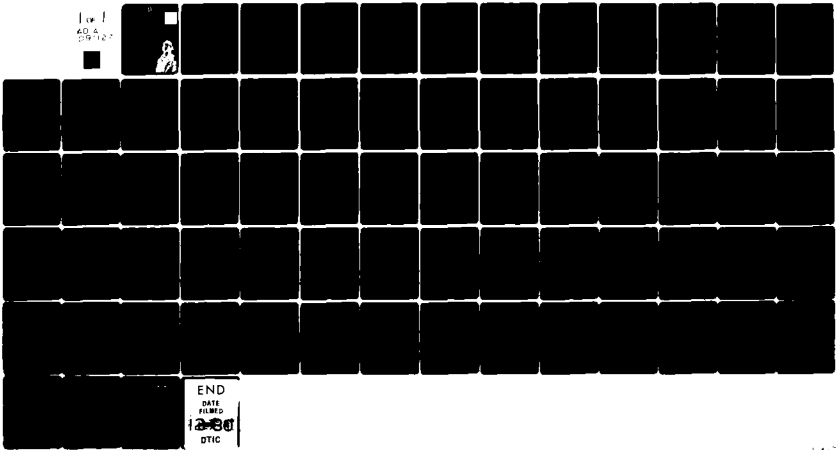
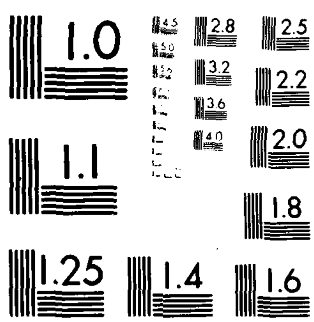
AD-A091 127 GEORGE WASHINGTON UNIV WASHINGTON D C PROGRAM IN LOG--ETC F/6 12/1
PROGRAM DESCRIPTION AND USER'S GUIDE FOR ZIPCAP -- A ZERO-ONE I--ETC(U)
JUL 80 K L CHHABRA, R M SOLAND N00014-75-C-0729
UNCLASSIFIED SERIAL T-423 NL

1 of 1
AD A
091127





MICROCOPY RESOLUTION TEST CHART

THE
GEORGE
WASHINGTON
UNIVERSITY

STUDENTS FACULTY STUDY R
ESEARCH DEVELOPMENT FUT
URE CAREER CREATIVITY CC
MMUNITY LEADERSHIP TECH
NOLOGY FRONTI ESIGN
ENGINEERING APP ENC
GEORGE WASHIN NIV

SCHOOL OF ENGINEERING
AND APPLIED SCIENCE

PROGRAM DESCRIPTION AND USER'S GUIDE FOR ZIPCAP -- A ZERO-ONE INTEGER PROGRAM TO SOLVE MULTIACTIVITY MULTIFACILITY CAPACITY-CONSTRAINED ASSIGNMENT PROBLEMS

by

Krishan L. Chhabra
Richard M. Soland

Serial T-423
1 July 1980

The George Washington University
School of Engineering and Applied Science
Institute for Management Science and Engineering

Program in Logistics
~~Contract~~ N00014-75-C-0729
Project NR 347 020
Office of Naval Research

405337

NONE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER T-423	2. GOVT ACCESSION NO. AD-A091	3. RECIPIENT'S CATALOG NUMBER 127
4. TITLE (and Subtitle) PROGRAM DESCRIPTION AND USER'S GUIDE FOR ZIPCAP-- A ZERO-ONE INTEGER PROGRAM TO SOLVE MULTIACTIVITY MULTIFACILITY CAPACITY-CONSTRAINED ASSIGNMENT PROBLEMS		5. TYPE OF REPORT & PERIOD COVERED SCIENTIFIC
		6. PERFORMING ORG. REPORT NUMBER T-423
7. AUTHOR(s) KRISHAN L. CHHABRA RICHARD M. SOLAND		8. CONTRACT OR GRANT NUMBER(s) N00014-75-C-0729
9. PERFORMING ORGANIZATION NAME AND ADDRESS THE GEORGE WASHINGTON UNIVERSITY PROGRAM IN LOGISTICS WASHINGTON, DC 20037		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS OFFICE OF NAVAL RESEARCH CODE 434 ARLINGTON, VIRGINIA 22217		12. REPORT DATE 1 July 1980
		13. NUMBER OF PAGES 66
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) NONE
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC SALE AND RELEASE: DISTRIBUTION UNLIMITED.

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

18. SUPPLEMENTARY NOTES

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

INTEGER PROGRAMMING
MULTIACTIVITY MULTIFACILITY ASSIGNMENT PROBLEM

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

ZIPCAP is a zero-one integer program developed to solve multiactivity multifacility capacity-constrained assignment problems. The objective in such problems is an assignment that minimizes the sum of the variable costs and fixed costs. This document describes ZIPCAP and provides guidelines for its use. Illustrative areas of application, program options, and test problems are included.

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

NONE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

THE GEORGE WASHINGTON UNIVERSITY
School of Engineering and Applied Science
Institute for Management Science and Engineering

Program in Logistics

Abstract
of
Serial T-423
1 July 1980

PROGRAM DESCRIPTION AND USER'S GUIDE FOR ZIPCAP -- A ZERO-ONE INTEGER PROGRAM TO SOLVE MULTIACTIVITY MULTIFACILITY CAPACITY-CONSTRAINED ASSIGNMENT PROBLEMS

by

Krishan L. Chhabra
Richard M. Soland

ZIPCAP is a zero-one integer program developed to solve multiactivity multifacility capacity-constrained assignment problems. The objective in such problems is an assignment that minimizes the sum of variable costs and fixed costs. This document describes ZIPCAP and provides guidelines for its use. Illustrative areas of application, program options, and test problems are included.

Research Supported by

Contract N00014-75-C-0729
Project NR 347 020
Office of Naval Research

TABLE OF CONTENTS

THE GEORGE WASHINGTON UNIVERSITY
School of Engineering and Applied Science
Institute for Management Science and Engineering

Program in Logistics

PROGRAM DESCRIPTION AND USER'S GUIDE FOR ZIPCAP -- A ZERO-ONE INTEGER PROGRAM TO SOLVE MULTIACTIVITY MULTIFACILITY CAPACITY-CONSTRAINED ASSIGNMENT PROBLEMS

by

Krishan L. Chhabra
Richard M. Soland

1. Introduction

Multiactivity multifacility assignment problems are important because of their practical applications. The objective of such a problem is to minimize the total cost of the system, which includes both variable costs and fixed costs. A problem which has been of recent interest takes into account capacity constraints as well [Gross, Pinkus, and Soland (1979)]. A solution algorithm and a computer program implementing it have been developed for this kind of problem, i.e., for multiactivity multifacility 0-1 assignment problems with capacity constraints.

The development of the solution algorithm and computational test results using the computer program are described in a separate report titled "Solving a Multiactivity Multifacility Capacity-constrained 0-1 Assignment Problem."

This document, on the other hand, provides a description of the computer program and instructions for its use. It is a FORTRAN program and is named ZIPCAP, an acronym for Zero-one Integer Program to solve multi-activity multifacility Capacity-constrained Assignment Problems.

Section 2 of this document provides a mathematical formulation of the basic problem and examples of application areas. Section 3 includes the computational steps and a program description, whereas Section 4 provides user information including program options and input data specifications. Test problems and output illustrations are covered in Section 5.

2. Problem Formulation

2.1 Basic Problem

The basic problem, called problem (P), is to find x_{ij} and y_k values that:

$$
(P)\left\{
\begin{array}{lll}
\text{Minimize} & \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}\, x_{ij} + \sum_{k=1}^{p} b_k y_k & (1) \\
\text{subject to} & \displaystyle\sum_{i=1}^{m} x_{ij} = 1 \quad j=1, \ldots, n & (2) \\
 & \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} d_{ijk}\, x_{ij} \leq s_k y_k \quad k=1, \ldots, p & (3) \\
 & x_{ij}\,,\ y_k = 0 \text{ or } 1 \quad \text{for all } i,\ j,\ k & (4)
\end{array}
\right.
$$

where m, n, and p represent the number of designs, activities, and facilities respectively; and i, j, k are the corresponding indices.

The parameters are interpreted as follows:

a_{ij} = variable cost of activity j using design i ,

b_k = fixed cost of facility k ,

s_k = capacity available at facility k , and

d_{ijk} = capacity required at facility k for activity j when activity j uses design i .

The decision variables are:

x_{ij} = 1 if activity j uses design i ,
= 0 otherwise.

y_k = 1 if facility k is used ,
= 0 otherwise.

Simply stated, in problem (P) one seeks to minimize the sum of variable and fixed costs subject to constraint set (2) that each activity be assigned a single design and subject to constraint set (3) that the capacity constraints imposed by the facility capacities are satisfied.

Problem (P) has $mn+p$ 0-1 variables and $n+p$ constraints.

2.2 Key Elements

The key elements in problem (P) are activities, facilities, designs, variable costs, and fixed costs. In general, these can be defined as follows.

- Activity -- a type of component, product or item under consideration.

- Facility -- an installation or location where activities can be served or serviced.

- Design -- a meaningful configuration of facilities along with a meaningful strategy for using them. For example, if there are four facilities 1, 2, 3 and 4, a design may include facilities 1, 2 and 4, whereas another design may include facility 2 only.

- Variable Cost -- the cost associated with a particular activity being assigned to a particular design.

- Fixed Cost -- the cost associated with a facility (such as part of its purchase, operation and maintenance cost) that is independent of the activities served at that facility.

2.3 Areas of Application

The computer program ZIPCAP is designed to solve a problem that can be formulated as problem (P). The formulation applies to existing as well as to proposed facilities. In other words, it is useful for a situation where the decision may be to delete some of the existing facilities, as well as for a situation involving a choice among proposed facilities (or locations).

Table 1 includes examples of areas where this formulation can be applied. Within each application area, activities and facilities are identified. The implications of designs, variable costs and fixed costs are apparent.

3. Computational Procedure and Program Description

The procedure to solve problem (P) basically involves branch and bound [Geoffrion and Marsten (1972)], and Lagrangean relaxation

Table 1

EXAMPLES OF APPLICATION AREAS

Area of Application	Activities	Facilities
1. Design of multi-echelon inventory systems*	Items to be stocked	Warehouses (Comprising different levels or echelons, e.g., central, regional, and local warehouses)
2. Assignment of repairable components**	Major components of a unit, e.g., components of an aircraft, a ship, a piece of machinery	Repair depots
3. Design of training programs	Training program categories or occupational classifications	Training schools
4. Location of facilities***	Types of services, e.g., health-care services	Buildings or installations, e.g., health-care centers

*Gross, Pinkus, and Soland (1979)

**Gross and Pinkus (1979)

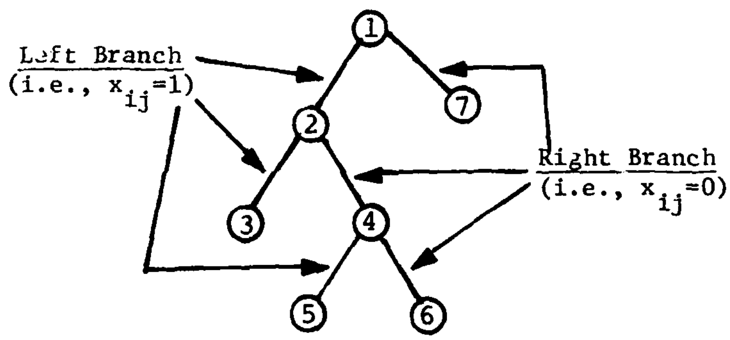
***Pinkus, Gross, and Soland (1973)

[Geoffrion (1974)]. In addition certain heuristics are used for branching, and to exclude infeasible assignments.

