDE FOR SLSPROGRAM 1.                  BOX10054
DO150 I=1, M7                                                       BOX10055
150 RHS(I)=B0FIG(I)                                                 BOX10056
DO160 J=1, N1P2                                                     BOX10057

```

EXHIBIT 4 (Continued)

```

IF (ABS(SIGNAL(J)).LE.EPSI)GOTO180
00160I=1,M7
160 PJ(I)=0.0
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,NC2,ND3,ND4,ND5,
1 J,NZERCS)
00170I1=1,NZEROS
I=IS(I1)
170 RHS(I)=RHS(I) - PJ(I)*SIGNAL(J)
IF (IPHASE.EQ.2)RHS(MP2)=RHS(MP2) - PJ(MP2)*SIGNAL(J)
180 CONTINUE
C ESTABLISH THE BASIS INVERSE, THE VALUES OF THE BASIC VARIABLES, AND
C THE VALUES OF THE NONBASIC VARIABLES FOR SUBPROGRAM 1.
00190K=1,L10
IF (IUPPER(K).EQ.0)GOTO190
J=NBV(K)
BN(K)=U(J)
190 CONTINUE
IF (LISTOP.EQ.1)GOTO270
00200I=1,M7
200 BI(I)=RHS(I)
00230K=1,L10
IF (IUPPER(K).EQ.0)GOTO230
J=NBV(K)
IF (ABS(U(J)).LE.EPSI)GOTO230
00210I=1,M7
210 PJ(I)=0.0
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
1 J,NZEROS)
00220I1=1,NZEROS
I=IS(I1)
220 BI(I)=BI(I) - PJ(I)*U(J)
BI(MP1)=BI(MP1) - PJ(MP1)*U(J)
IF (IPHASE.EQ.2)BI(MP2)=BI(MP2) - PJ(MP2)*U(J)
230 CONTINUE
00250I=1,M7
Q1=0.0
00240J=1,M7
240 Q1=Q1 + B(I,J)*BI(J)
250 PJ(I)=Q1
00260I=1,M7
260 BI(I)=PJ(I)
GOTO280
270 CALL BINVRT (NZ,NF,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,U,PJ,
1 BINV,8,ND1,ND2,ND3,N(4,ND5,ND6,ND7))
280 REWIND 4
WRITE(4) (IBV(I),I=1,ND4), (NBV(I),I=1,ND5), (IUPPER(I),I=1,ND5),
1 (BI(I),I=1,ND4), (BN(I),I=1,ND5),L10,NITER,NBINV,
2 ((B(I,J),I=1,ND4),J=1,ND4)
C CHECK IF THE SUBPROGRAM IS INCLUDED IN THE NEXT SUBPROGRAM.
IF (INT(NBRVAR).NE.0)GOTO370
IF (ABS(U(NBRVAR)).GT.TOL2)GOTO370
IF (ABS(SAVE - SIGNAL(NBRVAR)).LE.TOL2)GOTO370
IEOJ=1
IF (IOUTPT.NE.0)WRITE(6,1004)
RETURN
C*****
C INITIALIZE THE DATA REQUIRED FOR SUBPROGRAM 2.

```

EXHIBIT 4 (Continued)

```

C*****80X10115
290 SIGMAU(NBRVAR)=SAVE
    SAVE=SIGNAL(NBRVAR)
    IF (KBRAN.EQ.1)GOTO300
    IF (IOUPT.NE.0)WRITE(6,1001)NBRVAR,ICC(NBRVAR)
    SIGNAL(NBRVAR)=XBRNOD
    GOTO310
300 IF (IOUPT.NE.0)WRITE(6,1002)NBRVAR,INT(NBRVAR)
    SIGNAL(NBRVAR)=X1
310 DELTA=SIGNAL(NBRVAR) - SAVE
C ADJUST THE UPPER LIMITS.
    U(NBRVAR)=SIGMAU(NBRVAR) - SIGNAL(NBRVAR)
C ADJUST THE COST DATA.
    IF (ITYPE.EQ.1)GOTO330
    DO320J=1,N
320 C2(J)=C2OLD(J)
330 TSIG=TBRNCD + C2(NBRVAR)*DELTA
C ADJUST THE RIGHT-HAND-SIDE.
    DO340I=1,M7
340 PJ(I)=0.C
    CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,NC2,ND3,ND4,ND5,
    1 NBRVAR,NZEROS)
    DO350I1=1,NZEROS
    I=IS(I1)
350 RHS(I)=RHS(I) - PJ(I)*DELTA
    IF (IPHASE.EQ.2)RHS(MP2)=RHS(MP2) - PJ(MP2)*DELTA
C ADJUST THE VALUES OF THE BASIC VARIABLES.
    REWIND 4
    READ (4)(IBV(I),I=1,ND4),(NBV(I),I=1,ND5),(IUPPER(I),I=1,ND5),
    1 (BI(I),I=1,ND4),(BN(I),I=1,ND5),L10,NITER,NJINV,
    2 ((B(I,J),I=1,ND4),J=1,ND4)
    DO360I=1,M7
    BI(I)=BI(I) + B(I,MP1)*PJ(MP1)*DELTA
    IF (NBRVAR.NE.IBV(I))GOTO360
    BI(I)=BI(I) - DELTA
360 CONTINUE
C CHECK IF THE SUBPROGRAM IS INCLUDED IN THE LAST SUBPROGRAM.
    IF (INT(NBRVAR).NE.0)GOTO370
    IF (ABS(U(NBRVAR)).GT.TOL2)GOTO370
    IEQJ=1
    IF (IOUPT.NE.0)WRITE(6,1005)
    RETURN
C*****80X10157
C SOLVE SUBPROGRAM LNQE.
C*****80X10158
C*****80X10159
370 IEQJ=0
    DO390J=1,N1P2
    IF (U(J).LT.-TOL2)GOTO380
    IF (U(J).GE.TOL2*0.1)GOTO390
    U(J)=TOL2*0.1
    GOTO390
380 IEQJ=1
390 CONTINUE
    IF (IEQJ.EQ.0)GOTO410
C THE LOWER AND UPPER LIMITS ARE INCOMPATIBLE.
    IF (IOUPT.NE.0)WRITE(6,1006)
    IF (IOUPT.LE.1)RETURN
    80X10171

```

EXHIBIT 4 (Continued)

WRITE(6,1007)	80X10172
DO400J=1,N1P2	80X10173
IF(U(J).GE.-TOL2)GOTO400	80X10174
WRITE(6,1008)J,SIGMAL(J),SIGMAU(J)	80X10175
400 CONTINUE	80X10176
RETURN	80X10177
C THE LOWER AND UPPER LIMITS ARE COMPATIBLE.	80X10178
410 IF(ITYPE.EQ.2)GOTO460	80X10179
C APPLY THE DUAL SIMPLEX ALGORITHM FIRST.	80X10180
IF(IOUTPT.LE.1)GOTO450	80X10181
WRITE(6,1003)TSIG	80X10182
WRITE(6,1009)	80X10183
DO430J=1,N	80X10184
K=INT(J)	80X10185
L=ICC(J)	80X10186
IF(IOUTPT.GE.3)GOTO420	80X10187
IF(K.EQ.0 .AND. L.EQ.0)GOTO430	80X10188
420 WRITE(6,1010)J,K,L,SIGMAL(J),SIGMAU(J),C2(J)	80X10189
430 CONTINUE	80X10190
IF(IOUTPT.LE.2)GOTO450	80X10191
IF(NM1M3.EQ.N)GOTO450	80X10192
DO440J=NP1,NM1M3	80X10193
440 WRITE(6,1011)J,SIGMAL(J),SIGMAU(J)	80X10194
450 IALGO=2	80X10195
CALL SIMPLE 'IZ,NP,IR,IA,IS,NV,IBV,NBV,IUFPER,TC,FHS,C2,C1,BI,BN,	80X10196
1 U,PJ,BINV,XJ,V,XZ,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)	80X10197
IF(IEQJ.NE.0)RETURN	80X10198
460 IF(ITYPE.EQ.1)GOTO580	80X10199
C APPLY THE PRIMAL ALGORITHM SECOND.	80X10200
C ESTABLISH NEW COST DATA.	80X10201
TSIG=0.0	80X10202
DO490J=1,N	80X10203
K=ICC(J)	80X10204
IF(K.EQ.0)GOTO480	80X10205
CALL GETOBJ(K,SIGMAL(J),F0)	80X10206
TSIG=TSIG + F0	80X10207
IF(ABS(U(J)).LE.TOL2)GOTO470	80X10208
CALL GETOBJ(K,SIGMAU(J),F1)	80X10209
C2(J)=(F1 - F0)/U(J)	80X10210
GOTO490	80X10211
470 C2(J)=0.0	80X10212
GOTO490	80X10213
480 TSIG=TSIG + C2(J)*SIGMAL(J)	80X10214
490 CONTINUE	80X10215
IF(IOUTPT.LE.1)GOTO530	80X10216
WRITE(6,1003)TSIG	80X10217
WRITE(6,1009)	80X10218
DO510J=1,N	80X10219
K=INT(J)	80X10220
L=ICC(J)	80X10221
IF(IOUTPT.GE.3)GOTO500	80X10222
IF(K.EQ.0 .AND. L.EQ.0)GOTO510	80X10223
500 WRITE(6,1010)J,K,L,SIGMAL(J),SIGMAU(J),C2(J)	80X10224
510 CONTINUE	80X10225
IF(IOUTPT.LE.2)GOTO530	80X10226
IF(NM1M3.EQ.N)GOTO530	80X10227
DO520J=NP1,NM1M3	80X10228

EXHIBIT 4 (Continued)

520	WRITE(6,1011)J,SIGMAL(J),SIGMAU(J)	80X10229
530	INDEX=0	80X10230
C	BASIC VARIABLES.	80X10231
	00550 I=1,M7	80X10232
	IF (I.EQ.MP1) GOT0550	80X10233
	J=IBV(I)	80X10234
	IF (J.GT.N) GOT0550	80X10235
	T=C2(J) - C2OLD(J)	80X10236
	IF (ABS(T).LE.EPSI) GOT0550	80X10237
	INDEX=1	80X10238
	BI(MP1)=BI(MP1) - T*BI(I)	80X10239
	00540 J=1,M7	80X10240
540	B(MP1,J)=B(MP1,J) - T*B(I,J)	80X10241
550	CONTINUE	80X10242
C	NONBASIC VARIABLES.	80X10243
	00560 I=1,L10	80X10244
	J=NBV(I)	80X10245
	IF (J.GT.N) GOT0560	80X10246
	T=C2(J) - C2OLD(J)	80X10247
	IF (ABS(T).LE.EPSI) GOT0560	80X10248
	INDEX=1	80X10249
560	CONTINUE	80X10250
	IF (INDEX.EQ.0) GOT0570	80X10251
	IALGO=1	80X10252
	CALL SIMPLE (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,BN,	80X10253
	1 U,PJ,BINV,XJ,V,XZ,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)	80X10254
	IF (IEOJ.NE.0) RETURN	80X10255
	GOT0580	80X10256
570	IF (IOUTPT.GE.2) WRITE(6,1012)	80X10257
580	NFEAS=NFEAS + 1	80X10258
C	PRINT THE SOLUTION.	80X10259
	Z=Z + TSIG	80X10260
	00590 J=1,N1P2	80X10261
590	XZ(J)=XZ(J) + SIGMAL(J)	80X10262
	IF (IOUTPT.EQ.0) GOT0600	80X10263
	NUP=N	80X10264
	IF (IOUTPT.GE.3) NUP=NM1M3	80X10265
	WRITE(6,1013) Z	80X10266
	WRITE(6,1014) (XZ(J),J=1,NUP)	80X10267
600	IF (NBVRUL.GE.3 .AND. NTIGHT.EQ.1) RETURN	80X10268
C	DETERMINE THE SLOPES ASSOCIATED WITH THE OPTIMAL OBJECTIVE VALUE.	80X10269
	CALL SLOPES (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,PJ,XJ,SD,	80X10270
	1 S1,B,ND1,ND2,ND3,ND4,ND5,NC8)	80X10271
	RETURN	80X10272
1000	FORMAT(1H0,50(1H*)/(H0NODE ,I5/20H0BRANCHED FROM NODE ,I5)	80X10273
1001	FORMAT(23H0BRANCHING ON VARIABLE ,I5,27H WHICH IS CONCAVE VARIABLE	80X10274
	1 ,I5)	80X10275
1002	FORMAT(23H0BRANCHING ON VARIABLE ,I5,27H WHICH IS INTEGER VARIABLE	80X10276
	1 ,I5)	80X10277
1003	FORMAT(7H0TSIG =,E15.6)	80X10278
1004	FORMAT(76H0THE SUBPROGRAM NEED NOT BE SOLVED AS IT IS INCLUDED IN	80X10279
	1THE NEXT SUBPROGRAM.)	80X10280
1005	FORMAT(76H0THE SUBPROGRAM NEED NOT BE SOLVED AS IT IS INCLUDED IN	80X10281
	1THE LAST SUBPROGRAM.)	80X10282
1006	FORMAT(27H0THE PROGRAM IS INFEASIBLE.)	80X10283
1007	FORMAT(38H0THE FOLLOWING LIMITS ARE INCOMPATIBLE/	80X10284
	1 9H0VARIABLE,9X,5HLCMER,12X,5HUPPER/	80X10285

EXHIBIT 4 (Continued)

2	2X,6HNUMBER,10X,5HLIMIT,12X,5HLIMIT//)	BOX10286
1008	FORMAT(3X,I5,3X,E15.6,2X,E15.6)	BOX10287
1009	FORMAT(9H0VARIABLE,4X,3HINT,8X,2H(C,12X,5+LOWER,12X,5HUPPER,13X,	BOX10288
1	4HCO ST/2X,6HNUMBER,3X,8HVAFIABLE,2X,8HARIABLE,9X,5HLIMIT,	BOX10289
2	12X,5HLIMIT,9X,11HCOEFFICIENT/12X,6HNUMBER,4X,6HNUMBER//)	BOX10290
1010	FORMAT(3X,I5,2(5X,I5),3X,3(E15.6,2X))	BOX10291
1011	FORMAT(3X,I5,23X,E15.6,2X,E15.6)	BOX10292
1012	FORMAT(29H0THE LAST TABLEAU IS OPTIMAL.)	BOX10293
1013	FORMAT(17H0SOLUTION VALUE =,E15.6)	BOX10294
1014	FORMAT(17H0VARIABLES =,6E15.6/(17X,6E15.6))	BOX10295
1015	FORMAT(11H *****BOX15)	BOX10296
	END	BOX10297

EXHIBIT 4 (Continued)