Branching is done on the x_{ij} variables by setting them equal to 1 or 0 . The branching commences by setting a variable equal to 1 and moving to the left-branch node. When backtracking, the corresponding variable is set equal to 0 and we move to the right-branch node (if the right-branch has not yet been explored). Figure 1 illustrates a branching tree.

Figure 1

Tree Branching Illustration



Node (1) is the root node and also parent node for nodes (2) and (7)

Node (2) is parent node for nodes (3) and (4), etc.

The x_{ij} variables which have been set equal to 1 or 0 at any given node are termed <u>fixed variables</u>, and the remaining ones are termed <u>free variables</u>.

3.1 Computational Steps

Figure 2 presents a simplified flow diagram, showing the

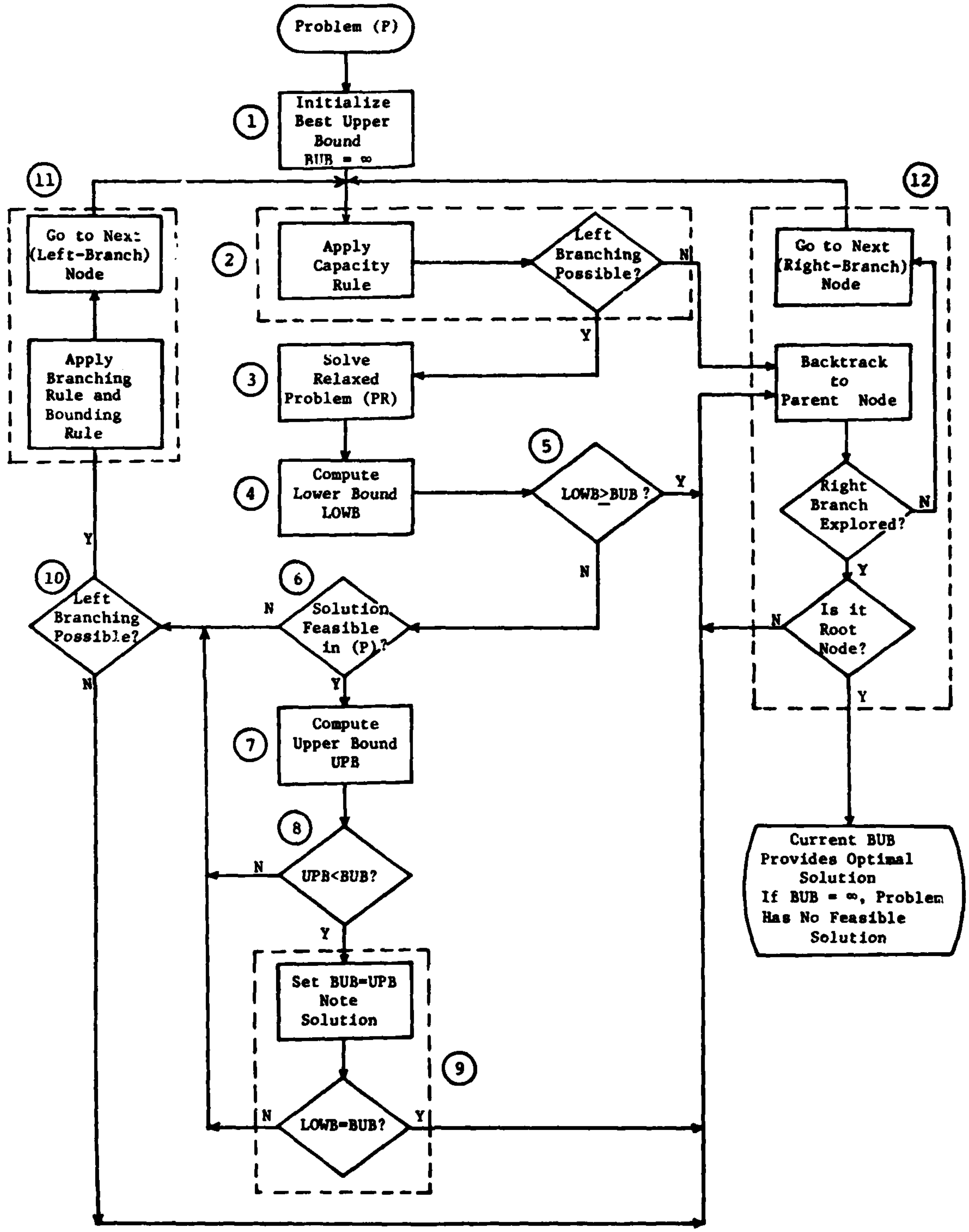


Figure 2

Simplified Flow Diagram Showing Computational Steps computational steps of the computer program. Following is a description of these steps.

Step 1. Initialize by setting best upper bound (BUB) equal to a large value and the node number to 1.

Step 2. Apply the capacity rule to exclude infeasible assignments prior to solving the relaxed problem.

Exclude a free x_{ij} if, for any facility k and activity j,

$$\left(d_{ijk} - \min_j d_{ijk}\right) > \left(s_k - \Sigma_j \min_j d_{ijk}\right) .$$

If an x_{ij} is already fixed as 1, the corresponding d_{ijk} replaces $\min_j d_{ijk}$.

If $\Sigma_j \min_j d_{ijk} > s_k$, then backtrack, i.e., go to step 12.

Step 3. Solve the relaxed problem (PR), i.e.,

$$(\text{PR}) \begin{cases} \text{Minimize} & \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} \; x_{ij} \\ \text{subject to} & \sum_{i=1}^{m} x_{ij} = 1 \quad j=1, \cdots, n, \\ & x_{ij} = 0 \text{ or } 1 \text{ for all } i, j, \end{cases}$$

where $c_{ij} = a_{ij} + \sum_{k=1}^{p} b_k \left(1 - \delta_k\right)\left(d_{ijk}/s_k\right)$,

$$\delta_k = 1 \text{ if } \sum_{\substack{i \\ \text{such that} \\ x_{ij} \text{ is fixed}}} \sum_j x_{ij} \; e_{ik} > C ,$$

$$= 0 \text{ otherwise, and}$$

$$e_{ik} = 1 \text{ if design } i \text{ uses facility } k\,,$$
$$= 0 \text{ otherwise.}$$

<u>Step 4</u>. Lower bound (LOWB) is the sum of the optimal solution value of problem (PR) and the fixed cost FC. FC is the cost of the facilities forced into the solution because of the x_{ij} variables fixed as 1:

$$FC = \sum_{k=1}^{p} \delta_k b_k$$

<u>Step 5</u>. Compare lower bound with best upper bound. If LOWB $\geq$ BUB , go to step 12.

<u>Step 6</u>. Check if the solution of problem (PR) obtained in step 3 satisfies the capacity constraints of problem (P) , i.e.,

$$\sum_{i=1}^{m} \sum_{j=1}^{n} d_{ijk} \quad x_{ij} \leq s_k y_k \qquad k=1, \ldots, p,$$

where $y_k = 1$ if $\Sigma_i \; \Sigma_j \; d_{ijk} \; x_{ij} > 0$,

$= 0$ otherwise

If the capacity constraints are not satisfied, go to step 10.

<u>Step 7</u>. Compute upper bound (UPB) , i.e.,

$$UPB = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} \; x_{ij} + \sum_{k=1}^{p} b_k y_k$$

Step 8. Compare upper bound with best upper bound. If UPB $\geq$ BUB, go to step 10.

Step 9. Retain UPB as BUB , i.e., set BUB = UPB and note the corresponding solution comprising x_{ij} and y_k variables.

If LOWB equals BUB (same as UPB in this case), go to step 12.

Step 10 If an x_{ij} variable has been fixed as 1 for each of the n activities, then go to step 12.

Step 11. Select an x_{ij} variable for branching to the left node. The choice of the branching variable depends on the c_{ij} values and is such that the corresponding x_{ij} , if perturbed, has the maximum impact on the optimal value of problem (PR). That is, for each j , obtain $D_j = c_{2j} - c_{1j}$ where c_{2j} is the second smallest permissible and c_{1j} is the smallest permissible c_{ij} value in column j .

A c_{ij} value is considered permissible if it does not correspond to an activity j having x_{ij} variable fixed as 1 , or to an x_{ij} variable fixed as 0 . The x_{ij} variables can be fixed as 1 or 0 because of the branching rule, backtracking, and/or use of the capacity rule. Further, x_{ij} may be fixed as 0 because of the bounding rule, i.e., if $\left(c_{ij} - c_{1j}\right) > \left(\text{BUB} - \text{LOWB}\right)$.
The x_{ij} variable selected has the maximum value of D_j and has $c_{ij} = c_{1j}$.

Set the selected x_{ij} variable to 1 . Increase the node number by one, and go to step 2.

Step 12. Backtracking continues until a node is reached for which the right branch has not been explored. Set the corresponding x_{ij} variable to 0 . Increase the node number by one, and go to step 2.

If no such node is found, it implies that the root node has been reached, and the procedure terminates.

The current BUB and the solution obtained in step 9 provide the optimal solution and its value. If BUB happens to be the initial value, the problem does not have a feasible solution, i.e., the capacity constraints cannot be satisfied.

3.2 Computer Program

The computer program ZIPCAP is written in FORTRAN IV. It is based on the flow diagram and the computational steps described in the preceding paragraphs.

Appendix A provides a listing of the program. Extensive use of comment cards provides details within each computational step.

A dictionary of the symbolic names used in the program listing is provided in Appendix B.

The program has been developed and tested on an IBM 3031 at The George Washington University.* The only internally supplied subroutine used by the program is TIMET from the PIJ Library. This subroutine provides the elapsed time for program execution and excludes the time to read and write the input and output.

*IBM 3031 CPU with 3 million Bytes Real Memory. Operating System OS/VS1 Release 6.7. Various compilers including FORTRAN levels G 1 to 6 and H 1 to 6.

The program comprising about 480 lines is currently dimensioned for a problem size of 35 designs (m) , 35 activities (n) and 30 facilities (p) . The user capacity required to execute this program is 346 K bytes. The functional relationship between the dimensions of the arrays and the storage requirement is given by

$$f\ (m\ ,\ n\ ,\ p) = 4\left[(p+4)mn + (m+5)p + 9n\right] \text{ bytes.}$$

Thus the program size to execute a problem has two components: one, due to the program itself, comprises 173 K bytes, and the other, dependent on the dimensions of the arrays, is given by the above functional relationship.

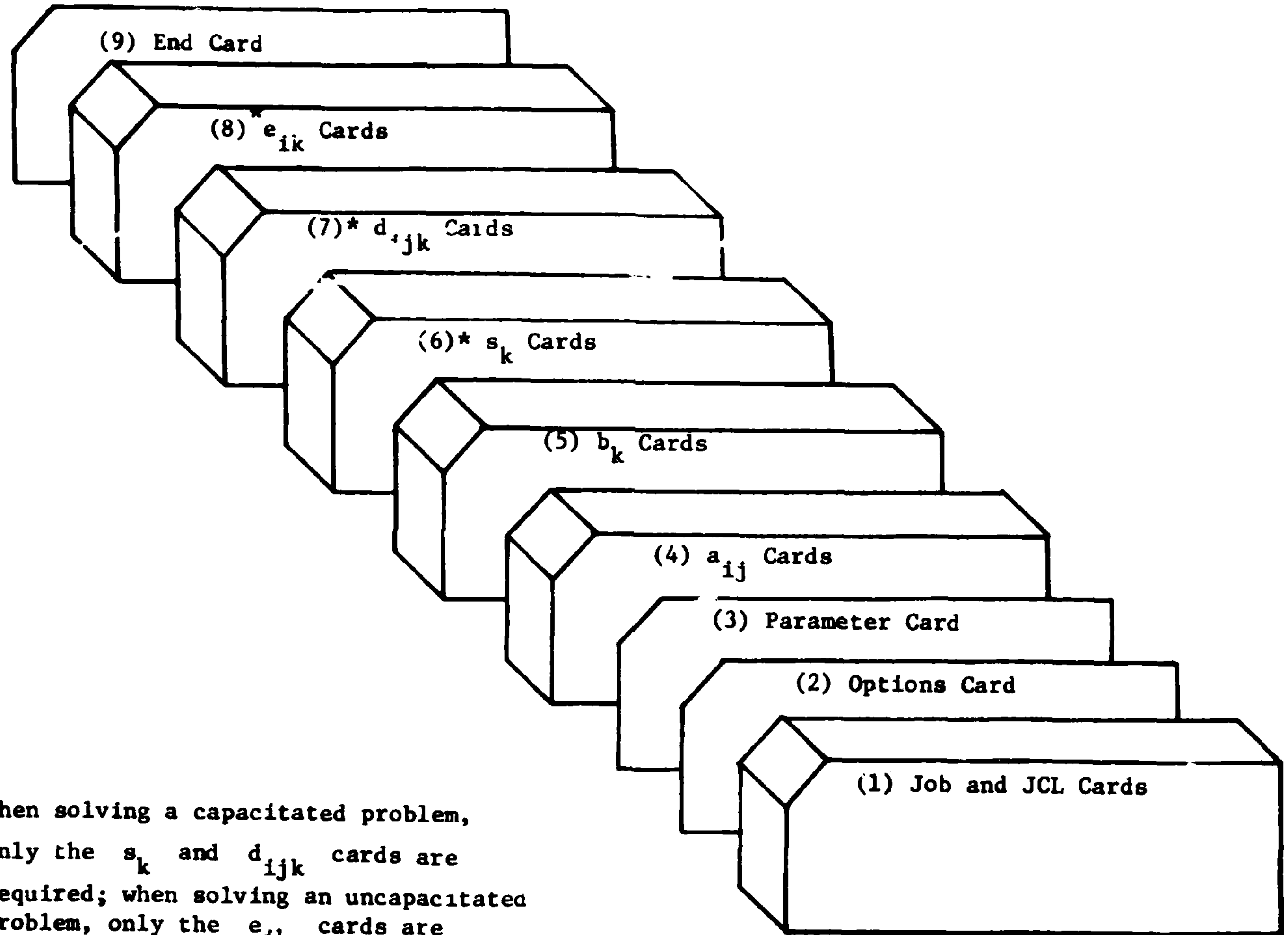
4. <u>User Information</u>

ZIPCAP is designed to solve a multiactivity multifacility capacity-constrained 0-1 assignment problem having formulation (P) . Such a problem is referred to as "capacitated" because the optimal solution must satisfy the capacity constraints. However, the program can be used in a situation where the capacity constraints are not applicable, in other words, for an "uncapacitated" problem. The program, in this case, will generate the necessary information to conform to formulation (P) .

Figure 3 presents a schematic diagram of the deck structure for using ZIPCAP. The cards include both the job control cards and the input data cards required to solve a problem.

Following is a detailed description of these cards (in the same order as presented in Figure 3); it provides the necessary coding information.

(1) <u>Job and JCL Cards</u>: Figures 4A and 4B show two alternatives for the use of JCL cards.



*When solving a capacitated problem, only the s_k and d_{ijk} cards are required; when solving an uncapacitated problem, only the e_{ik} cards are required.

Figure 3

ZIPCAP Data Structure

IBM

FORTRAN Coding Form

PROGRAM

PROGRAMMER | DATE | PUNCHING INSTRUCTIONS | GRAPHIC | PUNCH

FORTRAN STATEMENT

```
//  STANDARD JΦB CARD
//  EXEC FΦRG2
//FΦRT.SYSIN DD *

  (SΦURCE PRΦGRAM)

//GΦ.SYSLIB DD                              }JCL CARDS FΦR
//  DD DSN=GWU.PL1XLIB,DISP=SHR             }TIMET SUBRΦUTINE
//GΦ.SYSIN DD *

  (DATA CARDS)

//
```

Figure 4A

Job and JCL Cards -- Alternative I

IBM

FORTRAN Coding Form

FORTRAN STATEMENT

```
// STANDARD JØB CARD
// EXEC FØRG3, DSN='YØUR LIBRARY', PRØG=ZIPCAP
// FØRT.SYSIN DD *
       ↑
  (SØURCE PRØGRAM)
       ↓
//
```

Step 1: Compile and Store Procedure

```
// STANDARD JØB CARD
// EXEC FØRG6, DSN='YØUR LIBRARY', PRØG=ZIPCAP
//GØ.SYSLIB DD                                 } JCL CARDS FØR
// DD DSN=GWU.PLIXLIB,DISP=SHR                 } TIMET SUBRØUTINE
//GØ.SYSIN DD *
       ↑
  (DATA CARDS)
       ↓
//
```

Step 2: Load and Go Procedure

Figure 4B

Job and JCL Cards -- Alternative II

Alternative I uses the source program for the compilation and execution procedure. In Alternative II, the program is first compiled and stored, and thereafter, for a given problem, only the execution is necessary.

The program uses the TIMET subroutine from the PL1 Library in order to record the elapsed time. This time includes only the execution time of the program and excludes the time needed to read the input and to write the output. The JCL cards needed for this purpose are identified in Figures 4A and 4B . If this subroutine is not available, the corresponding JCL cards are not required. In that case the appropriate source program cards should also be removed. (These cards are identified in the program listing in Appendix A.)

(2) Options Card

The options card is the first input data card and is described in Table 2.

(3) Parameter Card

Table 3 describes the parameter card.

Table 3

THE PARAMETER CARD

Columns	Format	Name	Definition
1-5	I5	M	Number of designs (m)
6-10	I5	N	Number of activities (n)
11-15	I5	P	Number of facilities(p)

Table 2

OPTIONS CARD

Columns	Format	Name	Value and Definition	Remarks
1	I1	IINPT	1 if input listing desired 0 otherwise	Input listing includes: - parameters m , n , p - a_{ij} , b_k , s_k, d_{ijk} , and e_{ik}
2	I1	ICAPR	1 if capacity rule to be used 0 otherwise	When solving a capacitated problem (i.e., where capacity constraints are applicable), set ICAPR=1; for an uncapacitated problem, it should be set to 0.
3	I1	ISTEP	0 if listing of intermediate steps not desired 1 if summary of intermediate steps desired 2 if detailed intermediate steps desired	. Summary of intermediate steps includes, for each node, node number, lower bound, upper bound (if applicable), best upper bound, and branching variable. (This information is sufficient to construct a branch-and-bound tree.) . Detailed intermediate steps list every computational step at every node. This option is useful only when changing/debugging the program. . Even with ISTEP = 0, the output includes the total number of nodes explored.

Continued

Table 2
(Continued)

Columns	Format	Name	Value and Definition	Remarks
4	I1	IUNCAP	0 if solving a capacitated problem 1 if solving an uncapacitated problem	for IUNCAP=0, ICAPR should be 1, and for IUNCAP=1, ICAPR should be 0. (for IUNCAP=1, if ICAPR=1, the program overrides it and sets ICAPR=0)
5-10	F6.5	EPS	A fractional value if a sub-optimal solution is acceptable. This value should be zero if an optimal solution is desired.	EPS = 0.002 implies that the resulting solution may be suboptimal, but is guaranteed to have a value within 0.2% of the optimal value.
11-20	F10.3	ET	Elapsed time in seconds, if specified, at which the node and bounds information is printed.	. Specifying ET is useful in a case where ISTEP = 0 and the program terminates before reaching the final solution. . When applicable, the information printed includes: - the node being explored at the specified time - the best upper bound, the corresponding solution, and the node at which this solution was found - detailed steps for the node being explored, by assuming ISTEP=2 . The above information is useful in assessing the extent of the branch-and-bound tree explored until time ET. . This option uses the TIMET subroutine

(4) to (8) Other Input Data Cards

Table 4 describes the cards for the variable costs a_{ij}, the fixed costs b_k, the capacity availabilities s_k, and the capacity usage values d_{ijk}. However, if solving an uncapacitated problem, i.e., where capacity constraints are not active, the facility-design values e_{ik} are required as input data instead of the s_k and d_{ijk} values.

The input data values for a_{ij}, b_k, s_k, and d_{ijk} are required in integer format. Any real-valued data can be converted into integer values by scaling, and the optimal solution reconverted to real values later. For example, real-valued data of 215.35 , 116.50 , ..., 180.61 can be treated as 21535 , 11650, ..., 18061 ; and an optimal value of 584230 obtained from the program would correspond to 5842.30.

(9) End Card

This is the usual end card following input data cards. It contains '//' in the first two columns.

5. Test Problems and Output Illustrations

5.1 Test Problem 1

5.1.1 Problem formulation (stated for ease of reference)

The problem is to find x_{ij} and y_k variables that

Table 4

OTHER INPUT DATA CARDS

Columns	Format	Name	Definition	Remarks
1-10, 11-20, 21-30, etc.	8I10	A(I,J)	Variable cost of activity j using design i	The order of the input data a_{ij} is a_{11} , a_{21} , ..., a_{m1} ; a_{12} , a_{22} , ..., a_{m2} ; ..., a_{1n} , a_{2n} , ..., a_{mn}
1-10, 11-20, 21-30, etc.	8I10	B(K)	Fixed cost of facility k	The order for the b_k is b_1 , b_2 , ..., b_p
1-10, 11-20, 21-30, etc.	8I10	S(K)	Capacity available at facility k	. The order for the s_k is s_1 , s_2 , ..., s_p . This input is required when solving a capacitated problem only, i.e., for IUNCAP=0. For an uncapacitated problem, i.e., for IUNCAP=1, this information is generated by the program.

Continued

Table 4
(Continued)

Columns	Format	Name	Definition	Remarks
1-10, 11-20, 21-30, etc.	8I10	D(I,J,K)	Capacity required at facility k for activity j when activity j uses design i. A d_{ijk} value is positive if design i uses facility k; otherwise it is 0.	. The order of input data is all d_{ijk} for k=1, then all d_{ijk} for k=2, etc., starting on a new card when the value of k changes. . The order of input for a facility k is $d_{11k}, d_{21k}, \ldots, d_{m1k}$; $d_{12k}, d_{22k}, \ldots, d_{m2k}$; ....; $d_{1nk}, d_{2nk}, \ldots, d_{mnk}$. . This input, similar to s_k, is required for capacitated problems only, i.e., for IUNCAP=0. This information is generated by the program for an uncapacitated problem, i.e., for IUNCAP=1.