```

SUBROUTINE BOX17 (INT,ICC,IBV,C2,SIGMAL,SIGMAU,XZ,S0,S1,ND1,ND4, 80X10001
1 ND6,ND8) 80X10002
C DETERMINE LOWER BOUND. SELECT THE BRANCHING VARIABLE. 80X10003
COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVR1,NTITE1,NODRL2,NBVR2, 80X10004
1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUPT, 80X10005
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10) 80X10006
COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2, 80X10007
1 NM1M3,NIP2,NP1,NSUM,NTC,M10 80X10008
COMMON/P3/NOCNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST, 80X10009
1 NLISTS,NFEAS,LSTMX,ITRICT,ITRMAX,BLB,NBRNOD,PBRNOD, 80X10010
2 NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU, 80X10011
3 TSIG,IFEAS,IBRV1,IUPDN1,XBRVR1,IBRV2,IUPDN2,XBRVR2, 80X10012
4 L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ 80X10013
COMMON/P5/IRCUND 80X10014
DIMENSION INT(ND1),ICC(ND1),IBV(ND4) 80X10015
DIMENSION C2(ND1),SIGMAL(ND6),SIGMAU(ND6),XZ(ND6),S0(ND8),S1(ND8) 80X10016
IF(ITRACE.GE.1)WRITE(6,1009) 80X10017
IF(NBVRUL.EQ.5)GOTO420 80X10018
IF(NBVRUL.GE.3)GOTO250 80X10019
C***** 80X10020
C MAXMIN AND MAXMAX BRANCHING VARIABLE SELECTION RULES. 80X10021
C***** 80X10022
BOUNDU=Z 80X10023
BOUNDL=Z 80X10024
PEN0=Z 80X10025
PEN1=Z 80X10026
PEN2=Z 80X10027
IBRV=0 80X10028
JBRV=0 80X10029
KBRV=0 80X10030
IFEAS=1 80X10031
C BEGINNING OF LOOP. 80X10032
DO240I=1,M7 80X10033
IF(I.EQ.MP1)GOTO240 80X10034
J=IBV(I) 80X10035
IF(J.GT.N)GOTO240 80X10036
IF(ICC(J).NE.0)GOTO120 80X10037
IF(INT(J).EQ.0)GOTO240 80X10038
C INTEGER LINEAR VARIABLE. 80X10039
K=XZ(J) 80X10040
X0=K 80X10041
X1=K+1 80X10042
IF(ABS(XZ(J)-X0).LE.TOL2)GOTO240 80X10043
IF(ABS(X1-XZ(J)).LE.TOL2)GOTO240 80X10044
IFEAS=0 80X10045
P0=BIGN 80X10046
P1=P0 80X10047
IF(S0(I).EQ.-BIGN)GOTO100 80X10048
P0=Z + S0(I)*(X0 - XZ(J)) 80X10049
100 IF(S1(I).EQ.BIGN)GOTO110 80X10050
P1=Z + S1(I)*(X1 - XZ(J)) 80X10051
110 IF(IBRV.EQ.0)IBRV=J 80X10052
IF(JBRV.NE.0)GOTO190 80X10053
JBRV=J 80X10054
JUPDN=2 80X10055
IF(P1.LE.P0)GOTO190 80X10056
JUPDN=1 80X10057

```

EXHIBIT 4 (Continued)

GOTO190	80X10058
120 IF (INT(J).NE.0)GOTO160	80X10059
C CONCAVE VARIABLE.	80X10060
130 IF (ABS(XZ(J)-SIGMAL(J)).LE.TOL2)GOTO24G	80X10061
IF (ABS(SIGMAU(J)-XZ(J)).LE.TOL2)GOTO24C	80X10062
K=ICG(J)	80X10063
CALL GETOBJ (K,SIGMAL(J),F0)	80X10064
CALL GETOBJ (K,XZ(J),F1)	80X10065
DELTA=F1 - (F0 + C2(J)*(XZ(J)-SIGMAL(J)))	80X10066
BOUNDU=BOUNDU + DELTA	80X10067
PG=Z + DELTA	80X10068
P1=P0	80X10069
IF (S0(I).EQ.-BIGN)GOTO140	80X10070
T0=Z + S0(I)*(SIGMAL(J) - XZ(J))	80X10071
IF (T0.LT.P0)P0=T0	80X10072
140 IF (S1(I).EQ.BIGN)GOTO150	80X10073
T1=Z + S1(I)*(SIGMAU(J) - XZ(J))	80X10074
IF (T1.LT.P1)P1=T1	80X10075
150 IF (DELTA.EQ.0.0)GOTO190	80X10076
IF (IBRV.EQ.0)IBRV=J	80X10077
IF (JBRV.NE.0)GOTO190	80X10078
JBRV=J	80X10079
JUPDN=2	80X10080
IF (P1.LE.P0)GOTO190	80X10081
JUPDN=1	80X10082
GOTO190	80X10083
C INTEGER CONCAVE VARIABLE.	80X10084
160 K=XZ(J)	80X10085
X0=K	80X10086
X1=K+1	80X10087
IF (ABS(XZ(J)-X0).LE.TOL2)GOTO130	80X10088
IF (ABS(X1-XZ(J)).LE.TOL2)GOTO130	80X10089
IF (#S=0	80X10090
K=ICG(J)	80X10091
CALL GETOBJ (K,SIGMAL(J),F0)	80X10092
CALL GETOBJ (K,X0,F1)	80X10093
CALL GETOBJ (K,X1,F2)	80X10094
F0=SIGN	80X10095
P1=P0	80X10096
IF (S0(I).EQ.-BIGN)GOTO170	80X10097
PG=Z + S0(I)*(X0-XZ(J)) + F1 - (F0 + C2(J)*(X0-SIGMAL(J)))	80X10098
IF (ABS(X0-SIGMAL(J)).LE.TOL2)GOTO170	80X10099
T0=Z + S0(I)*(SIGMAL(J)-XZ(J))	80X10100
IF (T0.LT.P0)P0=T0	80X10101
170 IF (S1(I).EQ.BIGN)GOTO180	80X10102
P1=Z + S1(I)*(X1-XZ(J)) + F2 - (F1 + C2(J)*(X1-SIGMAL(J)))	80X10103
IF (ABS(SIGMAU(J)-X1).LE.TOL2)GOTO180	80X10104
T1=Z + S1(I)*(SIGMAU(J)-XZ(J))	80X10105
IF (T1.LT.P1)P1=T1	80X10106
180 IF (IBRV.EQ.0)IBRV=J	80X10107
IF (JBRV.NE.0)GOTO190	80X10108
JBRV=J	80X10109
JUPDN=2	80X10110
IF (P1.LE.P0)GOTO190	80X10111
JUPDN=1	80X10112
C ALL VARIABLE TYPES.	80X10113
C DETERMINE THE LOWER BOUND.	80X10114

EXHIBIT 4 (Continued)

190 PENA=P0	BOX10115
IF (P1.LT.PENA)PENA=P1	BOX10116
IF (PENA.LE.BOUNDL)GOTO200	BOX10117
BOUNDL=PENA	BOX10118
200 IF ((S0(I).EQ.-BIGN .OR. S1(I).EQ.BIGN) .AND. NTIGHT.EQ.C)GOTO240	BOX10119
C DETERMINE THE MAXMIN SELECTION.	BOX10120
IF (PENA.LE.PEN0)GOTO210	BOX10121
PEN0=PENA	BOX10122
IBRV=J	BOX10123
C DETERMINE THE MAXMAX SELECTION.	BOX10124
210 PENB=P0	BOX10125
JUD=2	BOX10126
IF (P1.LE.PENB)GOTO220	BOX10127
PENB=P1	BOX10128
JUD=1	BOX10129
220 IF (PENB.LE.PEN1)GOTO230	BOX10130
PEN1=PENB	BOX10131
JUPDN=JUD	BOX10132
JBRV=J	BOX10133
C DETERMINE THE MAXMAX SELECTION TAKEN OVER THOSE VARIABLES FOR WHICH	BOX10134
C THE MIN IS GREATER THAN Z.	BOX10135
230 IF (ABS(PENA - Z).LE.EPSI)GOTO240	BOX10136
IF (PENB.LE.PEN2)GOTO240	BOX10137
PEN2=PENB	BOX10138
KUPDN=JUD	BOX10139
KBRV=J	BOX10140
240 CONTINUE	BOX10141
C END OF LOOP.	BOX10142
GOTO290	BOX10143
C*****	BOX10144
C MIXED INTEGER LINEAR PROGRAM WITH THE MOST NONINTEGER	BOX10145
C OR WEIGHTED NONINTEGER BRANCHING VARIABLE SELECTION RULE.	BOX10146
C*****	BOX10147
250 BOUNDU=Z	BOX10148
BOUNDL=Z	BOX10149
PEN1=0.0	BOX10150
PEN2=0.0	BOX10151
JBRV=0	BOX10152
KBRV=0	BOX10153
IFEAS=1	BOX10154
C BEGINNING OF LOOP.	BOX10155
DO280I=1,M7	BOX10156
IF (I.EQ.MP1)GOTO280	BOX10157
J=IBV(I)	BOX10158
IF (J.GT.N)GOTO280	BOX10159
IF (INT(J).EQ.C)GOTO280	BOX10160
K=XZ(J)	BOX10161
X0=K	BOX10162
P0=XZ(J) - X0	BOX10163
P1=1.0 - P0	BOX10164
IF (P0.LE.TOL2)GOTO280	BOX10165
IF (P1.LE.TOL2)GOTO280	BOX10166
IFEAS=0	BOX10167
C DETERMINE THE MOST NONINTEGER SELECTION.	BOX10168
PENB=P0	BOX10169
JUD=1	BOX10170
IF (P1.GE.PENB)GOTO260	BOX10171

EXHIBIT 4 (Continued)

```

PENB=P1
JUD=2
260 IF(PENB.LE.PEN1)GOTO270
PEN1=PENB
JUPCN=JUD
JBRV=J
C DETERMINE THE WEIGHTED NONINTEGER SELECTION.
270 PENB=PENB*ABS(C2(J))
IF(PENB.LE.PEN2)GOTO280
PEN2=PENB
KUPDN=JUD
KBRV=J
280 CONTINUE
C END OF LOOP.
C*****
C COMMON LOGIC FOR MAXMIN, MAXMAX, MOST NONINTEGER AND WEIGHTED
C NONINTEGER BRANCHING VARIABLE SELECTION RULES.
C*****
290 IF(ITYPE.NE.1)GOTO320
IF(IROUND.EQ.0)GOTO320
C ROUND THE LOWER BOUND UP IF THE OBJECTIVE FUNCTION IS INTEGER VALUED.
IF(BOUNDL.LE.0.0)GOTO300
INTB0=BOUNDL + 1.0 + EPSIM
GOTO310
300 INTB0=BOUNDL + EPSIM
310 BOUNDL=INTB0
320 IF(KBRV.NE.0)GOTO330
KBRV=JBRV
KUPDN=JUPDN
C SELECT THE BRANCHING VARIABLE FOR A ONE PHASE METHOD OR PHASE 1 OF
C A TWO PHASE METHOD.
330 GOTO(340,350,360,350,350),NBVRL1 + 1
340 IBRVR1=IBRV
IUPDN1=2
GOTO370
350 IBRVR1=JBRV
IUPDN1=JUPDN
GOTO370
360 IBRVR1=KBRV
IUPDN1=KUPDN
370 XBRVR1=0.0
IF(IBRVR1.NE.0)XBRVR1=XZ(IBRVR1)
IF(NSTRAT.EQ.1)GOTO440
C SELECT THE BRANCHING VARIABLE FOR PHASE 2 OF A TWO PHASE METHOD.
GOTO(380,390,400,390,400),NBVRL2 + 1
380 IBRVR2=IBRV
IUPCN2=2
GOTO410
390 IBRVR2=JBRV
IUPDN2=JUPDN
GOTO410
400 IBRVR2=KBRV
IUPDN2=KUPDN
410 XBRVR2=0.0
IF(IBRVR2.NE.0)XBRVR2=XZ(IBRVR2)
GOTO440
C*****

```

```

80X10172
80X10173
80X10174
80X10175
80X10176
80X10177
80X10178
80X10179
80X10180
80X10181
80X10182
80X10183
80X10184
80X10185
80X10186
80X10187
80X10188
80X10189
80X10190
80X10191
80X10192
80X10193
80X10194
80X10195
80X10196
80X10197
80X10198
80X10199
80X10200
80X10201
80X10202
80X10203
80X10204
80X10205
80X10206
80X10207
80X10208
80X10209
80X10210
80X10211
80X10212
80X10213
80X10214
80X10215
80X10216
80X10217
80X10218
80X10219
80X10220
80X10221
80X10222
80X10223
80X10224
80X10225
80X10226
80X10227
80X10228

```

AD-A062 012

NAVAL SURFACE WEAPONS CENTER DAHLGREN LAB VA
SOLUTION OF THE INTEGER CONCAVE PROGRAM USING THE IC PHI N ALGO--ETC(U)
NOV 78 H W LOOMIS

F/G 12/2

UNCLASSIFIED

NSWC/DL-TR-3120-VOL-2

NL

2 OF 2
AD
A062 012



END
DATE
FILMED
3-79
DDC

EXHIBIT 4 (Continued)

C CONCAVE NONLINEAR PROGRAM WITH THE CONVENTIONAL	80X10229
C BRANCHING VARIABLE SELECTION RULE.	80X10230
C.....	80X10231
420 BOUNDU=Z	80X10232
PEN=0.0	80X10233
IBRVR1=0	80X10234
DO430 I=1,M7	80X10235
IF (I.EQ.MP1) GOTO430	80X10236
J=IBV(I)	80X10237
IF (J.GT.N) GOTO430	80X10238
K=ICC(J)	80X10239
IF (K.EQ.0) GOTO430	80X10240
CALL GETOBJ (K,SIGMAL(J),F0)	80X10241
CALL GETOBJ (K,XZ(J),F1)	80X10242
DELTA=F1 - (F0 + C2(J)*(XZ(J)-SIGMAL(J)))	80X10243
BOUNDU=BOUNDU + DELTA	80X10244
IF (DELTA.LE.PEN) GOTO430	80X10245
PEN=DELTA	80X10246
IBRVR1=J	80X10247
430 CONTINUE	80X10248
XBRVR1=0.0	80X10249
IF (IBRVR1.NE.0) XBRVR1=XZ (IBRVR1)	80X10250
BOUNDL=Z	80X10251
IFEAS=1	80X10252
C.....	80X10253
C PRINT OUT THE RESULTS.	80X10254
C.....	80X10255
440 IF (IOUTPT.EQ.0) GOTO540	80X10256
IF (NSTRAT.EQ.2 .AND. LPHASE.EQ.2) GOTO500	80X10257
IF (IBRVR1.EQ.0) GOTO530	80X10258
IF (1.LE.NBURL1 .AND. NBURL1.LE.4) GOTO450	80X10259
WRITE(6,1000) IBRVR1	80X10260
GOTO470	80X10261
450 IF (IUPDN1.EQ.2) GOTO460	80X10262
WRITE(6,1001) IBRVR1	80X10263
GOTO470	80X10264
460 WRITE(6,1002) IBRVR1	80X10265
470 IF (NSTRAT.EQ.1) GOTO530	80X10266
IF (IBRVR2.EQ.0) GOTO530	80X10267
IF (1.LE.NBURL2 .AND. NBURL2.LE.4) GOTO480	80X10268
WRITE(6,1003) IBRVR2	80X10269
GOTO530	80X10270
480 IF (IUPDN2.EQ.2) GOTO490	80X10271
WRITE(6,1004) IBRVR2	80X10272
GOTO530	80X10273
490 WRITE(6,1005) IBRVR2	80X10274
GOTO530	80X10275
500 IF (IBRVR2.EQ.0) GOTO530	80X10276
IF (1.LE.NBURL2 .AND. NBURL2.LE.4) GOTO510	80X10277
WRITE(6,1000) IBRVR2	80X10278
GOTO530	80X10279
510 IF (IUPDN2.EQ.2) GOTO520	80X10280
WRITE(6,1001) IBRVR2	80X10281
GOTO530	80X10282
520 WRITE(6,1002) IBRVR2	80X10283
530 WRITE(6,1006) BOUNDL	80X10284
IF (IFEAS.EQ.1) WRITE(6,1007) BOUNDU	80X10285

EXHIBIT 4 (Continued)

540 IF(NODE.NE.1 .OR. IBUROP.EQ.0)RETURN	80X10286
C ESTABLISH THE INITIAL BEST UPPER BOUND.	80X10287
UNOT=BOUNDL + PCBUB	80X10288
IF(IOUTPT.NE.0)WRITE(6,1008)UNOT	80X10289
RETURN	80X10290
1000 FORMAT(10H0VARIABLE ,I5,27H IS THE BRANCHING VARIABLE.)	80X10291
1001 FORPAT(10H0VARIABLE ,I5,69H IS THE BRANCHING VARIABLE. CONTINUE BR	80X10292
1ANCHING FROM THE LOWER BRANCH.)	80X10293
1002 FORMAT(10H0VARIABLE ,I5,69H IS THE BRANCHING VARIABLE. CONTINUE BR	80X10294
1ANCHING FROM THE UPPER BRANCH.)	80X10295
1003 FORMAT(10H0VARIABLE ,I5,39H IS THE BRANCHING VARIABLE FOR PHASE 2.	80X10296
1)	80X10297
1004 FORMAT(10H0VARIABLE ,I5,81H IS THE BRANCHING VARIABLE FOR PHASE 2.	80X10298
1 CONTINUE BRANCHING FROM THE LOWER BRANCH.)	80X10299
1005 FORMAT(10H0VARIABLE ,I5,81H IS THE BRANCHING VARIABLE FOR PHASE 2.	80X10300
1 CONTINUE BRANCHING FROM THE UPPER BRANCH.)	80X10301
1006 FORPAT(14H0LOWER BOUND =,E15.6)	80X10302
1007 FORPAT(14H0UPPER BOUND =,E15.6)	80X10303
1008 FORPAT(31H0THE PHASE 1 BEST UPPER BOUND =,E15.6)	80X10304
1009 FORMAT(11H *****80X17)	80X10305
END	80X10306

EXHIBIT 4 (Continued)