Continued

Table 4
(Continued)

Columns	Format	Name	Definition	Remarks
1,2,3,...	80I1	E(I,K)	$e_{ik} = 1$ if design i uses facility k $= 0$ otherwise	. The order of input for the e_{ik} is $e_{11}, e_{21}, \ldots, e_{m1}$; $e_{12}, e_{22}, \ldots, e_{m2}; \ldots, e_{1p}, e_{2p}, \ldots, e_{mp}$. . This input is required when solving an uncapacitated problem only, i.e., for IUNCAP=1. The program, then generates d_{ijk} and s_k as follows: $d_{ijk} = 1$ for all j if $e_{ik} = 1$ $= 0$ for all j if $e_{ik} = 0$ $s_k = n$ for all j . e_{ik} is generated by the program for a capacitated problem, as follows: $e_{ik} = 1$ if $\sum_j d_{ijk}$ is positive (>0) $= 0$ otherwise

$$
(P)\quad \begin{cases}
\text{Minimize} & \sum_{j=1}^{n} \sum_{i=1}^{m} a_{ij}\, x_{ij} + \sum_{k=1}^{p} b_k y_k & (1) \\
\text{subject to} & \sum_{i=1}^{m} x_{ij} = 1 \qquad j=1, \cdots, n & (2) \\
 & \sum_{j=1}^{n} \sum_{i=1}^{m} d_{ijk}\, x_{ij} \leq s_k y_k \quad k=1,\ldots,p & (3) \\
 & x_{ij}\,,\ y_k = 0 \text{ or } 1, & (4)
\end{cases}
$$

where $m = 3$, $n = 4$, $p = 5$.

The a_{ij} values are:

i \ j	1	2	3	4
1	9,847	15,718	4,450	8,925
2	10,192	10,542	4,422	8,118
3	11,019	9,750	4,609	8,337

$b_1 = 1750$ $b_2 = 2000$ $b_3 = 1750$ $b_4 = 1350$ $b_5 = 1000$

$s_1 = 400$ $s_2 = 400$ $s_3 = 1000$ $s_4 = 400$ $s_5 = 400$

The d_{ijk} values are as follows:

k=1

i \ j	1	2	3	4
1	0	0	0	0
2	186	60	261	148
3	154	38	175	100

k=2

i	j=1	j=2	j=3	j=4
1	0	0	0	0
2	0	0	0	0
3	154	38	175	100

k=3

i	j=1	j=2	j=3	j=4
1	525	144	1,011	396
2	186	60	261	148
3	154	38	175	100

k=4

i	j=1	j=2	j=3	j=4
1	0	0	0	0
2	0	0	0	0
3	154	38	175	100

k=5

i	j=1	j=2	j=3	j=4
1	0	0	0	0
2	186	60	261	148
3	154	38	175	100

5.1.2 Coded Input: Figure 5 shows the coded information for test problem 1. Various segments in this figure are numbered in parentheses to correspond to the deck structure of Figure 3.

IBM FORTRAN Coding Form GX28-7327-6 U/M 050** Printed in U.S.A.

PROGRAM | PROGRAMMER | DATE | PUNCHING INSTRUCTIONS | GRAPHIC | PUNCH | PAGE OF | CARD ELECTRO NUMBER

STATEMENT NUMBER | CONT. | FORTRAN STATEMENT | IDENTIFICATION SEQUENCE

```
//  STANDARD JØB CARD                                                            (1)
//  EXEC FØRG6,DSN='YØUR LIBRARY',PRØG=ZIPCAP
//GØ.SYSLIB DD
//  DD  DSN=GWU.PLIXLIB,DISP=SHR
//GØ.SYSIN DD *
111                                                                              (2)
     3     4     5                                                               (3)
        9847     10192     11019     15718     10542     9750     4450     4422  (4)
        4609      8925      8118      8337
        1750      2000      1750      1350      1000                             (5)
         400       400      1000       400       400                             (6)
           0       186       154         0        60       38        0      261  (7)
         175         0       148       100
           0         0       154         0         0       38        0        0
         175         0         0       100
         525       186       154       144        60       38     1011      261
         175       346       148       100
           0         0       154         0         0       38        0        0
         175         0         0       100
           0       186       154         0        60       38        0      261
         175         0       148       100
//                                                                               (9)
```

Figure 5

Coded Input Data for Test Problem 1

T-425

5.1.3 Annotated Output: The computer output for this test problem is presented in Appendix C. Most segments of this output are self-explanatory. The order of these segments is lettered in circles as follows:

(a) Options selected -- information given on options card

(b) m, n, and p -- information from parameter card. This segment is printed if the option IINPT=1

(c) Other input data, i.e., a_{ij} , b_k , s_k and d_{ijk} for a capacitated problem; and e_{ik} as generated by the program. For an uncapacitated problem, input data a_{ij} , b_k , and e_{ik} ; and s_k and d_{ijk} as generated by the program. This segment is printed if the option IINPT=1

(d) Intermediate steps' display, depending on the value of the option ISTEP. No display for ISTEP=0, summary for ISTEP=1, and detailed steps for ISTEP=2

(e) Elapsed time in seconds to execute the program, excluding the time to read the input data and to write the input and output. It includes the time to print intermediate steps if option ISTEP=1 or 2

(f) Total number of nodes explored for the problem solved

(g) Optimal solution showing the x_{ij} variables with value 1, y_k variables with value 1 or 0, and the value of the objective function. If the problem does not have a feasible solution, this segment prints "Problem does not have a feasible solution because the capacity constraints cannot be satisfied."

The node and bounds information at a specified time is displayed only if option KT is assigned a particular value.

5.2 Test Problem 2

This is a capacitated problem having 5 designs, 4 activities and 8 facilities.

The problem formulation and coded input are similar to that of test problem 1.

The annotated output (which includes the listing of input data) is presented in Appendix D. The annotations are the same as those already described for test problem 1. Note that the options selected include ISTEP=0, hence no intermediate steps; and EPS=0.002, implying that the solution may be suboptimal within + 0.2% .

5.3 Test Problem 3

Although ZIPCAP is designed for capacitated problems, it may be used to solve uncapacitated problems as a special case. For demonstration, this test problem is uncapacitated, having 10 designs, 8 activities, and 8 facilities. The input data include the options card, the parameter card, input data values for a_{ij} , b_k , and e_{ik} . The program generates the d_{ijk} and s_k values.

Appendix E shows the annotated output for this problem. Note that the options selected include IINPT=1, hence the data listing; ISTEP=0, therefore no intermediate steps; ICAPR=0 and IUNCAP=1 being an uncapacitated problem. Note that option ET has been specified a value of 30.0 . Thus, if the optimal solution was not reached within 30 seconds, the node and bounds information at time ET would have been displayed.

REFERENCES

[1] GEOFFRION, A.M. (1974). Lagrangean relaxation for integer programming. Mathematical Programming Study 2 82-114.

[2] GEOFFRION, A.M., and R.E. MARSTEN (1972). Integer programming algorithms: a framework and state-of-the-art survey. Management Science 18 465-491.

[3] GROSS, D. and C. E. PINKUS (1979). Designing a support system for repairable items. Computers and Operations Research 6 59-68.

[4] GROSS, D., C.E. PINKUS, and R. M. SOLAND (1979). Designing a multi-product multi-echelon inventory system. Technical Paper Serial T-392. Program in Logistics, Institute for Management Science and Engineering, The George Washington University.

[5] PINKUS, C.E. (1975). Optimal design of multi-product multi-echelon inventory systems. Decision Sciences 6 492-507.

[6] PINKUS, C.E., D. GROSS, and R. M. SOLAND (1973). Optimal design of multiactivity multifacility systems by branch and bound. Operations Research 21 270-283.