```

SUBROUTINE BOX23 (INUSE,XNOT,CAPP,XZ,ND1,ND6,ND10)
C UPDATE THE BEST UPPER BOUND. IF IN PHASE 1, MERGE THE SUBLIST INTO
C THE LIST AND ENTER PHASE 2. EDIT THE LIST.
COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVR1,NTITE1,NODRL2,NBVR2,
1 NTITE2,MXLIST,LISTOP,ITAPE,IF6,MXITER,MBINV,IOUTPT,
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(1)
COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
1 NLISTS,NFEAS,LSTHX,ITRTOT,ITRMAX,BLB,NBRNOD,PBRNOD,
2 NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
3 TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
4 LIG,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
DIMENSION INUSE(ND10)
DIMENSION XNOT(ND1),CAPP(ND10),XZ(ND6)
IF(ITRACE.GE.1)WRITE(6,1004)
C UPDATE THE BEST UPPER BOUND.
NODNOT=NODE
UNOT=BOUNDU
DO100J=1,N
100 XNOT(J)=XZ(J)
C EDIT THE LIST.
NDELET=0
DO110I=1,MXLIST
IF(INUSE(I).EQ.0)GOTO113
IF(CAPP(I).LT.(1.-TOL1)*UNOT)GOTO113
INUSE(I)=0
NDELET=NDELET+1
110 CONTINUE
IF(IOUTPT.NE.0)WRITE(6,1000)NODNOT,UNOT
IF(NDELET.EQ.0)GOTO120
NLIST=NLIST-NDELET
IF(IOUTPT.EQ.0)GOTO120
WRITE(6,1001)NDELET
WRITE(6,1002)NLIST
120 IF(ITYPE.EQ.2)RETURN
IF(LPHASE.EQ.2)RETURN
C ENTER PHASE 2.
LPHASE=2
IF(NSTRAT.EQ.1)GOTO130
NODRUL=NODRL2
NBVRUL=NBVR2
NTIGHT=NTITE2
130 IF(IOUTPT.NE.0)WRITE(6,1003)
IF(IBUBOP.EQ.0)RETURN
IBUBOP=0
C MERGE THE SUBLIST INTO THE LIST.
DO140I=1,MXLIST
IF(INUSE(I).GE.0)GOTO140
INUSE(I)=-INUSE(I)
140 CONTINUE
RETURN
1000 FORMAT(6H NODE ,I5,35H PROVIDES THE NEW BEST UPPER BOUND ,E15.6)
1001 FORMAT(46H THE NUMBER OF NODES DELETED FROM THE LIST IS ,I5)
1002 FORMAT(26H THE CURRENT LIST SIZE IS ,I5)
1003 FORMAT(14H ENTER PHASE 2)
1004 FORMAT(11H *****BOX23)
END

```

BOX20001
 BOX20002
 BOX20003
 BOX20004
 BOX20005
 BOX20006
 BOX20007
 BOX20008
 BOX20009
 BOX20010
 BOX20011
 BOX20012
 BOX20013
 BOX20014
 BOX20015
 BOX20016
 BOX20017
 BOX20018
 BOX20019
 BOX20020
 BOX20021
 BOX20022
 BOX20023
 BOX20024
 BOX20025
 BOX20026
 BOX20027
 BOX20028
 BOX20029
 BOX20030
 BOX20031
 BOX20032
 BOX20033
 BOX20034
 BOX20035
 BOX20036
 BOX20037
 BOX20038
 BOX20039
 BOX20040
 BOX20041
 BOX20042
 BOX20043
 BOX20044
 BOX20045
 BOX20046
 BOX20047
 BOX20048
 BOX20049
 BOX20050
 BOX20051
 BOX20052
 BOX20053
 BOX20054
 BOX20055
 BOX20056

EXHIBIT 4 (Continued)

```

SUBROUTINE BOX25 (INUSE,IMS,C2,CAPP,CAPL,SIGMAL,SIGMAU,SLOLD,      BOX20001
1      SUOLD,C2OLD,FMS,ND1,ND6,ND9,ND11,ND11,NDMS2,          BOX20002
2      NDMS3)                                                BOX20003
C STORE NODE LNODE IN THE SUBLIST (WHEN IN PHASE 1 AND LOWER BOUND LESS      BOX20004
C THAN BEST UPPER BOUND) OR IN THE LIST (OTHERWISE).          BOX20005
COMMON/P1/N,M,ITYPE,NSTRAT,NOORL1,NBVR1,NTITE1,NOORL2,NBVR2,      BOX20006
1      NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,NBINV,IOUTPT,      BOX20007
2      ITRACE,MSTART,TIME1,TOL1,TOL2,PCSUB,ALPHA(10)          BOX20008
COMMON/P2/EPSI,EPSEM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,      BOX20009
1      NM1M3,N1P2,NP1,NSUM,NTC,M1.                            BOX20010
COMMON/P3/NOONOT,UNOT,IBUBOP,LPHASE,NOORUL,NBVRUL,NTIGHT,NLIST,      BOX20011
1      NLISTS,NFEAS,LSTMX,ITOT,ITRMAX,ELB,NBRNOU,PBRNOU,      BOX20012
2      NBRVAR,NUPDOWN,XBRNOU,TBRNOU,NODE,LNODE,Z,BOUNDL,BOUND2,      BOX20013
3      TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,      BOX20014
4      L10,NITER,NBINV,N7,IPHASE,NPHASE,NM3M7,IALGO,IJOB      BOX20015
DIMENSION INUSE(ND1C),IMS(NDMS2)                               BOX20016
DIMENSION C2(ND1),CAPP(ND1),CAPL(ND1),SIGMAL(ND6),SIGMAU(ND6),      BOX20017
1      SLOLD(ND6),SUOLD(ND6),C2OLD(ND9),FMS(NDMS3)           BOX20018
IF(ITRACE.GE.1)WRITE(6,1009)                                  BOX20019
C CHECK IF THE MAXIMUM LIST SIZE WILL BE EXCEEDED.           BOX20020
IF(NLIST+1.LE.MXLIST)GOTO131                                  BOX20021
IF(IOUTPT.NE.0)WRITE(6,1000)NODE                             BOX20022
C DETERMINE THAT NODE IN THE LIST WITH THE GREATEST LOWER BOUND.      BOX20023
GLB=-BIGN                                                      BOX20024
DO10JI=1,MXLIST                                               BOX20025
IF(INUSE(I).EQ.0)GOTO100                                       BOX20026
IF(CAPP(I).LE.GLB)GOTO100                                       BOX20027
GLB=CAPP(I)                                                    BOX20028
I0=I                                                            BOX20029
100 CONTINUE                                                  BOX20030
IF(BOUNDL.LT.GLB)GOTO110                                       BOX20031
IF(IOUTPT.EQ.0)RETURN                                          BOX20032
WRITE(6,1001)                                                  BOX20033
RETURN                                                         BOX20034
110 IF(IOUTPT.NE.0)WRITE(6,1002)INUSE(I0)                     BOX20035
IF(INUSE(I0).GE.0)GOTO120                                       BOX20036
NLISTS=NLISTS - 1                                             BOX20037
120 NLIST=NLIST - 1                                           BOX20038
INUSE(I0)=0                                                    BOX20039
GOTO150                                                         BOX20040
C FIND AVAILABLE SPACE IN THE LIST.                             BOX20041
130 DO14JI0=1,MXLIST                                          BOX20042
IF(INUSE(I0).EQ.0)GOTO150                                       BOX20043
140 CONTINUE                                                  BOX20044
150 IF(IBUBOP.EQ.0 .OR. BOUNDL.GE.UNOT)GOTO160                BOX20045
C *****                                                    BOX20046
C STORE THE NODE IN THE SUBLIST.                                BOX20047
C *****                                                    BOX20048
INUSE(I0)=-NODE                                               BOX20049
NLISTS=NLISTS + 1                                             BOX20050
GOTO170                                                         BOX20051
C *****                                                    BOX20052
C STORE THE NODE IN THE LIST.                                  BOX20053
C *****                                                    BOX20054
160 INUSE(I0)=NODE                                             BOX20055
170 NLIST=NLIST + 1                                           BOX20056
IF(NLIST.GT.LSTMX)LSTMX=NLIST                                  BOX20057

```

EXHIBIT 4 (Continued)

IF(IOUTPUT.EQ.0)GOTO180	80X20058
WRITE(6,1003)NODE	80X20059
WRITE(6,1004)NLIST	80X20060
C.....	80X20061
C SET CAPP AND CAPL.	80X20062
C.....	80X20063
C SET CAPP FOR THE NODE BEING SAVED.	80X20064
180 CAPP(I0)=BOUNDL	80X20065
IF(NSTRAT.EQ.2)GOTO190	80X20066
IF(NODRL1.EQ.1)GOTO200	80X20067
GOTO280	80X20068
190 IF(NODRL2.EQ.1)GOTO200	80X20069
IF(LPHASE.EQ.1 .AND. NODRL1.EQ.1)GOTO200	80X20070
GOTO280	80X20071
C DETERMINE CAPL FOR THE NODE BEING SAVED.	80X20072
200 IF(1.LE.NBVRUL .AND. NBVRUL.LE.+1)GOTO220	80X20073
IF(LNODE.EQ.2)GOTO210	80X20074
PNEWND=PBRND	80X20075
GOTO270	80X20076
210 PNEWND=PBRND + 1.0	80X20077
GOTO240	80X20078
220 IF(LNODE.EQ.NUPDOWN)GOTO230	80X20079
PNEWND=PBRND	80X20080
GOTO270	80X20081
230 PNEWND=PBRND + 1.0	80X20082
240 IF(18UBOP.EQ.0)GOTO270	80X20083
PMIN=BIGN	80X20084
DO250 I=1, MXLIST	80X20085
IF(I.EQ.I0)GOTO250	80X20086
IF(INUSE(I).EQ.0)GOTO250	80X20087
IF(CAPL(I).LE.PBRND)GOTO250	80X20088
IF(CAPL(I).GE.PMIN)GOTO250	80X20089
PMIN=CAPL(I)	80X20090
250 CONTINUE	80X20091
IF(PMIN.GT.PNEWND)GOTO270	80X20092
C INCREMENT CAPL FOR NODES SUBSEQUENT TO THE NODE BEING SAVED.	80X20093
DO260 I=1, MXLIST	80X20094
IF(INUSE(I).EQ.0)GOTO260	80X20095
IF(CAPL(I).LE.PBRND)GOTO250	80X20096
CAPL(I)=CAPL(I)+1.0	80X20097
260 CONTINUE	80X20098
C SET CAPL FOR THE NODE BEING SAVED.	80X20099
270 CAPL(I0)=PNEWND	80X20100
C.....	80X20101
C WRITE OUT THE DATA FOR THIS NODE.	80X20102
C.....	80X20103
280 IMS(1)=IBVR1	80X20104
IMS(2)=IUPDN1	80X20105
IMS(3)=L10	80X20106
IMS(4)=NITER	80X20107
IMS(5)=NBINV	80X20108
IMS(6)=M7	80X20109
IMS(7)=IPHASE	80X20110
IMS(8)=NPHASE	80X20111
IMS(9)=NM347	80X20112
IF(NSTRAT.EQ.1)GOTO290	80X20113
IMS(10)=IBVR2	80X20114

EXHIBIT 4 (Continued)

IMS(11)=IUPON2	BOX20115
290 FMS(1)=Z	BOX20116
FMS(2)=TSIG	BOX20117
FMS(3)=XBRVR1	BOX20118
IF(NSTRAT.EQ.1)GOTO300	BOX20119
FMS(4)=XBRVR2	BOX20120
300 IF(LNODE.EQ.2)GOTO330	BOX20121
C INTERCHANGE SIGNAL,SLOLD AND SIGMAU,SUOLD.	BOX20122
D0310J=1,N1P2	BOX20123
T1=SLOLD(J)	BOX20124
T2=SUOLD(J)	BOX20125
SLOLD(J)=SIGMAL(J)	BOX20126
SUOLD(J)=SIGMAU(J)	BOX20127
SIGMAL(J)=T1	BOX20128
310 SIGMAU(J)=T2	BOX20129
IF(ITYPE.EQ.1)GOTO360	BOX20130
C INTERCHANGE C2 AND C2OLD.	BOX20131
D0320J=1,N	BOX20132
TEMP=C2OLD(J)	BOX20133
C2OLD(J)=C2(J)	BOX20134
320 C2(J)=TEMP	BOX20135
GOTO360	BOX20136
C PUT SIGNAL INTO SLOLD, SIGMAU INTO SUOLD.	BOX20137
330 D0340J=1,N1P2	BOX20138
SLOLD(J)=SIGMAL(J)	BOX20139
340 SUOLD(J)=SIGMAU(J)	BOX20140
IF(ITYPE.EQ.1)GOTO360	BOX20141
C PUT C2 INTO C2OLD.	BOX20142
D0350J=1,N	BOX20143
350 C2OLD(J)=C2(J)	BOX20144
360 CALL WRITMS (2,IMS,NOMS2,IG)	BOX20145
CALL WRITMS (3,FMS,NOMS3,IG)	BOX20146
IF(LNODE.EQ.2)GOTO390	BOX20147
C INTERCHANGE SIGNAL,SLOLD AND SIGMAU,SUOLD.	BOX20148
D0370J=1,N1P2	BOX20149
T1=SLOLD(J)	BOX20150
T2=SUOLD(J)	BOX20151
SLOLD(J)=SIGMAL(J)	BOX20152
SUOLD(J)=SIGMAU(J)	BOX20153
SIGMAL(J)=T1	BOX20154
370 SIGMAU(J)=T2	BOX20155
IF(ITYPE.EQ.1)GOTO390	BOX20156
C INTERCHANGE C2 AND C2OLD.	BOX20157
D0380J=1,N	BOX20158
TEMP=C2OLD(J)	BOX20159
C2OLD(J)=C2(J)	BOX20160
380 C2(J)=TEMP	BOX20161
C *****	BOX20162
C PRINT OUT THE LIST.	BOX20163
C *****	BOX20164
390 IF(IOUTPUT.LE.2)RETURN	BOX20165
INDEX=0	BOX20166
IF(NSTRAT.EQ.2)GOTO400	BOX20167
IF(NODRL1.EQ.1)GOTO420	BOX20168
GOTO410	BOX20169
400 IF(NODRL2.EQ.1)GOTO420	BOX20170
IF(LPHASE.EQ.1 .AND. NODRL1.EQ.1)GOTO420	BOX20171

EXHIBIT 4 (Continued)

410 INDEX=1	BOX20172
WRITE (6,1005)	BOX20173
GOTO430	BOX20174
420 WRITE (6,1006)	BOX20175
430 DO45, I=1, MXLIST	BOX20176
IF (INUSE(I).EQ.0) GOTO450	BOX20177
IF (INDEX.EQ.0) GOTO440	BOX20178
WRITE (6,1007) INUSE(I), CAPP(I)	BOX20179
GOTO450	BOX20180
440 WRITE (6,1008) INUSE(I), CAPP(I), CAPL(I)	BOX20181
450 CONTINUE	BOX20182
RETURN	BOX20183
1000 FORMAT (48H0LIST SIZE EXCEEDED WITH ATTEMPT TO RECORD NODE ,I5)	BOX20184
1001 FORMAT (29H THE NODE IS NOT BEING SAVED.)	BOX20185
1002 FORMAT (6H NODE ,I5,31H IS BEING PURGED FROM THE LIST.)	BOX20186
1003 FORMAT (6H NODE ,I5,19H SAVED IN THE LIST.)	BOX20187
1004 FORMAT (26H0THE CURRENT LIST SIZE IS ,I5)	BOX20188
1005 FORMAT (1H0, 3X, 4HNODE, 9X, 5HLOWER/17X, 5HBOUND//)	BOX20189
1006 FORMAT (1H0, 3X, 4HNODE, 9X, 5HLOWER, 8X, 10HPROCESSING/17X, 5HBOUND,	BOX20190
1 11X, 5HORDER//)	BOX20191
1007 FORMAT (3X, I5, 2X, E15.6)	BOX20192
1008 FORMAT (3X, I5, 2X, E15.6, 2X, F11.0)	BOX20193
1009 FORMAT (11H *****BCX25)	BOX20194
END	BOX20195

EXHIBIT 4 (Continued)

```

SUBROUTINE ADJUST (V1,V2,V3,V4,T1,T2,T3,T4,SL,SU)
C ADJUST THE LOWER AND UPPER LIMITS ON A VARIABLE USING THE BEST
C UPPER BOUND.
COMMON/P1/N,M, ITYPE,NSTRAT,NODRL1,NBVR1,NTITE1,NODRL2,NBVR2,
1     NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MEINV,IOUTPT,
2     ITRACE,MSTART,TIME1,TOL1,TOL2,PCGUB,ALPHA(10)
COMMON/P3/NUONOT,UNOT,IBUOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
1     NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,BLB,NBRNOD,PBRNOD,
2     NBRVAR,NUPDOWN,XBRNOD,YBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
3     TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
4     LIJ,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
IF(ITRACE.GE.2)WRITE(6,1000)
C ASSUME THAT THE CURVE CONSISTS OF TWO LINEAR SEGMENTS, ONE CONNECTING
C THE POINTS (V1,T1) AND (V2,T2), THE OTHER CONNECTING THE POINTS
C (V3,T3) AND (V4,T4).
SL=V1
IF(T1.LE.UNOT)GOTO130
IF(T2.GE.UNOT)GOTO100
SL=V2 + (UNOT - T2)*(V1 - V2)/(T1 - T2)
GOTO130
100 IF(T3.GT.UNOT)GOTO110
SL=V3
GOTO130
110 IF(T4.GE.UNOT)GOTO120
SL=V4 + (UNOT - T4)*(V3 - V4)/(T3 - T4)
GOTO130
120 SL=V4
130 SU=V4
IF(T4.LE.UNOT)RETURN
IF(T3.GE.UNOT)GOTO140
SU=V3 + (UNOT - T3)*(V3 - V4)/(T3 - T4)
RETURN
140 IF(T2.GT.UNOT)GOTO150
SU=V2
RETURN
150 IF(T1.GE.UNOT)GOTO160
SU=V1 + (UNOT - T1)*(V1 - V2)/(T1 - T2)
RETURN
160 SU=V1
RETURN
1000 FORMAT(12H *****ADJUST)
END

```

ADJU0001
ADJU0002
ADJU0003
ADJU0004
ADJU0005
ADJU0006
ADJU0007
ADJU0008
ADJU0009
ADJU0010
ADJU0011
ADJU0012
ADJU0013
ADJU0014
ADJU0015
ADJU0016
ADJU0017
ADJU0018
ADJU0019
ADJU0020
ADJU0021
ADJU0022
ADJU0023
ADJU0024
ADJU0025
ADJU0026
ADJU0027
ADJU0028
ADJU0029
ADJU0030
ADJU0031
ADJU0032
ADJU0033
ADJU0034
ADJU0035
ADJU0036
ADJU0037
ADJU0038
ADJU0039
ADJU0040
ADJU0041
ADJU0042

EXHIBIT 4 (Continued)

```

SUBROUTINE BINVRT (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI, BINV0001
1          U,PJ,BINV,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7) BINV0002
C COMPUTE THE BASIS INVERSE CORRESPONDING TO THE BASIS SPECIFIED BINV0003
C IN ARRAY IBV. BINV0004
COMMON/P1/N,M,ITYFE,NSTRAT,NOORL1,NBVRL1,NTITE1,NOORL2,NBVRL2, BINV0005
1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,M8INV,IOUTPT, BINV0006
2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCSUB,ALPHA(10) BINV0007
COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2, BINV0008
1          NM1M3,N1P2,NP1,NSUM,NTC,M10 BINV0009
COMMON/P3/NOODN01,UNOT,IBUBOP,LPHASE,NOORL,NBVRL,NTIGHT,NLIST, BINV0010
1          NLISTS,NFEAS,LSTMX,ITRT(1),ITRMAX,BLE,NBRNOD,PBRNOD, BINV0011
2          NBRVAR,NUPDOWN,XERNOC,TBRNOD,NODE,LNODE,Z,BOUNDL,BCUNDU, BINV0012
3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2, BINV0013
4          L10,NITER,N8INV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ BINV0014
DIMENSION IS(ND4),IBV(ND4),NBV(N05),IUPPER(N05) BINV0015
DIMENSION BI(ND4),U(ND6),PJ(ND4),B(ND4,ND4) BINV0016
IF(ITRACE.GE.1)WRITE(6,1000) BINV0017
C INITIALIZE THE BASIS MATRIX. BINV0018
DO100I=1,M7 BINV0019
DO100J=1,M7 BINV0020
100 B(I,J)=0.0 BINV0021
DO130J=1,M7 BINV0022
KIND=IBV(J) BINV0023
DO110I=1,M7 BINV0024
110 PJ(I)=0.0 BINV0025
CALL GETCOL (NZ,NF,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,N05, BINV0026
1          KIND,NZEROS) BINV0027
DO120I1=1,NZEROS BINV0028
I=IS(I1) BINV0029
120 B(I,J)=PJ(I) BINV0030
B(MP1,J)=PJ(MP1) BINV0031
IF(IPHASE.EQ.2)B(MP2,J)=PJ(MP2) BINV0032
130 CONTINUE BINV0033
C INITIALIZE THE RIGHT-HAND-SIDE, ADJUSTING FOR VARIABLES AT UPPER BINV0034
C BOUND IF NECESSARY. BINV0035
NRH=N+M3+M7+1 BINV0036
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,N05, BINV0037
1          NRH,NZEROS) BINV0038
DO140J=1,M7 BINV0039
140 BI(J)=PJ(J) BINV0040
DO170K=1,L10 BINV0041
IF(IUPPER(K).EQ.0)GOTO170 BINV0042
INDEX=NBV(K) BINV0043
DO150I=1,M7 BINV0044
150 PJ(I)=0.0 BINV0045
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,N05, BINV0046
1          INDEX,NZEROS) BINV0047
DO160I1=1,NZEROS BINV0048
I=IS(I1) BINV0049
160 BI(I)=BI(I) - PJ(I)*U(INDEX) BINV0050
BI(MP1)=BI(MP1) - PJ(MP1)*U(INDEX) BINV0051
IF(IPHASE.EQ.2)BI(MP2)=BI(MP2) - PJ(MP2)*U(INDEX) BINV0052
170 CONTINUE BINV0053
C OBTAIN THE BASIS INVERSE AND THE CORRESPONDING RIGHT-HAND-SIDE. BINV0054
CALL INVERT (IS,BI,PJ,BINV,B,ND4,ND7) BINV0055
RETURN BINV0056
1000 FORMAT(12H *****BINVRT) BINV0057
END BINV0058

```


EXHIBIT 4 (Continued)

SUBROUTINE INPUT1 (NZ,NF,ND1)	INPU0001
C READ THE NUMBER OF LESS THAN, EQUALITY, GREATER THAN CONSTRAINTS.	INPU0002
C READ THE NUMBER OF NONZERO ENTRIES BY COLUMN. DEVELOP THE COLUMN	INPU0003
C POINTERS AND TOTAL STORAGE REQUIRED.	INPU0004
COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,	INPU0005
1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,M&INV,IOUTPT,	INPU0006
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(1J)	INPU0007
COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,	INPU0008
1 NM1M3,N1P2,NP1,NSUM,NTC,M10	INPU0009
DIMENSION NZ(ND1),NP(ND1)	INPU0010
IF(ITRACE.GE.1)WRITE(6,1004)	INPU0011
READ(ITAPE,1000)M1,M2,M3	INPU0012
WRITE(6,1001)M1,M2,M3	INPU0013
M4=M2+M3	INPU0014
N1=N+(M3+M1)+(M2+M3)	INPU0015
MP1=M+1	INPU0016
MP2=M+2	INPU0017
NM3=N+M3	INPU0018
NM1M2=N-M1-M2	INPU0019
NM1M3=N+M1+M3	INPU0020
N1P2=N1+2	INPU0021
NP1=N+1	INPU0022
READ(ITAPE,1000)(NZ(J),J=1,N)	INPU0023
WRITE(6,1002)(NZ(J),J=1,N)	INPU0024
NP(1)=0	INPU0025
DO100 J=2,N	INPU0026
100 NP(J)=NP(J-1)+NZ(J-1)	INPU0027
NSUM=NP(N)+NZ(N)	INPU0028
PERCT=NSUM	INPU0029
DENOM=M*N	INPU0030
PERCT=100.*PERCT/DENOM	INPU0031
WRITE(6,1003)NSUM,PERCT	INPU0032
RETURN	INPU0033
1000 FORMAT(16I5)	INPU0034
1001 FORMAT(38H0NUMBER OF CONSTRAINTS BY TYPE =,3I5)	INPU0035
1002 FORMAT(38H0NUMBER OF NONZERO ENTRIES BY COLUMN =,16I5/(38X,16I5))	INPU0036
1003 FORMAT(38H0TOTAL NUMBER OF NONZERO ENTRIES =,I10,	INPU0037
1 15H (A DENSITY OF ,F5.1,10H PERCENT).)	INPU0038
1004 FORMAT(12H *****INPUT1)	INPU0039
END	INPU0040

EXHIBIT 4 (Continued)

SUBROUTINE INPUT2 (NZ,NP,IR,IA,ND1,ND2)	INPUJL11
C READ THE CONSTRAINT MATRIX COLUMN-BY-COLUMN. IT IS ASSUMED THAT	INPUJ002
C THE CONSTRAINTS ARE ORDERED (LESS THAN, EQUALITY, GREATER THAN	INPUJ003
C CONSTRAINTS).	INPUJ004
COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,	INPUJL15
1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,	INPUJ006
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(1J)	INPUJ007
COMMON/P2/EPSI,EPSIM,BIGN,BEGTH,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,	INPUJ008
1 NM1M3,N1P2,NP1,NSUM,NTC,M1L	INPUJL19
DIMENSION NZ(ND1),NP(ND1),IR(ND2),IA(ND2)	INPUJ010
IF(ITRACE.GE.1)WRITE(6,1003)	INPUJ011
DO10JJ=1,N	INPUJ012
K1=NP(J)+1	INPUJL13
K2=NP(J)+NZ(J)	INPUJ014
READ(ITAPE,1000)(IR(K),IA(K),K=K1,K2)	INPUJ015
100 WRITE(6,1001)J,(IR(K),IA(K),K=K1,K2)	INPUJ016
READ(ITAPE,1000)NTC	INPUJL17
WRITE(6,1002)NTC	INPUJ018
RETURN	INPUJ019
1000 FORMAT(16I5)	INPUJ020
1001 FORMAT(8HGCOLUMN ,I5,16H, ROW/CONSTANT =,16I5/(29X,16I5))	INPUJL21
1002 FORMAT(22HNUMBER OF CONSTANTS =,I5)	INPUJ022
1003 FORMAT(12H *****INPUT2)	INPUJ023
END	INPUJ024

EXHIBIT 4 (Continued)

```

SUBROUTINE INPUT3 (INT, ICC, NV, TC, BORIG, C2, SIGMAL, SIGMAU, V, ND1, ND3, INPU0001
1          ND4, ND6) INPU0002
C READ THE TABLE OF CONSTANTS, THE RIGHT-HAND-SIDE, THE LOWER AND INPU0003
C UPPER BOUNDS, THE COST DATA, AND THE LISTS OF INTEGER AND CONCAVE INPU0004
C VARIABLES. INPU0005
COMMON/P1/N, M, ITYPE, NSTRAT, NOORL1, NBVRL1, NTITE1, NOORL2, NBVRL2, INPU0006
1          NTITE2, MXLIST, LISTOP, ITAPE, IFB, MXITER, MBINV, IOUPT, INPU0007
2          ITRACE, MSTART, TIME1, TOL1, TOL2, PC8UB, ALPHA(10) INPU0008
COMMON/P2/EPSI, EPSIM, BIGN, BEGTH, M1, M2, M3, M4, N1, MP1, MP2, NM3, NM1M2, INPU0009
1          NM1M3, N1P2, NP1, NSUM, NTC, M10 INPU0010
COMMON/P5/IROUND INPU0011
DIMENSION INT(ND1), ICC(ND1), NV(ND6) INPU0012
DIMENSION TC(ND3), BORIG(ND4), C2(ND1), SIGMAL(ND6), SIGMAU(ND6), INPU0013
1          V(ND6) INPU0014
DATA BIGINT/1.0E+14/ INPU0015
IF(ITRACE.GE.1)WRITE(6,1014) INPU0016
C READ THE TABLE OF CONSTANTS AND THE RIGHT-HAND-SIDES. IT IS ASSUMED INPU0017
C THAT THE RIGHT-HAND-SIDES ARE NONNEGATIVE. INPU0018
READ(ITAPE,1001) (TC(K),K=1,NTC) INPU0019
READ(ITAPE,1001) (BORIG(I),I=1,M) INPU0020
IF(ITAPE.NE.5)REWIND ITAPE INPU0021
BORIG(MP1)=0.0 INPU0022
WRITE(6,1002) (TC(K),K=1,NTC) INPU0023
WRITE(5,1003) (BORIG(I),I=1,M) INPU0024
C READ LOWER AND UPPER BOUNDS ON THE VARIABLES. INPU0025
DO100J=1,N1P2 INPU0026
SIGMAU(J)=0.0 INPU0027
100 SIGMAU(J)=BIGN INPU0028
READ(5,1000)NDN INPU0029
WRITE(6,1004)NDN INPU0030
IF(NDN.EQ.0)GOTO120 INPU0031
READ(5,1000) (NV(K),K=1,NDN) INPU0032
READ(5,1001) (V(K),K=1,VDN) INPU0033
WRITE(5,1005) (NV(K),K=1,NDN) INPU0034
WRITE(5,1006) (V(K),K=1,NDN) INPU0035
DO110K=1,NDN INPU0036
J=NV(K) INPU0037
110 SIGMAU(J)=V(K) INPU0038
120 READ(5,1000)NUP INPU0039
WRITE(5,1007)NUP INPU0040
IF(NUP.EQ.0)GOTO140 INPU0041
READ(5,1000) (NV(K),K=1,NUP) INPU0042
READ(5,1001) (V(K),K=1,VUP) INPU0043
WRITE(5,1005) (NV(K),K=1,NUP) INPU0044
WRITE(5,1008) (V(K),K=1,NUP) INPU0045
DO130K=1,NUP INPU0046
J=NV(K) INPU0047
130 SIGMAU(J)=V(K) INPU0048
C READ COST DATA. INPU0049
140 READ(5,1001) (C2(J),J=1,N) INPU0050
WRITE(5,1009) (C2(J),J=1,N) INPU0051
DO150J=1,N INPU0052
150 INT(J)=0 INPU0053
IF(ITYPE.EQ.2 .OR. ITYPE.EQ.3)GOTO170 INPU0054
C READ THE LIST OF INTEGER VARIABLES. INPU0055
READ(5,1000)NINT INPU0056
READ(5,1000) (NV(K),K=1,NINT) INPU0057

```

EXHIBIT 4 (Continued)

WRITE(6,1010)NINT	INPU0058
WRITE(5,1011)(NV(K),K=1,NINT)	INPU0059
DO160<=1,NINT	INPU0060
J=NV(K)	INPU0061
INT(J)=K	INPU0062
IF(SIGMAU(J).LE.BIGINT)GOTO160	INPU0063
SIGMAU(J)=BIGINT	INPU0064
160 CONTINJE	INPU0065
170 DO180J=1,N	INPU0066
180 ICC(J)=0	INPU0067
IF(ITYPE.EQ.1 .OR. ITYPE.EQ.3)GOTO200	INPU0068
C READ THE LIST OF CONCAVE VARIABLES.	INPU0069
READ(5,1000)NCC	INPU0070
READ(5,1000)(NV(K),K=1,NCC)	INPU0071
WRITE(6,1012)NCC	INPU0072
WRITE(5,1013)(NV(K),K=1,NCC)	INPU0073
DO190<=1,NCC	INPU0074
J=NV(K)	INPU0075
190 ICC(J)=K	INPU0076
200 IF(ITYPE.NE.1)RETURN	INPU0077
C FOR THE MIXED INTEGER LINEAR PROGRAM, DETERMINE IF THE OBJECTIVE	INPU0078
C FUNCTION IS INTEGER VALUED.	INPU0079
IROUND=0	INPU0080
DO220J=1,N	INPU0081
IF(INT(J).NE.0)GOTO210	INPU0082
IF(C2(J).NE.0.0)RETURN	INPU0083
GOTO220	INPU0084
210 IC2=C2(J)	INPU0085
FC2=IC2	INPU0086
IF(C2(J).NE.FC2)RETURN	INPU0087
220 CONTINJE	INPU0088
IROUND=1	INPU0089
RETURN	INPU0090
1000 FORMAT(16I5)	INPU0091
1001 FORMAT(6E12.0)	INPU0092
1002 FORMAT(22H0TABLE OF CONSTANTS =,6E15.6/(22X,6E15.6))	INPU0093
1003 FORMAT(22H0RIGHT-HAND-SIDE =,6E15.6/(22X,6E15.6))	INPU0094
1004 FORMAT(42H0NUMBER OF VARIABLES HAVING LOWER BOUNDS =,I5)	INPU0095
1005 FORMAT(22H0VARIABLES =,16I5/(22X,16I5))	INPU0096
1006 FORMAT(22H0LOWER BOUNDS =,6E15.6/(22X,6E15.6))	INPU0097
1007 FORMAT(42H0NUMBER OF VARIABLES HAVING UPPER BOUNDS =,I5)	INPU0098
1008 FORMAT(22H0UPPER BOUNDS =,6E15.6/(22X,6E15.6))	INPU0099
1009 FORMAT(22H0COST COEFFICIENTS =,6E15.6/(22X,6E15.6))	INPU0100
1010 FORMAT(30H0NUMBER OF INTEGER VARIABLES =,I5)	INPU0101
1011 FORMAT(22H0INTEGER VARIABLES =,16I5/(22X,16I5))	INPU0102
1012 FORMAT(30H0NUMBER OF CONCAVE VARIABLES =,I5)	INPU0103
1013 FORMAT(22H0CONCAVE VARIABLES =,16I5/(22X,16I5))	INPU0104
1014 FORMAT(12H *****INPUT3)	INPU0105
END	INPU0106

EXHIBIT 4 (Continued)