APPENDIX A

ZIPCAP LISTING

```
FORTRAN IV G LEVEL  21                MAIN              DATE = 80206          14/35/07

                C                    ZIPCAP, A ZERO-ONE INTEGER PROGRAM IS DESIGNED
                C                    TO SOLVE MULTIACTIVITY MULTIFACILITY CAPACITY-
                C                    CONSTRAINED PROBLEMS HAVING VARIABLE AND FIXED
                C                    COSTS. IT ALSO SOLVES UNCAPACITATED PROBLEMS AS A
                C                    SPECIAL CASE
      0001            INTEGER    D(35,35,30), A(35,35), CX(35,35), E(35,30),
                     1           B(30), BSOLX(35), BSOLY(30), FLB(30),FIX(35),FIXI(35),
                     2           FUB(30), S(30), SOLX(35), STX(1225)
      0002            REAL                MINC(35), NMINC(35)
      0003            DIMENSION C(35,35), DIFBR(35), KT2(35), MIND(35)
      0004            INTEGER    BRO, BR1, FC, FCUB, P
      0005            REAL       LOWB, MAXDIF, MINSC
                C     **************OPTIONS AVAILABLE: IINPT, ICAPP, ISTEP, IUNCAP,EPS
                C                    IINPT=1 IF INPUT LISTING DESIRED; 0 OTHERWISE
                C                    ICAPR=1 IF CAPACITY RULE TO BE USED; 0 OTHERWISE
                C                    ISTEP=0 IF LISTING OF INTERMEDIATE STEPS
                C                    NOT DESIRED. ISTEP=1 IF SUMMARY OF BRANCH &
                C                    BOUND NODES DESIRED. ISTEP=2 IF DETAILED
                C                    LISTING OF INTERMEDIATE STEPS DESIRED.
                C                    IUNCAP=1 IF SOLVING AN UNCAPACITATED PROBLEM,
                C                    0 OTHERWISE.
                C                    EPS= A FRACTIONAL VALUE IF SUBOPTIMAL
                C                    SOLUTION DESIRED, E.G., EPSILON AS 0.005
                C                    IMPLIES SOLUTION TO BE WITHIN +0.5 PERCENT
                C                    OF THE OPTIMAL SOLUTION. EPS=0.0 IF OPTIMAL
                C                    SOLUTION DESIRED.
                C                    ET= ELAPSED TIME IN SECONDS, IF SPECIFIED, AT
                C                    WHICH THE NODE AND BOUNDS RELATED INFORMATION
                C                    IS PRINTED. THIS IS USEFUL IN A SITUATION IF
                C                    ISTEP=0 AND THE PROGRAM TERMINATES BEFORE
                C                    REACHING THE FINAL SOLUTION.
                C     **************READ INPUT DATA**********************************
      0006            READ 10, IINPT, ICAPR, ISTEP, IUNCAP, EPS, ET
      0007         10 FORMAT (4I1, F6.5, F10.3)
                C                    M= NUMBER OF DESIGNS
                C                    N= NUMBER OF ACTIVITIES
                C                    P= NUMBER OF FACILITIES
      0008            READ 20,M,N,P
      0009         20 FORMAT (3I5)
                C                    A(I,J): VARIABLE COST MATRIX
      0010            READ 30,  ((A(I,J), I=1,M),J=1,N)
      0011         30 FORMAT   (8I10)
                C                    B(K): FIXED COST VECTOR
      0012            READ 30, (B(K),K=1,P)
      0013            IF (IUNCAP.EQ.1) GO TO 40
                C                    S(K): CAPACITY LIMIT VECTOR; REQUIRED ONLY
                C                    IF IUNCAP=0
      0014            READ 30, (S(K),K=1,P)
                C                    D(I,J,K): CAPACITY USAGE MATRIX; REQUIRED
                C                    ONLY IF IUNCAP=0
      0015            DO 32 K=1,P
      0016            READ 30,((D(I,J,K), I=1,M),J=1,N)
      0017         32 CONTINUE
      0018            DO 37 K=1,P
      0019            DO 37 I=1,M
      0020               IF (D(I,1,K).EQ.0) GO TO 35
      0021               E(I,K)=1
      0022               GO TO 37
```

```
FORTRAN IV G LEVEL  21                 MAIN              DATE = 80206          14/35/07

0023          35    E(I,K)=0
0024          37 CONTINUE
0025             GO TO 90
         C                    E(I,K): DESIGN-FACILITY MATRIX; REQUIRED ONLY
         C                    IF IUNCAP=1
0026          40 READ 45,((E(I,K),I=1,M),K=1,P)
0027          45 FORMAT  (80I1)
0028             DO 80 K=1,P
0029                S(K)=N
0030             DO 75 I=1,M
0031                IF (E(I,K).EQ.1) GO TO 65
0032             DO 60 J=1,N
0033                D(I,J,K)=0
0034          60 CONTINUE
0035                GO TO 75
0036          65 DO 70 J=1,N
0037                D(I,J,K)=1
0038          70 CONTINUE
0039          75 CONTINUE
0040          80 CONTINUE
         C       **************PRINT INPUT DATA**********************************
0041          90 PRINT 95, IINPT,  ICAPR, ISTEP, IUNCAP, EPS, ET
0042          95 FORMAT ( '1','  OPTIONS SELECTED :  IINPT=',I1,
               1          '  ICAPR=',I1, '  ISTEP=',I1, '  IUNCAP=',I1,
               2          '  EPS=',F8.5, '  ET=', F10.3///)
0043             IF (IINPT.EQ.0) GO TO 168
0044             PRINT  100,M,N,P
0045         100 FORMAT ('0', T55, 'INPUT DATA',/1X, T55, '---------- ----------',//////1X,
               1T41, 'NUMBER OF DESIGNS (M)=', 4X,I4//1X,T41,
               2'NUMBER OF ACTIVITIES (N)=', 1X,I4//1X, T41,
               3'NUMBER OF FACILITIES (P)=',1X, I4///)
0046             PRINT  105
0047         105 FORMAT ( 4X,     'VARIABLE COST MATRIX  A(I,J)',/4X,
               1'-------------------------------',/)
0048             DO  110 I=1,M
0049         110 PRINT 115, I, (A(I,J),J=1,N)
0050         115 FORMAT ('0', T6, 'I=', I3, 4X,8I13, 4(/, 14X,8I13))
0051             PRINT 120
0052         120 FORMAT('0',//4X,'FIXED COST VECTOR  B(K)',/4X,
               1'--------------------------',      /)
0053             PRINT 122, (B(K),K=1,P)
0054         122 FORMAT ('0', T15, 8I13, 3(/, 14X,8I13))
0055             PRINT  125
0056         125 FORMAT('0',//4X,'CAPACITY LIMIT VECTOR  S(K)',/4X,
               1'-----------------------------',  /)
0057             PRINT 128, (S(K),K=1,P)
0058         128 FORMAT ('0', T15, 8I13, 3(/, 14X,8I13))
0059             PRINT 130
0060         130 FORMAT('0',//4X, 'CAPACITY USAGE MATRIX  D(I,J,K)',/4X,
               1'---------------------------------',/)
0061             DO  150 K=1,P
0062             PRINT  135,K
0063         135 FORMAT ('0',//5X,'K=',I3/)
0064             DO  145 I=1,M
0065             PRINT  140,I,(D(I,J,K), J=1,N)
0066         140 FORMAT ('0', T6, 'I=', I3, 4X,8I13, 4(/, 14X,8I13))
0067         145 CONTINUE
0068         150 CONTINUE
```

```
FORTRAN IV G LEVEL  21                    MAIN                DATE = 80206          14/35/07
 0069              PRINT 155                                                          00001120
 0070          155 FORMAT('0',//4X,'DESIGN-FACILITY MATRIX E(I,K)',/4X,               00001130
              1'------------------------------',/)                                    00001140
 0071              DO 160  I=1,M                                                      00001150
 0072              PRINT 158, I, (E(I,K),K=1,P)                                       00001160
 0073          158 FORMAT ('0', T6, 'I=', I3, 4X,8I13, 3(/, 14X,8I13))                00001170
 0074          160 CONTINUE                                                           00001180
 0075          168 IF (ISTEP.EQ.0) GO TO 190                                          00001190
 0076              IF (ISTEP.EQ.1) GO TO 175                                          00001200
 0077              PRINT 170                                                          00001210
 0078          170 FORMAT ('0',///55X,'DETAILED LISTING OF STEPS',/)                  00001220
 0079              GO TO 190                                                          00001230
 0080          175 PRINT 180                                                          00001240
 0081          180 FORMAT ('0',///55X,'SUMMARY OF STEPS',/)                           00001250
             C     **************INITIALIZE***************************************** 00001260
 0082          190 BUB=9999999.                                                       00001270
 0083              BUBS= BUB/ (1.0+EPS)                                               00001280
 0084              NSX=0                                                              00001290
 0085              NOD=1                                                              00001310
 0086              IBNOD=1                                                            00001315
 0087              INFT=0                                                             00001320*
 0088              INSET=0                                                            00001330*
 0089              DO 205 J=1,N                                                       00001390
 0090                 FIX(J)=0                                                        00001400
 0091                 KT2(J)=0                                                        00001410
 0092              DO 205 I=1,M                                                       00001420
 0093                 CX(I,J)=0                                                       00001430
 0094          205 CONTINUE                                                           00001433
 0095              LQ1=0                                                              00001436
 0096              LQ2=0                                                              00001440
 0097              LR2=0                                                              00001443
 0098              CALL TIMET(ITO)                                                    00001445*
 0099              IF(NSX.EQ.0) GO TO 283                                             00001447
             C                         CX(I,J) CONTAINS FIXED AND FREE X(I,J) VARIABLES.  00001450
             C                         STX(INS) CONTAINS FIXED X(I,J) VARIABLES.      00001460
             C                         CX(I,J) AND STX(INS) ARE UPDATED BY THE CAPACITY   00001480
             C                         RULE, THE BOUNDING RULE, AND THE RULE FOR      00001490
             C                         BRANCHING AND BACKTRACKING.                    00001500
             C                         IN CX(I,J) A FIXED VARIABLE IS RECORDED AS 1 OR    00001505
             C                         2, AND A FREE VARIABLE AS 0.                   00001510
             C                         A VALUE OF 1 IMPLIES THAT THAT PARTICULAR VARIABLE 00001515
             C                         IS FIXED, AND FIX(J) IS SET EQUAL TO 1 IMPLYING    00001520
             C                         THAT COLUMN J HAS A FIXED VARIABLE OF VALUE 1. 00001525
             C                         A VALUE OF 2 IMPLIES THAT THAT PARTICULAR VARIABLE 00001530
             C                         SHOULD NOT BE CONSIDERED FOR CURRENT COMPUTATIONS. 00001535
             C                         AN X(I,J) RECORDED IN CX(I,J) AS 1 DUE TO THE  00001540
             C                         BRANCHING RULE IS RECORDED IN STX(INS) AS X*100+J. 00001545
             C                         AN X(I,J) RECORDED IN CX(I,J) AS 1 DUE TO THE  00001550
             C                         CAPACITY RULE OR THE BOUNDING RULE IS RECORDED IN  00001555
             C                         STX(INS) AS (X*100+J)+1000000.                 00001560
             C                         AN X(I,J) RECORDED IN CX(I,J) AS 2 IS RECORDED IN  00001565
             C                         STX(INS) AS -(X*100+J)-1000000.                00001570
 0100          210 IF (ISTEP.EQ.0) GO TO 225                                          00001580
 0101          215 PRINT 220,NOD                                                      00001590
 0102          220 FORMAT ('0',// 6X,'NODE NUMBER',I5/)                               00001600
             C     **************UPDATE CX(I,J) FOR BRO***************************   00001610
             C                         BRO IS THE RIGHT BRANCHING VARIABLE            00001615
 0103          225 LX=BRO                                                             00001620
```