```

SUBROUTINE INPUT4 (NZ, NP, IR, IA, IS, NV, IBV, NBV, IUPPER, TC, BORIG, RHS, INPU0001
1      C2, C1, BI, BN, L, PJ, BINV, B, ND1, ND2, ND3, ND4, ND5, INPU0002
2      ND6, ND7) INPU0003
C ESTABLISH THE INITIAL BASIS, BASIS INVERSE, AND RIGHT-HAND-SIDE FOR INPU0004
C THE LP. INPU0005
COMMON/P1/N, M, ITYPE, NSTRAT, NOCR1, NBVRL1, NTITE1, NODRL2, NBVRL2, INPU0006
1      NTITE2, MXLIST, LISTOP, ITAPE, IFB, MXITER, MBINV, IOUTPT, INPU0007
2      ITRACE, PSTART, TIME1, TOL1, TOL2, PCBUB, ALPHA(10) INPU0008
COMMON/P2/EPSI, EPSIM, BIGN, BEGTH, M1, M2, M3, M4, N1, MP1, MP2, NM3, NM1M2, INPU0009
1      NM1M3, N1P2, NP1, NSUM, NTC, M10 INPU0010
COMMON/P3/NOOBT, UNOT, IEUBOP, LPHASE, NODRUL, NBVRUL, NTIGHT, NLIST, INPU0011
1      NLISTS, NFEAS, LSTMX, ITRCT, ITRMAX, BLB, NBRNOD, PBRNOD, INPU0012
2      NBRVAR, NUPDWN, XBRNOD, TBRNOD, NODE, LNODE, Z, BOUNDL, BCUNDU, INPU0013
3      TSIG, IFEAS, IBVR1, IUPDN1, XBRV1, IBVR2, IUFQ2, XBRVR2, INPU0014
4      L10, NITER, NBINV, M7, IPHASE, NPHASE, NM3M7, IALGO, IE0J INPU0015
DIMENSION NV(ND6), IBV(ND4), NBV(ND5), IUPPER(ND5) INPU0016
DIMENSION BI(ND4), BN(ND5), U(ND6), FJ(ND4), B(ND4, ND4) INPU0017
IF(ITRACE.GE.1)WRITE(6,1003) INPU0018
C READ INITIAL FEASIBLE BASIS. INITIALIZE PARAMETERS USED IN LP. INPU0019
L10=N1-M INPU0020
DO100I=1, L10 INPU0021
IUPPER(I)=0 INPU0022
100 BN(I)=0.0 INPU0023
IF(IFB.EQ.0)GOTO210 INPU0024
C INITIAL FEASIBLE BASIS PROVIDED AS INFUT. INPU0025
READ(5,1000)(IBV(I), I=1, M) INPU0026
IF(M4.EQ.0)GOTO130 INPU0027
DO110I=1, M INPU0028
IF(IBV(I).GT.NM1M3)GOTO120 INPU0029
110 CONTINUE INPU0030
GOTO130 INPU0031
C THERE ARE ARTIFICIAL VARIABLES IN THE INITIAL BASIS. INPU0032
120 IBV(MP1)=N1+1 INPU0033
IBV(MP2)=N1P2 INPU0034
M7=MP2 INPU0035
IPHASE=2 INPU0036
NPHASE=1 INPU0037
CALL OBJ1 (NZ, NP, IR, IA, IS, IBV, TC, BORIG, RHS, C1, ND1, ND2, ND3, ND4, ND5) INPU0038
GOTO140 INPU0039
C THERE ARE NO ARTIFICIAL VARIABLES IN THE INITIAL BASIS. INPU0040
130 IBV(MP1)=N1+1 INPU0041
M7=MP1 INPU0042
IPHASE=1 INPU0043
NPHASE=0 INPU0044
140 WRITE(6,1001)(IBV(I), I=1, M7) INPU0045
C FORM THE LIST OF NON-BASIC VARIABLES. INPU0046
INDEX=0 INPU0047
DO160K=1, NM1M3 INPU0048
DO150I=1, M INPU0049
IF(K.EQ.IBV(I))GOTO160 INPU0050
150 CONTINUE INPU0051
INDEX=INDEX+1 INPU0052
NBV(INDEX)=K INPU0053
160 CONTINUE INPU0054
L10=INDEX INPU0055
C READ NON-BASIC VARIABLES INITIALLY AT UPPER BOUND. INPU0056
READ(5,1000)NUP INPU0057

```

EXHIBIT 4 (Continued)

```

IF (NUP.EQ.0)GOTO20G
READ(5,1000) (NV(K),K=1,NUP)
WRITE(6,10G2) (NV(K),K=1,NUP)
DO100K=1,NUP
INDEX=NV(K)
DO170I=1,L10
IF (NBV(I).EQ.INDEX)GOTO180
170 CONTINUE
180 IUPPER(I)=1
BN(I)=U(INDEX)
190 CONTINUE
200 CONTINUE
C FORM THE BASIS INVERSE AND INITIALIZE THE RIGHT-HAND-SIDE.
NM3M7=N+M3+M7
CALL @INVRT (NZ,NF,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,U,PJ,
1 6INV,0,ND1,ND2,ND3,N[4,ND5,NC6,ND7)
M10=M7
IEOJ=0
NITER=0
NBINV=0
RETURN
C INITIAL FEASIBLE BASIS NOT PROVIDED AS INPUT.
210 IF (M4.EQ.0)GOTO230
M7=MP2
LIMIT=N+M3+1
JCOUNT=N1P2
K=0
DO220I=LIMIT,JCCUNT
K=K+1
220 IBV(K)=I
IPHASE=2
NPHASE=1
CALL OBJ1 (NZ,NP,IR,IA,IS,IBV,TC,@ORIG,RHS,C1,ND1,ND2,ND3,ND4,ND5)
GOTO250
230 M7=MP1
LIMIT=N+M3+1
JCOUNT=N1+1
K=0
DO240I=LIMIT,JCOUNT
K=K+1
240 IBV(K)=I
IPHASE=1
NPHASE=0
250 IF (IOUTPT.GE.3)WRITE(6,1001) (IBV(I),I=1,M7)
DO260I=1,L10
260 NBV(I)=I
NM3M7=N+M3+M7
DO270I=1,M7
DO270J=1,M7
270 B(I,J)=0.0
DO280K=1,M7
280 B(K,K)=1.0
NRH=N+M3+M7+1
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
1 NRH,NZEROS)
DO290J=1,M7
290 BI(J)=PJ(J)

```

```

INPU0058
INPU0059
INPU0060
INPU0061
INPU0062
INPU0063
INPU0064
INPU0065
INPU0066
INPU0067
INPU0068
INPU0069
INPU0070
INPU0071
INPU0072
INPU0073
INPU0074
INPU0075
INPU0076
INPU0077
INPU0078
INPU0079
INPU0080
INPU0081
INPU0082
INPU0083
INPU0084
INPU0085
INPU0086
INPU0087
INPU0088
INPU0089
INPU0090
INPU0091
INPU0092
INPU0093
INPU0094
INPU0095
INPU0096
INPU0097
INPU0098
INPU0099
INPU0100
INPU0101
INPU0102
INPU0103
INPU0104
INPU0105
INPU0106
INPU0107
INPU0108
INPU0109
INPU0110
INPU0111
INPU0112
INPU0113
INPU0114

```

EXHIBIT 4 (Continued)

```
M10=M7
IEOJ=0
NITER=0
NBINV=0
RETURN
1000 FORMAT(16I5)
1001 FORMAT(25H0 INITIAL FEASIBLE BASIS =,16I5/(25X,16I5))
1002 FORMAT(37H0 NON-BASIC VARIABLES AT UPPER BOUND =,16I5/(37X,16I5))
1003 FORMAT(12H *****INPUT4)
END
```

INPU0115
INPU0116
INPU0117
INPU0118
INPU0119
INPU0120
INPUJ121
INPU0122
INPU0123
INPUJ124

EXHIBIT 4 (Continued)

```

SUBROUTINE INPUTS (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI, INPU0001
1 U,PJ,B,ND1,ND2,ND3,ND4,ND5,ND6) INPU0002
C DETERMINE THE APPLICABLE LP ALGORITHM. INPU0003
COMMON/P1/N,M,ITYPE,NSTRAT,NOORL1,NBVRL1,NTITE1,NOORL2,NBVRL2, INPU0004
1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT, INPU0005
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBLB,ALPHA(15) INPU0006
COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2, INPU0007
1 NM1M3,N1P2,NP2,NSUM,NTC,M1 INPU0008
COMMON/P3/NODNOT,UNOT,IUBSOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST, INPU0009
1 NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,BLB,NBRNOD,PBRNOD, INPU0010
2 NBRVAR,NUPDWH,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU, INPU0011
3 TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2, INPU0012
4 L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ INPU0013
DIMENSION IS(ND4),IBV(ND4),NBV(ND5),IUPPER(ND5) INPU0014
DIMENSION BI(ND4),J(ND6),PJ(ND4),B(ND4,ND4) INPU0015
IF(ITRACE.GE.1)WRITE(6,1010) INPU0016
C CHECK PRIMAL FEASIBILITY. INPU0017
IPRIM=0 INPU0018
DO100 I=1,M INPU0019
IF(BI(I).LT.EPSIM)GOTO120 INPU0020
100 CONTINUE INPU0021
DO110 I=1,M INPU0022
I1=IBV(I) INPU0023
IF(U(I1)-BI(I).LT.EPSIM)GOTO120 INPU0024
110 CONTINUE INPU0025
IPRIM=1 INPU0026
120 CONTINUE INPU0027
C CHECK DUAL FEASIBILITY. INPU0028
IDUAL=0 INPU0029
DO150 IPOS=1,L10 INPU0030
KIND=NBV(IPOS) INPU0031
DO130 J=1,M7 INPU0032
130 PJ(I)=0.0 INPU0033
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5, INPU0034
1 KIND,NZEROS) INPU0035
Q1=0.0 INPU0036
DO140 J1=1,NZEROS INPU0037
J=IS(J1) INPU0038
140 Q1=Q1 + B(MP1,J)*PJ(J) INPU0039
Q1=Q1 + B(MP1,MP1)*PJ(MP1) INPU0040
IF(IPHASE.EQ.2)Q1=Q1 + B(MP1,MP2)*PJ(MP2) INPU0041
IF(IUPPER(IPOS).EQ.1)Q1=-Q1 INPU0042
IF(Q1.LT.EPSIM)GOTO160 INPU0043
150 CONTINUE INPU0044
IDUAL=1 INPU0045
160 CONTINUE INPU0046
C SELECT THE ALGORITHM TO BE USED. INPU0047
IF(IOUTPT.LE.2)GOTO170 INPU0048
IF(IPHASE.EQ.2)WRITE(6,1000) INPU0049
IF(IPHASE.EQ.1)WRITE(6,1001) INPU0050
170 IF(IPRIM.EQ.0)GOTO190 INPU0051
IF(IOUTPT.LE.2)GOTO180 INPU0052
WRITE(6,1002) INPU0053
IF(IPHASE.EQ.2)WRITE(6,1006) INPU0054
IF(IPHASE.EQ.1)WRITE(6,1007) INPU0055
180 IALGO=1 INPU0056
RETURN INPU0057

```

EXHIBIT 4 (Continued)

190 IF(IOUTPT.LE.2)GOTO200	INPU0050
WRITE(6,1003)	INPU0059
200 IF(IDUAL.EQ.0)GOTO220	INPU0060
IF(IOUTPT.LE.2)GOTO210	INPU0061
WRITE(6,1004)	INPU0062
WRITE(6,1008)	INPU0063
210 IALGO=2	INPU0064
IF(IPHASE.EQ.1)RETURN	INPU0065
NPHASE=2	INPU0066
M10=MP1	INPU0067
RETURN	INPU0068
220 IF(IOUTPT.LE.2)GOTO230	INPU0069
WRITE(6,1005)	INPU0070
WRITE(6,1009)	INPU0071
230 IALGO=0	INPU0072
RETURN	INPU0073
1000 FORMAT(44H0THERE ARE ARTIFICIALS IN THE INITIAL BASIS.)	INPU0074
1001 FORMAT(47H0THERE ARE NO ARTIFICIALS IN THE INITIAL BASIS.)	INPU0075
1002 FORMAT(38H THE INITIAL BASIS IS PRIMAL FEASIBLE.)	INPU0076
1003 FORMAT(42H THE INITIAL BASIS IS NOT PRIMAL FEASIBLE.)	INPU0077
1004 FORMAT(36H THE INITIAL BASIS IS DUAL FEASIBLE.)	INPU0078
1005 FORMAT(40H THE INITIAL BASIS IS NOT DUAL FEASIBLE.)	INPU0079
1006 FORMAT(54H THE PRIMAL ALGORITHM (TWO PHASE METHOD) WILL BE USED.)	INPU0080
1007 FORMAT(54H THE PRIMAL ALGORITHM (ONE PHASE METHOD) WILL BE USED.)	INPU0081
1008 FORMAT(41H THE DUAL SIMPLEX ALGORITHM WILL BE USED.)	INPU0082
1009 FORMAT(64H NEITHER THE PRIMAL NOR THE DUAL SIMPLEX ALGORITHMS CAN	INPU0083
10E USED.)	INPU0084
1010 FORMAT(12H *****INPUT5)	INPU0085
END	INPU0086

EXHIBIT 4 (Continued)

	SUBROUTINE INVERT (IS, BI, PJ, BINV, E, ND4, ND7)	INVE0001
C	GAUSS-JORDAN METHOD OF MATRIX INVERSION.	INVE0002
	COMMON/P1/N, M, ITYPE, NSTRAT, NOCRL1, NBVRL1, NTITE1, NOORL2, NBVRL2,	INVE0003
1	NTITE2, MXLIST, LISTOP, ITAPE, IFB, MXITER, MBINV, IOUTPT,	INVE0004
2	ITRACE, MSTART, TIME1, TOL1, TOL2, PCBUB, ALPHA (10)	INVE0005
	COMMON/P2/EPSI, EPSIM, BIGN, BEGTH, M1, M2, M3, M4, N1, MP1, MP2, NM3, NM1M2,	INVE0006
1	NM1M3, N1P2, NP1, NSUM, NTC, M10	INVE0007
	COMMON/P3/NODNOT, UNOT, IBUBOF, LPHASE, NODRUL, NBVRUL, NTIGHT, NLIST,	INVE0008
1	NLISTS, AFEAS, LSTMX, ITRCT, ITRMAX, BLE, NBRNOD, PBRNOC,	INVE0009
2	NBRVAR, NUPDWN, XBRNOD, TBFNOD, NODE, LNODE, Z, BOUNDL, BCUNDU,	INVE0010
3	TSIG, IFEAS, IBRVR1, IUPDN1, XBRVR1, IBRVR2, IUPDN2, XBRVR2,	INVE0011
4	L10, NITER, NBINV, M7, IPHASE, NPHASE, NM3M7, IALGO, IECJ	INVE0012
	DIMENSION IS (ND4)	INVE0013
	DIMENSION BI (ND4), PJ (ND4), BINV (ND7, ND7), B (ND4, ND4)	INVE0014
	IF (ITRACE.GE.1) WRITE (6, 1001)	INVE0015
C	SOLVE B (PJ) = BI, DEVELOPING THE INVERSE OF B IN THE PROCESS.	INVE0016
	DO100I=1, M7	INVE0017
	DO100J=1, M7	INVE0018
100	BINV (I, J) = 0.0	INVE0019
	DO110K=1, M7	INVE0020
	BINV (K, K) = 1.0	INVE0021
	PJ (K) = BI (K)	INVE0022
110	IS (K) = J	INVE0023
	DO170L=1, M7	INVE0024
	DO120K=1, M7	INVE0025
	IF (IS (K).NE.0) GOTC120	INVE0026
	IF (ABS (B (K, L)).GT.EPSI) GOTC130	INVE0027
120	CONTINUE	INVE0028
C	CAN DROP OUT OF THIS LOOP ONLY IF A IS ILL-CONDITIONED OR SINGULAR.	INVE0029
	WRITE (6, 1000)	INVE0030
	CALL EXIT	INVE0031
	RETURN	INVE0032
130	IS (K) = L	INVE0033
	T = 1./B (K, L)	INVE0034
	PJ (K) = PJ (K) * T	INVE0035
	DO140J=1, M7	INVE0036
	B (K, J) = B (K, J) * T	INVE0037
140	BINV (K, J) = BINV (K, J) * T	INVE0038
	DO160I=1, M7	INVE0039
	IF (I.EQ.K) GOTC160	INVE0040
	T = B (I, L)	INVE0041
	IF (ABS (T).LE.EPSI) GOTC160	INVE0042
	T1 = PJ (I) - T * PJ (K)	INVE0043
	IF (ABS (T1).LE.EPSI) T1 = 0.0	INVE0044
	PJ (I) = T1	INVE0045
	DO150J=1, M7	INVE0046
	T1 = B (I, J) - T * B (K, J)	INVE0047
	IF (ABS (T1).LE.EPSI) T1 = 0.0	INVE0048
	B (I, J) = T1	INVE0049
	T1 = B INV (I, J) - T * B INV (K, J)	INVE0050
	IF (ABS (T1).LE.EPSI) T1 = 0.0	INVE0051
150	BINV (I, J) = T1	INVE0052
160	CONTINUE	INVE0053
170	CONTINUE	INVE0054
C	BINV CONTAINS THE INVERSE OF B, UP TO A PERMUTATION OF THE ROWS.	INVE0055
	DO180L=1, M7	INVE0056
	I = IS (L)	INVE0057

EXHIBIT 4 (Continued)

```
      BI(I)=PJ(L)
      DO100 J=1,N7
100  B(I,J)=EINV(L,J)
C B NOW CONTAINS THE INVERSE AND BI CONTAINS THE SOLUTION.
      RETURN
1000 FORMAT(60H MATRIX TO BE INVERTED IS ILL-CONDITIONED, PERHAPS SINGU
1LAR.)
1001 FORMAT(12H ****INVERT)
      END
```

```
INVE0058
INVE0059
INVE0060
INVE0061
INVE0062
INVE0063
INVE0064
INVE0065
INVE0066
```

EXHIBIT 4 (Continued)

```

SUBROUTINE OBJ1 (NZ,NP,IR,IA,IS,IBV,TC,BORIG,RHS,C1,ND1,ND2,ND3, 06J10001
1 ND4,ND5) 06J10002
C COMPUTE AND STORE THE PHASE 1 OBJECTIVE FUNCTION. 06J10003
COMMON/P1/N,M,ITYPE,NSTRAT,NDORL1,NBURL1,NTITE1,NDORL2,NBURL2, 06J10004
1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUPT, 06J10005
2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUS,ALPHA(10) 06J10006
COMMON/P2/EPSI,EPSIM,BIGN,BEGTH,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2, 06J10007
1 NM1M3,N1P2,NP1,NSUM,NTC,M10 06J10008
DIMENSION NZ(ND1),NP(ND1),IR(ND2),IA(ND2),IS(ND4),IBV(ND4) 06J10009
DIMENSION TC(ND3),BORIG(ND4),RHS(ND4),C1(ND5) 06J10010
IF(ITRACE.GE.1)WRITE(6,1000) 06J10011
DO100K=1,M 06J10012
100 IS(K)=0 06J10013
DO110K=1,M 06J10014
IF(IBV(K).LE.NM1M3)GOTO110 06J10015
KK=IBV(K)-NM3 06J10016
IS(KK)=1 06J10017
110 CONTINUE 06J10018
DO130J=1,N 06J10019
NPOINT=NP(J) 06J10020
NZEROS=NZ(J) 06J10021
Q1=0.0 06J10022
DO120K=1,NZEROS 06J10023
NPOINT=NPOINT+1 06J10024
KK=IR(NPOINT) 06J10025
IF(IS(KK).EQ.C)GOTO120 06J10026
INDEX=IA(NPCINT) 06J10027
Q1=Q1+TC(INDEX) 06J10028
120 CONTINUE 06J10029
130 C1(J)=-Q1 06J10030
IF(M3.EQ.0)GOTO150 06J10031
KK=M1+M2 06J10032
DO140J=NP1,NM3 06J10033
KK=KK+1 06J10034
C1(J)=0.0 06J10035
IF(IS(KK).EQ.C)GOTO140 06J10036
C1(J)=1.0 06J10037
140 CONTINUE 06J10038
150 BB=0.0 06J10039
CC=0.0 06J10040
LIMIT=M1+1 06J10041
DO160I=LIMIT,M 06J10042
IF(IS(I).EQ.C)GOTO160 06J10043
BB=BB-RHS(I) 06J10044
CC=CC-BORIG(I) 06J10045
160 CONTINUE 06J10046
RHS(MP2)=BB 06J10047
BORIG(MP2)=CC 06J10048
RETURN 06J10049
1000 FORMAT(10H *****OBJ1) 06J10050
END 06J10051

```

EXHIBIT 4 (Continued)

```

SUBROUTINE RSTART (IF, INUSE, IBV, NBV, IUPPER, IMS, F, BI, BN, B, FMS,
1 NI, NF, ND4, ND5, ND10, NDMS2, NDMS3, IENTRY) RSTA0001
C PREPARE A RESTART TAPE OR BEGIN A JOB FROM A PREVIOUSLY PREPARED RSTA0002
C RESTART TAPE. RSTA0003
COMMON/P1/N, M, ITYPE, NSTRAT, NOORL1, NBVRL1, NTITE1, NOORL2, NBVRL2, RSTA0004
1 NTITE2, MXLIST, LISTOP, ITAPE, IFB, FXITER, MBINV, IOUPT, RSTA0005
2 ITRACE, NSTART, TIME1, TOL1, TOL2, PCBLB, ALPHA (10) RSTA0006
COMMON/P2/EPSI, EPSIM, BIGN, BEGTH, M1, M2, M3, M4, N1, MP1, MP2, NM3, NM1M2, RSTA0008
1 NM1M3, NIP2, NP1, NSUM, NTC, M10 RSTA0009
COMMON/P3/NOENOT, UNOT, IBUBOP, LPHASE, NODRUL, NBVRUL, NTIGHT, NLIST, RSTA0010
1 NLISTS, NFEAS, LSTMX, ITRTCT, ITRMAX, BLB, NBRNOD, PBRNOD, RSTA0011
2 NBRVAR, NUPDWN, XBRNOD, TBRNOD, NODE, LNODE, Z, BOUNDL, BCUNDU, RSTA0012
3 TSIG, IFEAS, IBVR1, IUPON1, XBRVR1, IBVR2, IUPON2, XBRVR2, RSTA0013
4 L10, NITER, NBINV, M7, IPHASE, NPHASE, NM3M7, IALGO, IECJ RSTA0014
COMMON/P4/SAVE, KBRAN, X1 RSTA0015
DIMENSION IF (NI), INUSE (ND10), IBV (ND4), NBV (ND5), IUPPER (ND5), RSTA0016
1 IMS (NDMS2) RSTA0017
DIMENSION F (NF), JI (ND4), BN (ND5), B (NC4, NC4), FMS (NDMS3) RSTA0018
IF (ITRACE.GE.1) WRITE (6, 1002) RSTA0019
IF (IENTRY.EQ.1) GOTO130 RSTA0020
C BEGIN A JOB FROM A RESTART TAPE. RSTA0021
WRITE (6, 1000) RSTA0022
REWIND 4 RSTA0023
REWIND 7 RSTA0024
REWIND 8 RSTA0025
READ (7) M10, RSTA0026
1 NODNOT, UNOT, IBUBOP, LPHASE, NODRUL, NBVRUL, NTIGHT, NLIST, RSTA0027
2 NLISTS, NFEAS, LSTMX, ITRTCT, ITRMAX, BLB, NBRNOD, PBRNOD, RSTA0028
3 NBRVAR, NUPDWN, XBRNOD, TBRNOD, NODE, LNODE, Z, BOUNDL, BCUNDU, RSTA0029
4 TSIG, IFEAS, IBVR1, IUPON1, XBRVR1, IBVR2, IUPON2, XBRVR2, RSTA0030
5 L10, NITER, NBINV, M7, IPHASE, NPHASE, NM3M7, IALGO, IECJ, RSTA0031
6 SAVE, KBRAN, X1 RSTA0032
IF (NLIST.EQ.0) GOTO110 RSTA0033
DO100J=1, NLIST RSTA0034
READ (8) I0, (IMS (I), I=1, NDMS2), (FMS (I), I=1, NDMS3) RSTA0035
CALL WRITMS (2, IMS, NDMS2, I0) RSTA0036
CALL WRITMS (3, FMS, NDMS3, I0) RSTA0037
100 CONTINUE RSTA0038
110 IF (2*(NODE/2).NE.NODE) GOTO120 RSTA0039
READ (8) (IBV (I), I=1, ND4), (NBV (I), I=1, ND5), (IUPPER (I), I=1, ND5), RSTA0040
1 (BI (I), I=1, ND4), (BN (I), I=1, ND5), LL1, LL2, LL3, RSTA0041
2 ((B (I, J), I=1, ND4), J=1, ND4) RSTA0042
WRITE (4) (IBV (I), I=1, ND4), (NBV (I), I=1, ND5), (IUPPER (I), I=1, ND5), RSTA0043
1 (BI (I), I=1, ND4), (BN (I), I=1, ND5), LL1, LL2, LL3, RSTA0044
2 ((B (I, J), I=1, ND4), J=1, N(4)) RSTA0045
120 READ (7) (IF (I), I=1, NI), (F (I), I=1, NF) RSTA0046
REWIND 4 RSTA0047
REWIND 7 RSTA0048
REWIND 8 RSTA0049
RETURN RSTA0050
C PREPARE A RESTART TAPE. RSTA0051
130 WRITE (6, 1001) RSTA0052
REWIND 4 RSTA0053
REWIND 9 RSTA0054
REWIND 10 RSTA0055
WRITE (9) M10, RSTA0056
1 NODNOT, UNOT, IBUBOP, LPHASE, NODRUL, NBVRUL, NTIGHT, NLIST, RSTA0057

```

EXHIBIT 4 (Continued)

2	NLISTS, NFEAS, LSTMX, ITRT (T, ITRMAX, BLB, NBRNOD, PBRNOC,	RSTA0058
3	NBRVAR, NUPDOWN, XBRNOD, IBRNOD, NODE, LNODE, Z, BOUNDL, DCUNDU,	RSTA0059
4	TSIG, IFEAS, IBRVR1, IUPON1, XBRVR1, IBRVR2, IUPON2, XBRVR2,	RSTA0060
5	L10, NITER, NAINV, M7, IPHASE, NPHASE, NM3M7, IALGO, IE0J,	RSTA0061
6	SAVE, KBRAN, X1	RSTA0062
	WRITE (9) (IF (I), I=1, NI), (F (I), I=1, NF)	RSTA0063
	IF (NLIST.EQ.0) GOTO (150	RSTA0064
	00140 I0=1, MXLIST	RSTA0065
	IF (INUSE (I0).EQ.0) GOTO 140	RSTA0066
	CALL READMS (2, IMS, NDMS2, I0)	RSTA0067
	CALL READMS (3, FMS, NDMS3, I0)	RSTA0068
	WRITE (10) I0, (IMS (I), I=1, NDMS2), (FMS (I), I=1, NDMS3)	RSTA0069
140	CONTINUE	RSTA0070
150	IF (2*(NODE/2).NE.NODE) GOTO 160	RSTA0071
	READ (4) (IBV (I), I=1, ND4), (NBV (I), I=1, ND5), (IUPPER (I), I=1, ND5),	RSTA0072
1	(BI (I), I=1, ND4), (BN (I), I=1, ND5), LL1, LL2, LL3,	RSTA0073
2	((B (I, J), I=1, ND4), J=1, NC4)	RSTA0074
	WRITE (10) (IBV (I), I=1, ND4), (NBV (I), I=1, ND5), (IUPPER (I), I=1, ND5),	RSTA0075
1	(BI (I), I=1, ND4), (BN (I), I=1, ND5), LL1, LL2, LL3,	RSTA0076
2	((B (I, J), I=1, ND4), J=1, NC4)	RSTA0077
160	END FILE 9	RSTA0078
	END FILE 10	RSTA0079
	CALL EXIT	RSTA0080
	RETURN	RSTA0081
1000	FORMAT (38) (BEGINNING THE JOB FROM RESTART TAPES.)	RSTA0082
1001	FORMAT (24) (RESTART TAPES PREPARED.)	RSTA0083
1002	FORMAT (12) (****RSTART)	RSTA0084
	END	RSTA0085

EXHIBIT 4 (Continued)