*Relevant to the TIMET Subroutine

```
FORTRAN IV G LEVEL  21                 MAIN                DATE = 80206          14/35/07

 0104              IX=LX/100                                                     00001630
 0105              JX=LX-IX*100                                                  00001640
 0106              CX(IX,JX)=2                                                   00001650
 0107              KT2(JX)=KT2(JX)+1                                             00001660
 0108              FIX(JX)=0                                                     00001720
 0109              LQ1=LQ1-1                                                     00001725
 0110              IF (KT2(JX).LT.(M-1)) GO TO 270                               00001730
 0111              DO 255 I=1,M                                                  00001740
 0112                 IF (CX(I,JX).EQ.2) GO TO 255                               00001750
 0113                 CX(I,JX)=1                                                 00001760
 0114                 NSX=NSX+1                                                  00001763
 0115                 STX(NSX)=   (I*100+JX)+1000000                             00001766
 0116                 FIX(JX)=1                                                  00001770
 0117                 LQ1=LQ1+1                                                  00001780
 0118                 FIXI(JX)=I                                                 00001790
 0119                 GO TO 270                                                  00001800
 0120          255 CONTINUE                                                      00001810
 0121          270 LQ2=0                                                         00001820
 0122              LR2=0                                                         00001825
 0123              GO TO 283                                                     00001830
 0124          272 IF (ISTEP.EQ.0) GO TO 276                                     00001840
 0125              PRINT 220,NOD                                                 00001850
             C     **************UPDATE CX(I,J) FOR BR1************************* 00001853
             C                    BR1 IS THE LEFT BRANCHING VARIABLE             00001856
 0126          276 LQ2=0                                                         00001860
 0127              LR2=0                                                         00001866
 0128              LX=BR1                                                        00001870
 0129              IX=LX/100                                                     00001875
 0130              JX=LX-IX*100                                                  00001880
 0131              CX(IX,JX)=1                                                   00001885
 0132              FIX(JX)=1                                                     00001890
 0133              LQ1=LQ1+1                                                     00001892
 0134              DO 279 I=1,M                                                  00001895
 0135                 IF (IX.EQ.I) GO TO 281                                     00001897
 0136          279 CONTINUE                                                      00001900
 0137          281 FIXI(JX)=IX                                                   00001902
 0138          283 IF (ISTEP.NE.2) GO TO 300                                     00001905
 0139          285 DO 295 I=1,M                                                  00001910
 0140                 PRINT 290, I,(CX(I,J),J=1,N)                               00001920
 0141          290    FORMAT (/5X,'CX(I,J)',4X,'I=',I3,2X, 20I4/23X, 20I4)       00001930
 0142          295 CONTINUE                                                      00001940
 0143              PRINT 297,(FIX(J),J=1,N)                                      00001950
 0144          297 FORMAT (/5X,'FIX(J)',12X, 20I4/23X, 20I4)                     00001960
             C     **************APPLY CAPACITY RULE***************************** 00001970
             C                    AND UPDATE CX(I,J) AND STX(INS).               00001980
 0145          300 IF (IUNCAP.EQ.1) GO TO 2300                                   00002010
 0146          305 IF (ICAPR.EQ.0) GO TO 2300                                    00002020
 0147          310 DO 2000 K=1,P                                                 00002030
             C                    FIND THE SUM OF MINIMUM D(I,J,K) OVER EACH J FOR A 00002040
             C                    GIVEN K, I.E., MINSD= SUM OF MIND(J)           00002050
 0148              MINSD=0                                                       00002060
 0149              DO 900 J=1,N                                                  00002070
 0150                 IF(FIX(J).EQ.0) GO TO 350                                  00002080
             C                    IF  FIX(J)=1, SET MIND(J)=D(I,J,K) FOR CX(I,J)=1  00002090
             C                    AND MOVE TO NEXT COLUMN J                      00002100
 0151              INDI=FIXI(J)                                                  00002110
 0152              MIND(J)=D(INDI,J,K)                                           00002120
 0153              GO TO 800                                                     00002130
```

```
FORTRAN IV G LEVEL  21                  MAIN                 DATE = 80206          14/35/07

      0154          350     LK=0                                                              00002160
      0155                  I=1                                                               00002170
      0156                  MIND(J)=D(I,J,K)                                                  00002180
                 C                      SKIP D(I,J,K) WHEN CX(I,J)=2 & MOVE TO NEXT ROW I     00002190
      0157          400     IF(CX(I,J).EQ.2) GO TO 600                                        00002200
      0158          500     IF(D(I,J,K).LT.MIND(J)) MIND(J)=D(I,J,K)                          00002210
      0159                  GO TO 700                                                         00002220
      0160          600     LK=LK+1                                                           00002230
      0161                  IF(I.GT.LK) GO TO 700                                             00002240
      0162                  I=I+1                                                             00002250
      0163                  MIND(J)=D(I,J,K)                                                  00002260
      0164                  GO TO 750                                                         00002270
      0165          700     I=I+1                                                             00002280
      0166          750     IF(I.LE.M) GO TO 400                                              00002290
      0167          800     MINSD=MINSD+MIND(J)                                               00002300
      0168          900 CONTINUE                                                              00002310
      0169          910 IF (ISTEP.NE.2) GO TO 980                                             00002320
      0170                  PRINT 950, K, MINSD,(MIND(J),J=1,N)                               00002330
      0171          950     FORMAT('0','K, MINSD,(MIND(J),J=1,N)',10I10)                      00002340
                 C                      FIND BALANCE AVAILABLE CAPACITY IBALD FOR A GIVEN K   00002350
                 C                      IF IBALD IS NEGATIVE, THEN BACKTRACK.                 00002360
      0172          980     IBALD=S(K)-MINSD                                                  00002380
      0173         1000 IF (IBALD.LT.0) GO TO 6200                                            00002390
      0174              DO 1500 J=1,N                                                         00002400
                 C                      SKIP COLUMN J IF FIX(J)=1                             00002410
      0175                  IF (FIX(J).EQ.1) GO TO 1500                                       00002420
      0176              DO 1300 I=1,M                                                         00002430
                 C                      SKIP ROW I IF CX(I,J)=2                               00002440
      0177         1100     IF(CX(I,J).EQ.2) GO TO 1300                                       00002450
                 C                      COMPUTE DIFFERENCE BETWEEN D(I,J,K) AND MIND(J).      00002470
                 C                      IF IT IS MORE THAN AVAILABE BALANCE, SET CX(I,J)=2    00002480
      0178         1200     IDIFD=D(I,J,K)-MIND(J)                                            00002490
      0179                  IF ((IDIFD-IBALD).LE.0)  GO TO 1300                               00002510
      0180                  CX(I,J)=2                                                         00002520
      0181                  NSX=NSX+1                                                         00002523
      0182                  STX(NSX)=-(I*100+J)-1000000                                       00002526
                 C                      LQ2 COUNTS THE NUMBER OF CX(I,J) VALUES SET EQUAL     00002530
                 C                      TO 2 IN A CYCLE                                       00002540
      0183                  LQ2=LQ2+1                                                         00002550
                 C                      KT2(J) KEEPS AN ACCOUNT OF CX(I,J) VALUES SET EQUAL   00002560
                 C                      TO 2 FOR COLUMN J                                     00002570
      0184                  KT2(J)=KT2(J)+1                                                   00002580
                 C                      FOR COLUMN J, IF ALL BUT ONE CX(I,J) VALUES ARE       00002590
                 C                      EQUAL TO 2, SET THAT CX(I,J)=1 &  SET FIX(J)=1        00002600
      0185                  IF(KT2(J).LT.(M-1)) GO TO 1300                                    00002610
      0186              DO 1250 LR=1,M                                                        00002620
      0187                  IF(CX(LR,J).EQ.2) GO TO 1250                                      00002630
      0188                  CX(LR,J)=1                                                        00002640
      0189                  NSX=NSX+1                                                         00002643
      0190                  STX(NSX)=    (LR*100+J)+1000000                                   00002646
      0191                  FIX(J)=1                                                          00002650
                 C                      LQ1 KEEPS AN ACCOUNT OF COLUMNS FOR WHICH FIX(J)=1    00002655
      0192                  LQ1=LQ1+1                                                         00002660
                 C                      FIXI(J) SPECIFIES INDEX I FOR WHICH FIX(J)=1          00002662
      0193                  FIXI(J)=LR                                                        00002665
      0194                  GO TO 1500                                                        00002670
      0195         1250 CONTINUE                                                              00002680
      0196         1300 CONTINUE                                                              00002690
```

```
FORTRAN IV G LEVEL  21                MAIN              DATE = 80206          14/35/07

 0197          1500 CONTINUE                                                    00002700
 0198          1800 IF (ISTEP.NE.2) GO TO 2000                                  00002710
 0199               PRINT 1900, K, LQ2, LQ1                                     00002720
 0200          1900 FORMAT ('0','K=',I3,'  LQ2=',I3, '  LQ1=', I3)              00002730
 0201               DO 1930 I=1,M                                               00002740
 0202                  PRINT 290, I,(CX(I,J),J=1,N)                             00002750
 0203          1930 CONTINUE                                                    00002770
 0204               PRINT 297,  (FIX(J),J=1,N)                                  00002780
 0205          2000 CONTINUE                                                    00002800
             C                       A CYCLE EXAMINES ALL THE FACILITIES.       00002803
             C                       IF IN A CYCLE, THE CAPACITY RULE RESULTS IN SETTING 00002810
             C                       ADDITIONAL CX(I,J) VALUES EQUAL TO 2, THEN REPEAT   00002820
             C                       THE CYCLE. BUT IF FIX(J)=1 FOR ALL J, THEN DO NOT   00002830
             C                       REPEAT THE CYCLE.                          00002835
 0206               IF (LQ1.EQ.N) GO TO 2300                                    00002840
 0207               IF (LQ2.EQ.LR2) GO TO 2300                                  00002845
 0208          2200 LR2=LQ2                                                     00002860
 0209               GO TO 300                                                   00002870
             C      **************SOLVE (LAGRANGIAN) RELAXED PROBLEM************** 00002880
             C                       FIND FLB(K) -- VECTOR OF FACILIES FOR COMPUTING     00002890
             C                       C(I,J) MATRIX & LOWER BOUND. IT HAS VALUE 1 IF A    00002900
             C                       FACILITY IS USED, OTHERWISE IT HAS 0 VALUE.         00002910
 0210          2300 DO 2500 K=1,P                                               00002920
 0211                  FLB(K)=0                                                 00002930
 0212          2500 CONTINUE                                                    00002940
 0213               DO 3000 J=1,N                                               00002950
 0214                  IF (FIX(J).EQ.0) GO TO 3000                              00002960
 0215                  INDI=FIXI(J)                                             00002970
 0216               DO 2550 K=1,P                                               00002990
 0217                  IF (E(INDI,K).EQ.0) GO TO 2550                           00003000
 0218                  IF (FLB(K).EQ.1) GO TO 2550                              00003010
 0219                  FLB(K)=1                                                 00003020
 0220          2550 CONTINUE                                                    00003030
 0221          3000 CONTINUE                                                    00003060
 0222               IF (ISTEP.NE.2) GO TO 3150                                  00003070
 0223               PRINT 3100, (FLB(K),K=1,P)                                  00003080
 0224          3100 FORMAT('0','(FLB(K),K=1,P)  ',  20I4/16X,20I4)              00003090
             C                       COMPUTE COST MATRIX C(I,J) FOR THE RELAXED PROBLEM  00003100
 0225          3150 DO 3400 J=1,N                                               00003110
 0226               DO 3300 I=1,M                                               00003120
 0227                  BSUM=0.0                                                 00003130
 0228               DO 3200 K=1,P                                               00003140
 0229                  IF (FLB(K).EQ.1) GO TO 3200                              00003150
 0230                  IF (E(I,K).EQ.0) GO TO 3200                              00003160
 0231                  BSUM=BSUM+(B(K) * (FLOAT(D(I,J,K))/ FLOAT(S(K))))        00003170
 0232          3200 CONTINUE                                                    00003180
 0233          3250    C(I,J)=A(I,J)+BSUM                                       00003190
 0234          3300 CONTINUE                                                    00003200
 0235          3400 CONTINUE                                                    00003210
 0236               IF (ISTEP.NE.2) GO TO 3445                                  00003220
 0237               DO 3430 I=1,M                                               00003230
 0238                  PRINT 3420, I, (C(I,J),J=1,N)                            00003250
 0239          3420    FORMAT (/5X, 'C(I,J)',5X, 'I=', I3,2X, 5F15.4,           00003260
                  1            6(/23X, 5F15.4))                                 00003265
 0240          3430 CONTINUE                                                    00003270
             C                       FIND SUM OF MINIMUM C(I,J) VALUES OVER EACH J,      00003290
             C                       I.E., MINSC=SUM OF MINC(J).                00003300
             C                       IF FIX(J)=1, THEN MINC(J)=C(I,J) WHERE CX(I,J)=1    00003310
```

```
       3445 MINSC=0.0
            DO 3900 J=1,N
               IF (FIX(J).EQ.0) GO TO 3500
               INDI=FIXI(J)
               MINC(J)=C(INDI,J)
               SOLX(J)=INDI
               GO TO 3850
       3500    LK=0
               I=1
      C                   SKIP C(I,J) ELEMENT IF CX(I,J)=2 & MOVE TO NEXT I
       3550    IF (CX(I,J).EQ.2) GO TO 3700
               IF ((I-LK).EQ.1) GO TO 3600
               IF (C(I,J).GE.MINC(J)) GO TO 3750
       3600    MINC(J)=C(I,J)
               IMIN=I
               GO TO 3750
       3700    LK=LK+1
       3750    I=I+1
       3800    IF (I.LE.M) GO TO 3550
               SOLX(J)=IMIN
       3850    MINSC=MINSC+MINC(J)
       3900 CONTINUE
            IF (ISTEP.NE.2) GO TO 3940
            DO 3920 J=1,N
               PRINT 3910, J,MINC(J),SOLX(J)
       3910 FORMAT ('0','J,MINC(J),SOLX(J)', I5,F15.4,I6)
       3920 CONTINUE
      C                   COMPUTE FIXED COST FC FOR LOWER BOUND
       3940 FC=0
            DO 4000 K=1,P
               IF (FLB(K).EQ.0) GO TO 4000
       3950    FC=FC+B(K)
       4000 CONTINUE
      C     **************COMPUTE LOWER BOUND LOWB************************
       4050 LOWB=MINSC+FC
            IF (ISTEP.EQ.0) GO TO 4150
            PRINT 4120, MINSC, FC, LOWB
       4120 FORMAT ('0',' MINSC, FC, LOWB ', F15.4, I15, F15.4)
      C                   COMPARE LOWER BOUND WITH BEST UPPER BOUND STAR
      C                   BUBS WHICH EQUALS BUB/(1+EPS). IF LOWB IS
      C                   GREATER THAN OR EQUAL TO BUBS, THEN BACKTRACK
       4150 IF (LOWB.GE.BUBS)GO TO 6200
      C                   CHECK IF CURRENT SOLUTION SATISFIES CAPACITY
      C                   CONSTRAINTS
       4200 IF (IUNCAP.EQ.1) GO TO 4420
       4210 DO 4400 K=1,P
               NSUMD=0
            DO 4300 J=1,N
               IX=SOLX(J)
               NSUMD=NSUMD+D(IX,J, K)
       4300 CONTINUE
            IF (ISTEP.NE.2) GO TO 4320
               PRINT 4310, K,NSUMD
       4310    FORMAT ('0', 'K,NSUMD',2I10)
       4320    IF(NSUMD.LE.S(K)) GO TO 4400
               GO TO 5100
       4400 CONTINUE
      C     **************COMPUTE UPPER BOUND UPB IF CAPACITY CONSTRAINTS
```

```
FORTRAN IV G LEVEL  21                MAIN              DATE = 40206        14/35/07
                C                    ARE SATISFIED.                                   00004050
                C                    UPB=SUM OF A(I,J)+FIXED COST FCUB BASED ON       00004060
                C                    SOLUTION VECTOR SOLX(J)                          00004070
                C                    VECTOR OF FACILITIES FOR UPPER BOUND FUB(K) HAS  00004080
                C                    VALUES 1 OR 0 BASED ON FACITILY USED OR OTHERWISE 00004090
  0290           4420 DO 4450 K=1,P                                                   00004100
  0291                   FUB(K)=0                                                     00004110
  0292           4450 CONTINUE                                                        00004120
  0293                NSUMA=0                                                         00004130
  0294                FCUB=0                                                          00004140
  0295           4500 DO 4650 J=1,N                                                   00004150
  0296                   IX=SOLX(J)                                                   00004170
  0297                   NSUMA=NSUMA+A(IX,J)                                          00004180
  0298           4550 DO 4600 K=1,P                                                   00004190
  0299                   IF(E(IX,K).EQ.0) GO TO 4600                                  00004200
  0300                   IF(FUB(K).EQ.1) GO TO 4600                                   00004210
  0301                   FUB(K)=1                                                     00004220
  0302                   FCUB=FCUB+B(K)                                               00004230
  0303           4600 CONTINUE                                                        00004240
  0304           4650 CONTINUE                                                        00004250
  0305                IF (ISTEP.NE.2) GO TO 4700                                      00004260
  0306                PRINT 4660, (FUB(K),K=1,P)                                      00004270
  0307           4660 FORMAT('0','(FUB(K),K=1,P)  ',  20I4/16X,20I4)                  00004280
  0308           4700 UPB=NSUMA+FCUB                                                  00004290
  0309           4708 IF (ISTEP.EQ.0) GO TO 4750                                      00004300
  0310                PRINT 4710, NSUMA, FCUB, UPB, BUB, BUBS                         00004310
  0311           4710 FORMAT('0','NSUMA, FCUB, UPB, BUB, BUBS ',2I10, 3F15.4)         00004320
                C                    COMPARE UPPER BOUND WITH BEST UPPER BOUND        00004330
                C                    IF UPB IS LESS THAN BUB, SET IT AS BUB AND       00004340
                C                    NOTE THE SOLUTION                                00004350
  0312           4750 IF (UPB.GE.BUB) GO TO 5100                                      00004360
  0313           4770 BUB=UPB                                                         00004370
  0314                BUBS= BUB/ (1.0+EPS)                                            00004380
  0315                IBNOD=NOD                                                       00004385
  0316                DO 4800 J=1,N                                                   00004390
  0317           4800 BSOLX(J)=SOLX(J)                                                00004400
  0318                DO 4850 K=1,P                                                   00004410
  0319           4850 BSOLY(K)=FUB(K)                                                 00004420
                C     **************COMPARE LOWB WITH BUBS. IF LOWB IS GREATER        00004430
                C                    THAN OR EQUAL TO BUBS, THEN BACKTRACK            00004440
  0320           4900 IF (LOWB.GE.BUBS)GO TO 6200                                     00004450
                C     **************IF FIX(J) VALUES ARE 1 FOR EACH J, THEN BACKTRACK 00004480
  0321           5100 IF (LQ1.EQ.N) GO TO 6200                                        00004500
                C     **************APPLY THE BOUNDING RULE************************** 00004510
                C                    IF THE DIFFERENCE BETWEEN C(I,J) AND MINC(J) IS  00004515
                C                    GREATER THAN THE DIFFERENCE BETWEEN BUBS AND     00004520
                C                    LOWB, THEN CX(I,J)=2                             00004525
                C     **************APPLY BRANCHING RULE AND FIND BR1, THE NEXT       00004530
                C                    VARIABLE FOR LEFT BRANCHING.                     00004540
                C                    FIND NMINC(J), THE NEXT HIGHER VALUE THAN MINC(J) 00004550
                C                    AND DIFBR(J), THE DIFFERENCE BETWEEN THEM.       00004555
  0322                DBOUND=BUBS-LOWB                                                00004568
  0323           5200 DO 5250 J=1,N                                                   00004570
  0324                   NMINC(J)=0.0                                                 00004580
  0325                   DIFBR(J)=0.0                                                 00004590
  0326           5250 CONTINUE                                                        00004600
  0327                DO 5600 J=1,N                                                   00004610
                C                    SKIP TO NEXT J IF FIX(J)=1                       00004620
```

```
FORTRAN IV G LEVEL  21                  MAIN               DATE = 80206          14/35/07

 0328                  IF (FIX(J).EQ.1) GO TO 5600                                       00004630
 0329                  LK=0                                                              00004640
 0330                  I=1                                                               00004650
              C                     SKIP C(I,J) IF CX(I,J)=2 & MOVE TO NEXT I            00004670
 0331          5300    IF (CX(I,J).EQ.2) GO TO 5350                                      00004680
 0332                  IF (I.EQ.SOLX(J)) GO TO 5350                                      00004690
 0333                  IF ((C(I,J)-MINC(J)).GT.DBOUND) GO TO 5330                        00004700
 0334                  IF ((I-LK).EQ.1) GO TO 5320                                       00004710
 0335                  IF (C(I,J).GE.NMINC(J)) GO TO 5400                                00004720
 0336          5320    NMINC(J)=C(I,J)                                                   00004730
 0337                  GO TO 5400                                                        00004735
 0338          5330    CX(I,J)=2                                                         00004740
 0339                  NSX=NSX+1                                                         00004742
 0340                  STX(NSX)=-(I*100+J)-1000000                                       00004745
 0341                  KT2(J)=KT2(J)+1                                                   00004747
 0342                  IF(KT2(J).LT.(M-1)) GO TO 5350                                    00004750
 0343                  INDI=SOLX(J)                                                      00004752
 0344                  CX(INDI,J)=1                                                      00004755
 0345                  NSX=NSX+1                                                         00004758
 0346                  STX(NSX)= (INDI*100+J)+1000000                                    00004760
 0347                  FIX(J)=1                                                          00004762
 0348                  LQ1=LQ1+1                                                         00004764
 0349                  FIXI(J)=INDI                                                      00004766
 0350                  GO TO 5600                                                        00004768
 0351          5350    LK=LK+1                                                           00004770
 0352          5400    I=I+1                                                             00004775
 0353                  IF(I.LE.M) GO TO 5300                                             00004780
 0354          5500    DIFBR(J)=NMINC(J)-MINC(J)                                         00004785
 0355          5600 CONTINUE                                                             00004790
 0356               IF (ISTEP.NE.2) GO TO 5650                                           00004795
 0357               DO 5620 I=1,M                                                        00004820
 0358                  PRINT 290, I,(CX(I,J),J=1,N)                                      00004830
 0359          5620 CONTINUE                                                             00004850
 0360               PRINT 297,  (FIX(J),J=1,N)                                           00004860
              C                     IF FIX(J)=1 FOR ALL J, THEN BACKTRACK.               00004880
 0361          5650 IF (LQ1.EQ.N) GO TO 6200                                             00004890
              C                     FIND MAXDIF, THE MAXIMUM DIFFERENCE DIFBR(J)         00004900
 0362               LF=0                                                                 00004905
 0363               DO 5800 J=1,N                                                        00004910
 0364                  IF (FIX(J).EQ.1) GO TO 5690                                       00004915
 0365                  IF ((J-LF).EQ.1) GO TO 5660                                       00004920
 0366                  IF (DIFBR(J).LT.MAXDIF) GO TO 5800                                00004925
 0367          5660    MAXDIF=DIFBR(J)                                                   00004930
 0368                  LJ=J                                                              00004935
 0369                  GO TO 5800                                                        00004940
 0370          5690 LF=LF+1                                                              00004943
 0371          5800 CONTINUE                                                             00004946
 0372               IF (ISTEP.NE.2) GO TO 5840                                           00004950
 0373               DO 5820 J=1,N                                                        00004953
 0374                  IF (FIX(J).EQ.1) GO TO 5820                                       00004956
 0375                  PRINT 5810, J, NMINC(J), MINC(J), DIFBR(J)                        00004960
 0376          5810 FORMAT ('0','J,NMINC(J),MINC(J),DIFBR(J)', I5,3F15.4)                00004963
 0377          5820 CONTINUE                                                             00004966
              C     ***************BRANCHING VARIABLE BR1 CORRESPONDS TO MAXDIF*****     00004970
 0378          5840 DO 5900 J=1,N                                                        00004980
 0379                  IF (J.NE.LJ) GO TO 5900                                           00004990
 0380          5850    BR1=SOLX(J)*100+J                                                 00005000
 0381               IF (ISTEP.EQ.0) GO TO 6020                                           00005010
```

```
FORTRAN IV G LEVEL  21                MAIN              DATE = 80206          14/35/07
0382                PRINT 5880,  BR1                                        00005020
0383          5880  FORMAT('0',' BR1',I10)                                  00005030
0384                GO TO 6020                                              00005040
0385          5900 CONTINUE                                                 00005050
              C     **************UPDATE STX(INS) AND NSX*************************  00005060
              C                    NSX REPRESENTS THE NUMBER OF VARIABLES IN STX(INS)  00005070
0386          6020 NSX=NSX+1                                                00005090
0387          6040 STX(NSX)=BR1                                             00005100
0388               IF (ISTEP.NE.2) GO TO 6100                               00005150
0389               PRINT 6088, (STX(INS), INS=1,NSX)                        00005160
0390          6088 FORMAT('0',' STX(INS)', 10I10, 122(/, 10X,10I10))        00005170
              C     **************MOVE TO THE NEXT (LEFT BRANCH) NODE AND APPLY  00005220
              C                    CAPACITY RULE                            00005230
0391          6100 NOD=NOD+1                                                00005240
0392          6110 IF (ET.EQ.0.0) GO TO 6150                                00005242*
0393               IF(INSET.EQ.1) GO TO 6147                                00005244*
0394               IF (INET.EQ.1) GO TO 6150                                00005246*
0395               CALL TIMET(INT)                                          00005248*
0396               ELTN=(INT-ITO)*26.04E-6                                  00005250*
0397               IF (ELTN.LT.ET) GO TO 6150                               00005253*
0398          6120 PRINT 6125, NOD, ELTN, BUB, BUBS, IBNOD                  00005256*
0399          6125 FORMAT ('0', 'WAS AT NODE',I6, ' AT ELAPSED TIME =', F10.4,  00005260*
                 1          ' SECONDS.',/1X, '  BUB=',F15.4, '  BUBS=',F15.4,  00005263*
                 2           '  AT NODE=',I7)                               00005266*
0400               IBUB=BUB                                                 00005267*
0401               IF (IBUB.EQ.9999999) GO TO 6146                          00005268*
0402          6130 PRINT 6135, (BSOLX(J),J=1,N)                             00005270*
0403          6135 FORMAT('0', 'SOLUTION CORRESPONDING TO BUB IS',  //1X,   00005273*
                 1             '(BSOLX(J), J=1,N)',10I8,3(/18X,10I8))       00005276*
0404          6140 PRINT 6145, (BSOLY(K), K=1,P)                            00005280*
0405          6145 FORMAT(/1X,'(BSOLY(K), K=1,P)',10I8,2(/18X, 10I8))       00005290*
0406          6146 INET=1                                                   00005292*
0407               INIS=ISTEP                                               00005294*
0408               INSET=1                                                  00005296*
0409               ISTEP=2                                                  00005298*
0410               GO TO 6150                                               00005300*
0411          6147 ISTEP=INIS                                               00005302*
0412               INSET=0                                                  00005304*
0413          6150 GO TO 272                                                00005306
              C     ***************END IF AT THE ROOT NODE**************************  00005308
0414          6200 IF (NSX.EQ.0) GO TO 8100                                 00005310
0415          6220 IF ( IABS(STX(NSX)).GT.1000000) GO TO 6500               00005320
0416          6250 BRO=STX(NSX)                                             00005330
0417          6270 STX(NSX)=-BRO-1000000                                    00005340
0418               IF (ISTEP.EQ.0) GO TO 6308                               00005390
0419               PRINT 6305,  BRO                                         00005400
0420          6305 FORMAT('0', 'BRO  ',I10)                                 00005410
0421          6308 IF (ISTEP.NE.2) GO TO 6330                               00005420
0422               PRINT 6088, (STX(INS), INS=1,NSX)                        00005430
              C     **************MOVE TO THE NEXT (RIGHT BRANCH) NODE AND APPLY  00005490
              C                    CAPACITY RULE                            00005500
0423          6330 NOD=NOD+1                                                00005510
0424          6410 IF (ET.EQ.0.0) GO TO 6450                                00005512*
0425               IF (INSET.EQ.1) GO TO 6445                               00005516*
0426               IF (INET.EQ.1) GO TO 6450                                00005518*
0427               CALL TIMET(INT)                                          00005520*
0428               ELTN=(INT-ITO)*26.04E-6                                  00005523*
0429               IF (ELTN.LT.ET) GO TO 6450                               00005526*
```