```

SUBROUTINE SIMPLE (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,RHS,C2,C1, SIMP0001
1      BI,BN,U,PJ,BINV,XJ,V,XZ,B,ND1,ND2,ND3,ND4,ND5, SIMP0002
2      ND6,ND7) SIMP0003
C PRIMAL AND DUAL SIMPLEX ALGORITHMS FOR SOLVING THE LP. SIMP0004
COMMON/P1/N,M,ITYFE,NSTRAT,NODRL1,NBVR1,NTITE1,NODRL2,NBVR2, SIMP0005
1      NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT, SIMP0006
2      ITRACE,PSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10) SIMP0007
COMMON/P2/EPS1,EPSIM,BIGN,BEGTH,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2. SIMP0008
1      NM1M3,N1P2,NP1,NSUM,NTC,M10 SIMP0009
COMMON/P3/NOONOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST, SIMP0010
1      NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BL0,NBRNOD,PBRNOD, SIMP0011
2      NBRVAR,NUPDN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BCUNDU, SIMP0012
3      TSIG,IFEAS,IBRV1,IUPDN1,XBRV1,IBRV2,IUPDN2,XBRV2, SIMP0013
4      L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ SIMP0014
DIMENSION IS(ND4),NV(ND6),IBV(ND4),NBV(ND5),IUPPER(ND5) SIMP0015
DIMENSION BI(ND4),BN(ND5),U(ND6),PJ(ND4),XJ(ND4),V(ND6),XZ(ND6), SIMP0016
1      B(ND4,NC4) SIMP0017
IF(ITRACE.GE.1)WRITE(6,1022) SIMP0018
IF(IOUTPT.LE.1)GOTO110 SIMP0019
IF(IALGO.EQ.2)GOTO100 SIMP0020
WRITE(6,1040) SIMP0021
IF(NODE.NE.1)GOTO110 SIMP0022
IF(IPHASE.EQ.1)WRITE(6,1001) SIMP0023
IF(IPHASE.EQ.2)WRITE(6,1002)NPHASE SIMP0024
GOTO110 SIMP0025
100 WRITE(6,1003) SIMP0026
110 ITER=0 SIMP0027
C***** SIMP0028
C INCREMENT THE ITERATION COUNTER. REINVERT THE BASIS MATRIX IF SIMP0029
C NECESSARY. SIMP0030
C***** SIMP0031
120 ITER=ITER+1 SIMP0032
NITER=NITER+1 SIMP0033
NBINV=NBINV+1 SIMP0034
IF(ITER.GT.MXITER)GOTO630 SIMP0035
IF(NBINV.NE.MBINV)GOTO160 SIMP0036
C REINVERT THE BASIS MATRIX EVERY MBINV ITERATIONS. SIMP0037
NBINV=0 SIMP0038
IF(IOUTPT.GE.2)WRITE(6,1004)NITER SIMP0039
IF(IOUTPT.LE.4)GOTO140 SIMP0040
WRITE(6,1005) SIMP0041
DO130I=1,M7 SIMP0042
130 WRITE(6,1006)IBV(I),(B(I,J),J=1,M7),BI(I) SIMP0043
140 CALL BINVRT (NZ,NF,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,U,PJ, SIMP0044
1      BINV,B,ND1,NC2,ND3,N(4,ND5,ND6,ND7) SIMP0045
IF(IOUTPT.LE.4)GOTO160 SIMP0046
WRITE(6,1005) SIMP0047
DO150I=1,M7 SIMP0048
150 WRITE(6,1006)IBV(I),(B(I,J),J=1,M7),BI(I) SIMP0049
160 GOTO(170,300),IALGO SIMP0050
C***** SIMP0051
C PRIMAL ALGORITHM. SIMP0052
C PRICE-OUT THE NON-BASIC VARIABLES. THE ENTERING VARIABLE IS THE FIRST SIMP0053
C ONE ENCOUNTERED HAVING NEGATIVE REDUCED COST. SIMP0054
C***** SIMP0055
170 DO200IPOS=1,L10 SIMP0056
KIND=NBV(IPOS) SIMP0057

```

EXHIBIT 4 (Continued)

DO180I=1,M7	SIMP0058
180 PJ(I)=0.0	SIMP0059
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RMS,C2,C1,PJ,NB1,NB2,NB3,NB4,NB5,	SIMP0060
1 KIND,NZEROS)	SIMP0061
Q1=0.	SIMP0062
DO190J1=1,NZEROS	SIMP0063
J=IS(J1)	SIMP0064
190 Q1=Q1 + B(M10,J)*PJ(J)	SIMP0065
Q1=Q1 + B(M10,MP1)*PJ(MP1)	SIMP0066
IF(IPHASE.EQ.2)Q1=Q1 + B(M10,MP2)*PJ(MP2)	SIMP0067
C Q1 IS THE REDUCED COST FOR VARIABLE J=KIND.	SIMP0068
IF(IUPPER(IPOS).EQ.1)Q1=-Q1	SIMP0069
IF(Q1.LT.EPSIM*1000.)GOTO240	SIMP0070
200 CONTINUE	SIMP0071
C FOR ALL NON-BASIC VARIABLES, THE REDUCED COST IS NON-NEGATIVE.	SIMP0072
IF(IPHASE.NE.2)GOTO640	SIMP0073
IF(NPHASE.EQ.2)GOTO640	SIMP0074
C CHECK IF ANY ARTIFICIALS REMAIN IN THE BASIS.	SIMP0075
INDEX=0	SIMP0076
DO210I=1,M7	SIMP0077
J=IDV(I)	SIMP0078
IF(J.GT.NM1M3 .AND. J.LE.N1)INDEX=1	SIMP0079
210 CONTINUE	SIMP0080
IF(BI(MP2).GE.EPSIM*1000.)GOTO220	SIMP0081
IF(INDEX.EQ.1)GOTO580	SIMP0082
C PHASE 1 TERMINATES. ENTER PHASE 2.	SIMP0083
220 IF(IOUTPUT.GE.2)WRITE(6,1007)NITER	SIMP0084
M10=M10-1	SIMP0085
IF(INDEX.EQ.1)GOTO230	SIMP0086
C THERE ARE NO ARTIFICIALS IN THE BASIS. DELETE THE PHASE 1 OBJECTIVE	SIMP0087
C FUNCTION FROM THE PROGRAM.	SIMP0088
M7=MP1	SIMP0089
IPHASE=1	SIMP0090
NPHASE=0	SIMP0091
NM3M7=N+M3+M7	SIMP0092
GOTO120	SIMP0093
C THERE ARE ARTIFICIALS IN THE BASIS. MAINTAIN THE PHASE 1 OBJECTIVE	SIMP0094
C FUNCTION AS A CONSTRAINT.	SIMP0095
230 NPHASE=2	SIMP0096
GOTO120	SIMP0097
C*****	SIMP0098
C PRIMAL ALGORITHM.	SIMP0099
C SELECT THE LEAVING BASIC VARIABLE.	SIMP0100
C*****	SIMP0101
C COMPUTE THE UPDATED COLUMN, XJ = B-INVERSE * PJ.	SIMP0102
240 DO260I=1,M7	SIMP0103
Q1=0.0	SIMP0104
DO250J1=1,NZEROS	SIMP0105
J=IS(J1)	SIMP0106
250 Q1=Q1 + B(I,J)*PJ(J)	SIMP0107
Q1=Q1 + B(I,MP1)*PJ(MP1)	SIMP0108
IF(IPHASE.EQ.2)Q1=Q1 + B(I,MP2)*PJ(MP2)	SIMP0109
260 XJ(I)=Q1	SIMP0110
C COMPUTE THE MINIMUM OVER XJ(I).GT.0 OF THE BASIC VARIABLE VALUES	SIMP0111
C DIVIDED BY XJ(I).	SIMP0112
XPIV=SIGN	SIMP0113
DO280I=1,M7	SIMP0114

EXHIBIT 4 (Continued)

IF(I.EQ.MP1)GOTO260	SIMP0115
Q1=XJ(I)	SIMP0116
IF(IUPPER(IPOS).EQ.1)Q1=-Q1	SIMP0117
C Q1 IS XJ(I).	SIMP0118
IF(Q1.LT.EPSI)GOTO270	SIMP0119
Q2=9I(I)	SIMP0120
C Q2 IS THE BASIC VARIABLE VALUE.	SIMP0121
IF(Q2/Q1.GE.XPIV)GOTO280	SIMP0122
KPIV=I	SIMP0123
C KPIV INDICATES THE LEAVING VARIABLE.	SIMP0124
XPIV=Q2/Q1	SIMP0125
ITHIA=1	SIMP0126
C ENTERING VARIABLE DOES NOT FORCE ANY VARIABLE TO ITS UPPER BOUND	SIMP0127
C (ITHIA = 1).	SIMP0128
GOTO280	SIMP0129
270 Q1=-Q1	SIMP0130
IF(Q1.LT.EPSI)GOTO280	SIMP0131
I1=IBV(I)	SIMP0132
Q2=U(I1)-BI(I)	SIMP0133
IF(Q2/Q1.GE.XPIV)GOTO280	SIMP0134
KPIV=I	SIMP0135
XPIV=Q2/Q1	SIMP0136
ITHIA=2	SIMP0137
C ENTERING VARIABLE FORCES LEAVING VARIABLE TO ITS UPPER BOUND	SIMP0138
C (ITHIA = 2).	SIMP0139
280 CONTINUE	SIMP0140
IF(U(KIND).GE.XPIV)GOTO290	SIMP0141
ITHIA=3	SIMP0142
C ENTERING VARIABLE ENTERS AT ITS UPPER BOUND (ITHIA = 3).	SIMP0143
XPIV=U(KIND)	SIMP0144
290 IF(XPIV.GE.BIGN)GOTO610	SIMP0145
IBVK=IBV(KPIV)	SIMP0146
GOTO440	SIMP0147
C*****	SIMP0148
C DUAL SIMPLEX ALGORITHM.	SIMP0149
C SELECT THE LEAVING BASIC VARIABLE.	SIMP0150
C*****	SIMP0151
300 IF(UBUBOP.EQ.1)GOTO310	SIMP0152
C TEST THE OBJECTIVE VALUE AGAINST THE BEST UPPER BOUND.	SIMP0153
IF(-BI(MP1)+TSIG.GE.(1.-TOL1)*UNOT)GOTO620	SIMP0154
C COMPUTE THE MINIMUM OF THE RIGHT-HAND-SIDES.	SIMP0155
310 BMIN=BIGN	SIMP0156
KPIV=.	SIMP0157
D0320I=1,M7	SIMP0158
IF(I.EQ.MP1)GOTO320	SIMP0159
IF(BI(I).GE.BMIN)GOTO320	SIMP0160
BMIN=BI(I)	SIMP0161
KPIV=I	SIMP0162
320 CONTINUE	SIMP0163
JPIV=0	SIMP0164
D0330I=1,M7	SIMP0165
IF(I.EQ.MP1)GOTO330	SIMP0166
I1=IBV(I)	SIMP0167
IF(U(I1)-BI(I).GE.BMIN)GOTO330	SIMP0168
BMIN=U(I1)-BI(I)	SIMP0169
JPIV=I	SIMP0170
330 CONTINUE	SIMP0171

EXHIBIT 4 (Continued)

```

C IF THIS MINIMUM IS NONNEGATIVE, WE ARE AT THE OPTIMUM.
IF (BMIN.GE.EPSIM*1000.)GOTO640
ITHIA=1
IF (JPIV.EC.0)GOTO340
ITHIA=2
KPIV=JPIV
340 IBVK=IBV(KPIV)
C*****
C DUAL SIMPLEX ALGORITHM.
C SELECT THE ENTERING VARIABLE.
C*****
C COMPUTE THE MINIMUM OVER XJ(I).LT.0 OF THE DUAL VARIABLE VALUES
C DIVIDED BY -XJ(I).
XPIV=8IGN
KIND=0
D0400JPOS=1,L10
JIND=N8V(JPOS)
D0350I=1,M7
350 PJ(I)=0.0
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,NC2,ND3,ND4,ND5,
1 JIND,NZEROS)
Q1=0.0
D0360J1=1,NZEROS
J=IS(J1)
360 Q1=Q1 + B(KPIV,J)*PJ(J)
Q1=Q1 + B(KPIV,MP1)*PJ(MP1)
IF (IPHASE.EQ.2)Q1=Q1 + B(KPIV,MP2)*PJ(MP2)
IF (ITHIA.EQ.2)GOTO370
IF (IUPPER(JPOS).EQ.0)Q1=-Q1
GOTO380
370 IF (IUPPER(JPOS).EQ.1)Q1=-Q1
380 IF (Q1.LE.EPSI)GOTO400
C Q1 IS -XJ(I).
Q2=0.0
D0390J1=1,NZEROS
J=IS(J1)
390 Q2=Q2 + B(MP1,J)*PJ(J)
Q2=Q2 + B(MP1,MP1)*PJ(MP1)
IF (IPHASE.EQ.2)Q2=Q2 + B(MP1,MP2)*PJ(MP2)
IF (IUPPER(JPOS).EQ.1)Q2=-Q2
C Q2 IS THE DUAL VARIABLE VALUE.
IF (Q2/Q1.GE.XPIV)GOTO400
XPIV=Q2/Q1
KIND=JIND
IPOS=JPOS
400 CONTINUE
C IF THERE IS NO PIVOT ELEMENT, THE PRIMAL PROGRAM IS INFEASIBLE
C (THE DUAL PROGRAM IS UNBOUNDED).
IF (KIND.EQ.0)GOTO580
C COMPUTE THE UPDATED COLUMN, XJ = B-INVERSE * PJ.
D0410I=1,M7
410 PJ(I)=0.0
CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
1 KIND,NZEROS)
D0430I=1,M7
Q1=0.0
D0420J1=1,NZEROS

```

SIMPO172
SIMPO173
SIMPO174
SIMPO175
SIMPO176
SIMPO177
SIMPO178
SIMPO179
SIMPO180
SIMPO181
SIMPO182
SIMPO183
SIMPO184
SIMPO185
SIMPO186
SIMPO187
SIMPO188
SIMPO189
SIMPO190
SIMPO191
SIMPO192
SIMPO193
SIMPO194
SIMPO195
SIMPO196
SIMPO197
SIMPO198
SIMPO199
SIMPO200
SIMPO201
SIMPO202
SIMPO203
SIMPO204
SIMPO205
SIMPO206
SIMPO207
SIMPO208
SIMPO209
SIMPO210
SIMPO211
SIMPO212
SIMPO213
SIMPO214
SIMPO215
SIMPO216
SIMPO217
SIMPO218
SIMPO219
SIMPO220
SIMPO221
SIMPO222
SIMPO223
SIMPO224
SIMPO225
SIMPO226
SIMPO227
SIMPO228

EXHIBIT 4 (Continued)

```

      J=IS(J1)
      Q1=Q1 + B(I,J)*PJ(J)
      Q1=Q1 + B(I,MP1)*PJ(MP1)
      IF(IPHASE.EQ.2)Q1=Q1 + B(I,MP2)*PJ(MP2)
430 XJ(I)=Q1
C*****
C PRIMAL AND DUAL SIMPLEX ALGORITHMS.
C PIVOT THE ENTERING AND LEAVING VARIABLES.
C*****
440 IF (IOUTPT.EQ.5) WRITE(6,1008)NITER,KIND,IBVK,ITHIA
      INDEX=0
      IF (IUPPER(I FOS).EG. 1) GOTO480
      GOTO(510,470,450), ITHIA
450 IUPPER(IPCS)=1
      JN(IPOS)=L(KIND)
      D0460I=1,M7
460 BI(I)=BI(I) - U(KIND)*XJ(I)
      GOTO120
470 BI(KPIV)=BI(KPIV) - U(IBVK)
      IUPPER(IPCS)=1
      BN(IPOS)=U(IBVK)
      GOTO510
480 IUPPER(IPCS)=0
      BN(IPOS)=0.0
C CHANGE THE RIGHT-HAND-SIDE AFTER THE PIVOT.
      INDEX=1
      GOTO(510,470,490), ITHIA
490 D0500I=1,M7
500 BI(I)=BI(I) + U(KIND)*XJ(I)
      GOTO120
510 T=1./XJ(KPIV)
      BI(KPIV)=BI(KPIV)*T
      D0520K=1,M7
520 B(KPIV,K)=B(KPIV,K)*T
      D0540J=1,M7
      IF (J.EQ.KPIV)GOTO540
      T=XJ(J)
      IF (ABS(T).LE.EPSI)GOTO540
      T1=BI(J) - T*BI(KPIV)
      IF (ABS(T1).LE.EPSI)T1=0.0
      BI(J)=T1
      D0530K=1,M7
      T1=B(J,K) - T*B(KPIV,K)
      IF (ABS(T1).LE.EPSI)T1=0.0
530 B(J,K)=T1
540 CONTINUE
      IF (INDEX.EQ.0)GOTO550
      BI(KPIV)=BI(KPIV) + U(KIND)
C UPDATE THE NBV AND IBV ARRAYS.
550 I=IBV(KPIV)
      IBV(KPIV)=NBV(IPOS)
      IF (I.GT.NM1M3 .AND. I.LE.N1)GOTO560
      IF (IALGO.EQ.2 .AND. I.EQ.N1P2)GOTO560
      NBV(IPOS)=I
      GOTO120
560 NBV(IPOS)=NBV(L10)
      IUPPER(IPCS)=IUPPER(L10)

```

SIMP0229
 SIMP0230
 SIMP0231
 SIMP0232
 SIMP0233
 SIMP0234
 SIMP0235
 SIMP0236
 SIMP0237
 SIMP0238
 SIMP0239
 SIMP0240
 SIMP0241
 SIMP0242
 SIMP0243
 SIMP0244
 SIMP0245
 SIMP0246
 SIMP0247
 SIMP0248
 SIMP0249
 SIMP0250
 SIMP0251
 SIMP0252
 SIMP0253
 SIMP0254
 SIMP0255
 SIMP0256
 SIMP0257
 SIMP0258
 SIMP0259
 SIMP0260
 SIMP0261
 SIMP0262
 SIMP0263
 SIMP0264
 SIMP0265
 SIMP0266
 SIMP0267
 SIMP0268
 SIMP0269
 SIMP0270
 SIMP0271
 SIMP0272
 SIMP0273
 SIMP0274
 SIMP0275
 SIMP0276
 SIMP0277
 SIMP0278
 SIMP0279
 SIMP0280
 SIMP0281
 SIMP0282
 SIMP0283
 SIMP0284
 SIMP0285

EXHIBIT 4 (Continued)

BN(IPOS)=BN(L10)	SIMP0286
L10=L10-1	SIMP0287
IF(NPHASE.EQ.1)GOTO120	SIMP0288
DO570I=1,M7	SIMP0289
J=IBV(I)	SIMP0290
IF(J.GT.NM1M3 .AND. J.LE.N1)GOTO120	SIMP0291
570 CONTINUE	SIMP0292
C THERE ARE NO ARTIFICIALS IN THE BASIS. DELETE THE PHASE 1 OBJECTIVE	SIMP0293
C FUNCTION FROM THE PROGRAM.	SIMP0294
M7=MP1	SIMP0295
IPHASE=1	SIMP0296
NPHASE=0	SIMP0297
NM3M7=N+M3+M7	SIMP0298
GOTO120	SIMP0299
C*****	SIMP0300
C FINAL OUTPUT.	SIMP0301
C*****	SIMP0302
580 IF(IOUTPT.EQ.0)GOTO600	SIMP0303
WRITE(6,1009)	SIMP0304
IF(IOUTPT.LE.2)GOTO600	SIMP0305
IF(IALGO.EQ.1)GOTO590	SIMP0306
WRITE(6,1010)IBVK,MIN	SIMP0307
590 WRITE(6,1011)(IBV(I),I=1,M7)	SIMP0308
WRITE(6,1012)(BI(I),I=1,M7)	SIMP0309
600 IE0J=1	SIMP0310
GOTO640	SIMP0311
610 WRITE(6,1013)	SIMP0312
WRITE(6,1014)KIND	SIMP0313
WRITE(6,1011)(IBV(I),I=1,M7)	SIMP0314
WRITE(6,1015)(XJ(I),I=1,M7)	SIMP0315
WRITE(6,1012)(BI(I),I=1,M7)	SIMP0316
IE0J=2	SIMP0317
GOTO640	SIMP0318
620 IF(IOUTPT.NE.C)WRITE(6,1016)	SIMP0319
IE0J=3	SIMP0320
GOTO640	SIMP0321
630 WRITE(6,1017)	SIMP0322
IE0J=4	SIMP0323
640 DO650I=1,N1P2	SIMP0324
650 XZ(I)=0.0	SIMP0325
DO660I=1,NM3M7	SIMP0326
NV(I)=0	SIMP0327
660 V(I)=0.0	SIMP0328
DO680K=1,M7	SIMP0329
I=IBV(K)	SIMP0330
NV(I)=I	SIMP0331
IF(K.EQ.MP1)GOTO670	SIMP0332
V(I)=BI(K)	SIMP0333
XZ(I)=BI(K)	SIMP0334
GOTO680	SIMP0335
670 V(I)=-BI(K)	SIMP0336
XZ(I)=-BI(K)	SIMP0337
680 CONTINUE	SIMP0338
Z=-BI(MP1)	SIMP0339
IF(IOUTPT.LE.3)GOTO710	SIMP0340
DO700K=1,M7	SIMP0341
DO690I=K,NM3M7	SIMP0342

EXHIBIT 4 (Continued)

IF (NV(I).EQ.0)GOTO690	SIMP0343
IF (I.EQ.K)GOTO700	SIMP0344
NV(K)=NV(I)	SIMP0345
V(K)=V(I)	SIMP0346
NV(I)=0	SIMP0347
GOTO700	SIMP0348
690 CONTINUE	SIMP0349
700 CONTINUE	SIMP0350
WRITE(6,1018)	SIMP0351
WRITE(6,1019)(NV(K),V(K),K=1,M7)	SIMP0352
710 CONTINUE	SIMP0353
DO720I=1,NM3M7	SIMP0354
NV(I)=0	SIMP0355
720 V(I)=0.0	SIMP0356
KK=0	SIMP0357
DO730K=1,L10	SIMP0358
IF (IUPPER(K).EQ.0)GOTO730	SIMP0359
KK=KK+1	SIMP0360
I=N3V(K)	SIMP0361
NV(I)=I	SIMP0362
V(I)=BN(K)	SIMP0363
XZ(I)=BN(K)	SIMP0364
730 CONTINUE	SIMP0365
IF (IOUTPT.LE.3)GOTO780	SIMP0366
IF (KK.EQ.0)GOTO760	SIMP0367
DO750K=1,KK	SIMP0368
DO740I=K,NM3M7	SIMP0369
IF (NV(I).EQ.0)GOTO740	SIMP0370
IF (I.EQ.K)GOTO750	SIMP0371
NV(K)=NV(I)	SIMP0372
V(K)=V(I)	SIMP0373
NV(I)=0	SIMP0374
GOTO750	SIMP0375
740 CONTINUE	SIMP0376
750 CONTINUE	SIMP0377
WRITE(6,1020)	SIMP0378
WRITE(6,1019)(NV(K),V(K),K=1,KK)	SIMP0379
760 IF (IOUTPT.LE.4)GOTO780	SIMP0380
WRITE(6,1005)	SIMP0381
DO770I=1,M7	SIMP0382
770 WRITE(6,1006)IBV(I),(J(I,J),J=1,M7),BI(I)	SIMP0383
780 IF (IOUTPT.GE.2)WRITE(6,1021)ITER,NITER	SIMP0384
ITRTOT=ITRTOT + ITER	SIMP0385
IF (NITER.GT.ITRMAX)ITRMAX=NITER	SIMP0386
RETURN	SIMP0387
1000 FORMAT(17HQPRIMAL ALGORITHM)	SIMP0388
1001 FORMAT(17H ONE PHASE METHOD)	SIMP0389
1002 FORMAT(47H TWO PHASE METHOD - BEGIN COMPUTATIONS IN PHASE,I2)	SIMP0390
1003 FORMAT(23HQQUAL SIMPLEX ALGORITHM)	SIMP0391
1004 FORMAT(32HQBASIS REINVERTED ON ITERATION =,I5)	SIMP0392
1005 FORMAT(48HQBASIS VARIABLES/BASIS INVERSE/RIGHT-HAND-SIDE =)	SIMP0393
1006 FORMAT(11H0,I5,8E15.6/(6X,8E15.6))	SIMP0394
1007 FORMAT(27H ENTER PHASE 2 ON ITERATION,I5)	SIMP0395
1008 FORMAT(1X,I5,21H. ENTERING VARIABLE =,I5,20H, LEAVING VARIABLE =,	SIMP0396
1 I5,16H, THETA = THETA(,I1,3H))	SIMP0397
1009 FORMAT(34H0THE FRIMAL PROGRAM IS INFEASIBLE.)	SIMP0398
1010 FORMAT(20H0LEAVING VARIABLE =,I5/	SIMP0399

EXHIBIT 4 (Continued)

1	20H0RIGHT-HAND-SIDE	=,E15.6)	SIMP0400
1011	FORMAT(20H0BASIC VARIABLES	=,16I5/(20X,16I5))	SIMP0401
1012	FORMAT(20H0RIGHT-HAND-SIDE	=,6E15.6/(20X,6E15.6))	SIMP0402
1013	FORMAT(43H0THE PRIMAL PROGRAM HAS UNBOUNDED SOLUTION.)		SIMP0403
1014	FORMAT(20H0ENTERING VARIABLE	=,I5)	SIMP0404
1015	FORMAT(20H0UPDATED COLUMN	=,6E15.6/(20X,6E15.6))	SIMP0405
1016	FORMAT(45H0THE DUAL VALUE EXCEEDS THE BEST UPPER BOUND.)		SIMP0406
1017	FORMAT(55H0THE MAXIMUM NUMBER OF LP ITERATIONS HAS BEEN EXCEEDED.)		SIMP0407
1018	FORMAT(16H0BASIC VARIABLES)		SIMP0408
1019	FORMAT(17H0 VARIABLE/VALUE	=,5(I5,E15.6)/(17X,5(I5,E15.6)))	SIMP0409
1020	FORMAT(35H0NON-BASIC VARIABLES AT UPPER BOUND)		SIMP0410
1021	FORMAT(43H0NUMBER OF LP ITERATIONS THIS COMPUTATION	=,I5,	SIMP0411
1	14H, CUMULATIVE	=,I5)	SIMP0412
1022	FORMAT(12H *****SIMPLE)		SIMP0413
	END		SIMP0414

EXHIBIT 4 (Continued)

```

SUBROUTINE SLOPES (NZ, NP, IR, IA, IS, IBV, NBV, IUPPER, TC, RHS, C2, C1, PJ, SLOP0001
1 XJ, S0, S1, B, ND1, ND2, ND3, ND4, ND5, ND8) SLOP0002
C DETERMINE THE LEFT AND RIGHT SLOPES ASSOCIATED WITH THE OPTIMAL SLOP0003
C SOLUTION VALUE AS A FUNCTION OF A PARAMETER. SLOP0004
COMMON/P1/N, M, ITYPE, NSTRAT, NOCRL1, NBVRL1, NTITE1, NODRL2, NBVRL2, SLOP0005
1 NTITE2, MXLIST, LISTOP, ITAPE, IFB, MXITER, MBINV, IOUPT, SLOP0006
2 ITRACE, MSTART, TIME1, TCL1, TOL2, PCBUB, ALPHA(10) SLOP0007
COMMON/P2/EPSI, EPSIM, BIGN, BEGTH, M1, M2, M3, M4, N1, MP1, MP2, NM3, NM1M2, SLOP0008
1 NM1M3, N1P2, NP1, NSUM, NTC, M10 SLOP0009
COMMON/P3/NODNUT, LNOT, IUBOP, LPHASE, NODRUL, NBVRUL, NTIGHT, NLIST, SLOP0010
1 NLISTS, NFEAS, LSTMX, ITRCT, ITRMAX, BLB, NBRNOD, PBRNOD, SLOP0011
2 NBRVAR, NUPDOWN, XBRNOD, TBFNOD, NODE, LNODE, Z, BOUNDL, BCUNDU, SLOP0012
3 TSIG, IFEAS, IBKVR1, IUPDN1, XBRVR1, IBKVR2, IUPDN2, XBRVR2, SLOP0013
4 L10, NITER, NAINV, M7, IPHASE, NPHASE, NM3M7, IALGO, IEOJ SLOP0014
DIMENSION IS (ND4), IBV (ND4), NBV (ND5), IUPPER (ND5) SLOP0015
DIMENSION PJ (ND4), XJ (ND4), S0 (ND8), S1 (ND8), B (ND4, ND4) SLOP0016
IF (ITRACE.GE.1) WRITE (6, 1002) SLOP0017
DO 100 I=1, M7 SLOP0018
IF (I.EQ.MP1) GOTO 100 SLOP0019
SG (I)=-BIGN SLOP0020
S1 (I)=BIGN SLOP0021
100 CONTINUE SLOP0022
DO 150 IPOS=1, L10 SLOP0023
KIND=NBV (IPOS) SLOP0024
DO 110 I=1, M7 SLOP0025
110 PJ (I)=0.0 SLOP0026
CALL GETCOL (NZ, NP, IR, IA, IS, TC, RHS, C2, C1, PJ, ND1, ND2, ND3, ND4, ND5, SLOP0027
1 KIND, NZEROS) SLOP0028
DO 130 I=1, M7 SLOP0029
Q1=0.0 SLOP0030
DO 120 J=1, NZEROS SLOP0031
J=IS (J1) SLOP0032
120 Q1=Q1 + B (I, J)*PJ (J) SLOP0033
Q1=Q1 + B (I, MP1)*PJ (MP1) SLOP0034
IF (IPHASE.EQ.2) Q1=Q1 + B (I, MP2)*PJ (MP2) SLOP0035
IF (IUPPER (IPOS).EQ.1) Q1=-Q1 SLOP0036
130 XJ (I)=Q1 SLOP0037
IF (XJ (MP1).LE.EPSI) XJ (MP1)=0.0 SLOP0038
DO 150 I=1, M7 SLOP0039
IF (I.EQ.MP1) GOTO 150 SLOP0040
IF (ABS (XJ (I)).LE.EPSI) GOTC 150 SLOP0041
X0=XJ (MP1)/(-XJ (I)) SLOP0042
IF (XJ (I).LT.EPSIM) GOTO 140 SLOP0043
C S0 (I) IS THE MAXIMUM OVER X (I, J).GT.0 OF THE REDUCED COST DIVIDED SLOP0044
C BY -X (I, J). SLOP0045
IF (X0.LE.S0 (I)) GOTO 150 SLOP0046
S0 (I)=X0 SLOP0047
GOTO 150 SLOP0048
C S1 (I) IS THE MINIMUM OVER X (I, J).LT.0 OF THE REDUCED COST DIVIDED SLOP0049
C BY -X (I, J). SLOP0050
140 IF (X0.GE.S1 (I)) GOTO 150 SLOP0051
S1 (I)=X0 SLOP0052
150 CONTINUE SLOP0053
IF (IOUPT.LE.2) RETURN SLOP0054
WRITE (6, 1000) SLOP0055
DO 160 I=1, M7 SLOP0056
IF (I.EQ.MP1) GOTO 160 SLOP0057

```

EXHIBIT 4 (Continued)

```
WRITE (6,1001)IBV(I),S(I),SI(I)
160 CONTINUE
RETURN
1000 FORMAT(1H0,2X,5HBASIC,11X,4HLEFT,12X,5HRIGHT/
1      1X,8HVARIABLE,9X,5HSLOPE,12X,5HSLOPE//)
1001 FORMAT(3X,I5,3X,E15.6,2X,E15.6)
1002 FORMAT(12H *****SLOPES)
END
SLOP0058
SLOP0059
SLOP0060
SLOP0061
SLOP0062
SLOP0063
SLOP0064
SLOP0065
```

EXHIBIT 4 (Continued)

<pre> SUBROUTINE TIMEC C PRINT THE ELAPSED TIME SINCE THE BEGINNING OF THIS JOB. COMMON/P1/N,M,I,TYPE,NSTRAT,NOORL1,NBVRL1,NTITE1,NOORL2,NBVRL2, 1 NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT, 2 ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(16) COMMON/P2/EPSI,EPSTM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2, 1 NM1M3,N1P2,NP1,NSUM,NTC,M10 IF(ITRACE.GE.1)WRITE(6,10.1) CALL SECOND(X) X=X-BEGTM WRITE(6,1000)X RETURN 1000 FORMAT(7H)TIME =,F9.3,8H SECONDS) 1001 FORMAT(11H *****TIMEC) END </pre>	<pre> TIME0001 TIME0002 TIME0003 TIME0004 TIME0005 TIME0006 TIME0007 TIME0008 TIME0009 TIME0010 TIME0011 TIME0012 TIME0013 TIME0014 TIME0015 </pre>
---	---

APPENDIX G

REFERENCES

1. Robert S. Garfinkel and George L. Nemhauser, Integer Programming, John Wiley and Sons, Inc., New York, 1972.
2. Frederick S. Hillier and Gerald J. Lieberman, Introduction to Operations Research, Holden-Day, Inc., San Francisco, 1967.
3. Leon S. Lasdon, Optimization Theory for Large Systems, The MacMillan Company, London, 1970.

APPENDIX H
DISTRIBUTION

Defense Documentation Center (12)
Cameron Station
Alexandria, Virginia 22314

Library of Congress (4)
Attn: Gift and Exchange Division
Washington, D. C. 20540

Local:
K30 (25)
K70 (1)
E41 (2)
X210 (1)
X2101(GIDEP) (5)