*Relevant to the TIMET Subroutine

```
FORTRAN IV G LEVEL  21                  MAIN               DATE = 80206          14/35/07

 0430        6420 PRINT 6125, NOD, ELTN, BUB, BUBS, IBNOD                              00005528*
 0431             IBUB=BUB                                                             00005530*
 0432             IF (IBUB.EQ.9999999) GO TO 6442                                      00005532*
 0433        6430 PRINT 6135, (BSOLX(J),J=1,N)                                         00005533*
 0434        6440 PRINT 6145, (BSOLY(K), K=1,P)                                        00005536*
 0435        6442 INET=1                                                               00005538*
 0436             INIS=ISTEP                                                           00005540*
 0437             INSET=1                                                              00005542*
 0438             ISTEP=2                                                              00005544*
 0439             GO TO 6450                                                           00005546*
 0440        6445 ISTEP=INIS                                                           00005548*
 0441             INSET=0                                                              00005550*
 0442        6450 GO TO 210                                                            00005552
 0443        6500 IF ( STX(NSX).GT.1000000) GO TO 6520                                 00005555
 0444             LX=-STX(NSX)-1000000                                                 00005560
 0445             IX=LX/100                                                            00005570
 0446             JX=LX-IX*100                                                         00005580
 0447             CX(IX,JX)=0                                                          00005590
 0448             KT2(JX)=KT2(JX)-1                                                    00005595
 0449             GO TO 6550                                                           00005600
 0450        6520 LX= STX(NSX)-1000000                                                 00005610
 0451             IX=LX/100                                                            00005620
 0452             JX=LX-IX*100                                                         00005630
 0453             CX(IX,JX)=0                                                          00005640
 0454             FIX(JX)=0                                                            00005650
 0455             LQ1=LQ1-1                                                            00005660
 0456        6550 NSX=NSX-1                                                            00005690
 0457             GO TO 6200                                                           00005700
           C      **************PRINT THE OUTPUT*********************************     00005730
 0458        8100 IBUB=BUB                                                             00005740
 0459             CALL TIMET(IT1)                                                      00005750*
 0460             ELT1=(IT1-IT0)*26.04E-6                                              00005760*
 0461             PRINT 8105, ELT1                                                     00005770*
 0462        8105 FORMAT ('0',///1X, 'ELAPSED TIME IN SECONDS=', F15.8)                00005780*
 0463             PRINT 8120, NOD                                                      00005790
 0464        8120 FORMAT ( '0','TOTAL NUMBER OF NODES EXPLORED =',I3)                  00005800
 0465             IF (IBUB.EQ.9999999) GO TO 8350                                      00005810
 0466        8130 PRINT 8150                                                           00005820
 0467        8150 FORMAT ('0', 'NOTE: 1.  FOLLOWING X(I,J) VARIABLES SHOW DESIGN',     00005830
                 1          ' I TO WHICH ACTIVITY J IS ASSIGNED FOR J=1 TO N.',        00005840
                 2          /7X, '2.  IF EPSILON EPS WAS ASSIGNED A POSITIVE',         00005850
                 3          ' (NON-ZERO) VALUE, THE SOLUTION MAY BE SUBOPTIMAL.',/)    00005860
 0468        8180 PRINT 8200, (BSOLX(J),J=1,N)                                         00005870
 0469        8200 FORMAT('0',T55,  'OPTIMAL SOLUTION',/1X,T55,                         00005880
                 1          '----------------',//1X, 'X(I,J) WITH VALUE 1:',10I8,      00005890
                 2          3(/,21X,10I8))                                             00005900
 0470        8220 PRINT 8250, (BSOLY(K), K=1,P)                                        00005910
 0471        8250 FORMAT ('0',      'Y(K) VALUES:', 8X, 10I8, 2(/,21X,10I8))           00005920
 0472        8280 PRINT 8300,IBUB                                                      00005930
 0473        8300 FORMAT ('0',  'OPTIMAL VALUE OF OBJECTIVE FUNCTION:', I15///)        00005940
 0474             GO TO 8500                                                           00005950
 0475        8350 PRINT 8400                                                           00005960
 0476        8400 FORMAT ('0', ' PROBLEM DOES NOT HAVE A FEASIBLE SOLUTION',           00005970
                 1          /1X,     ' BECAUSE THE CAPACITY CONSTRAINTS CANNOT',       00005980
                 2          /1X, ' BE SATISFIED.',/)                                   00005990
 0477        8500 PRINT 8550                                                           00006000
 0478        8550 FORMAT ('0','*****NORMAL END OF JOB*****',/)                         00006010
 0479        8600 STOP                                                                 00006020
 0480             END                                                                  00006030
```

*Relevant to the TIMET Subroutine

APPENDIX B

DICTIONARY OF ZIPCAP SYMBOLIC NAMES

SYMBOLIC NAME	DEFINITION
A(I,J)	VARIABLE COST OF ACTIVITY J USING DESIGN I
B(K)	FIXED COST OF FACILITY K
BR0	BRANCHING VARIABLE WITH VALUE 0, I.E., RIGHT-BRANCH VARIABLE X(I,J)
BR1	BRANCHING VARIABLE WITH VALUE 1, I.E., LEFT-BRANCH VARIABLE X(I,J)
BSOLX(J)	CURRENT BEST SOLUTION CONTAINING X(I,J) VARIABLES
BSOLY(K)	CURRENT BEST SOLUTION CONTAINING Y(K) VARIABLES
BSUM	B(K) * D(I, J, K)/S(K) SUMMED OVER APPLICABLE K
BUB	BEST UPPER BOUND
BUBS	BEST UPPER BOUND IF SOLUTION MAY BE SUBOPTIMAL WITHIN EPS. BUBS = BUB/(1 + EPS)
C(I,J)	COST ELEMENTS FOR THE RELAXED PROBLEM
CX(I,J)	CONTAINS FIXED AND FREE X(I,J) VARIABLES. FIXED VARIABLES ARE ASSIGNED A VALUE OF 1 OR 2, AND FREE VARIABLES ARE ASSIGNED A VALUE 0. A VALUE OF 1 IMPLIES THAT THAT PARTICULAR X(I,J) HAS A VALUE 1, AND FIX(J) IS FIXED AS 1. A VALUE OF 2 IMPLIES THAT THAT PARTICULAR VARIABLE SHOULD NOT BE CONSIDERED FOR CURRENT COMPUTATIONS CX(I,J) IS UPDATED BY THE CAPACITY RULE, THE BOUNDING RULE, AND THE RULES FOR BRANCHING AND BACKTRACKING
D(I,J,K)	CAPACITY REQUIRED AT FACILITY K FOR ACTIVITY J WHEN ACTIVITY J USES DESIGN I
DBOUND	DIFFERENCE BETWEEN BOUNDS BUBS AND LOWB
DIFBR(J)	DIFFERENCE IN C(I,J) VALUES FOR SELECTING BRANCHING VARIABLE
E(I,K)	HAS VALUE 1 IF DESIGN I USES FACILITY K, OTHERWISE THE VALUE IS 0
ELT1	ELAPSED TIME IN SECONDS TO EXECUTE THE PROGRAM OBTAINED FROM THE TIMET SUBROUTINE

ELTN	ELAPSED TIME IN SECONDS TO COMPARE WITH ET IF SPECIFIED
EPS	EPSILON VALUE IF A SUBOPTIMAL SOLUTION ACCEPTABLE, E.G, EPS OF 0.002 IMPLIES THAT THE SOLUTION MAY BE SUBOPTIMAL, BUT IS GUARANTEED TO HAVE A VALUE WITHIN 0.2% OF THE OPTIMAL VALUE. EPS IS 0 IF AN OPTIMAL SOLUTION IS DESIRED
ET	ELAPSED TIME, IF SPECIFIED, AT WHICH THE NODE AND BOUNDS INFORMATION IS PRINTED
FC	FIXED COST FOR COMPUTING LOWER BOUND
FCUB	FIXED COST FOR COMPUTING UPPER BOUND
FIX(J)	VECTOR OF FIXED COLUMNS. A COLUMN IS FIXED IF IT HAS A FIXED X(I,J) VARIABLE WITH VALUE 1
FIXI(J)	VECTOR SPECIFYING INDEX I CORRESPONDING TO FIX(J) OF VALUE 1
FLB(K)	VECTOR OF FACILITIES FOR COMPUTING LOWER BOUND
FUB(K)	VECTOR OF FACILITIES FOR COMPUTING UPPER BOUND
I	INDEX FOR DESIGNS
IBALD	BALANCE AVAILABLE CAPACITY FOR A GIVEN FACILITY
IBNOD	NODE CORRESPONDING TO THE BEST UPPER BOUND
IBUB	BEST UPPER BOUND
ICAPR	OPTION TO USE CAPACITY RULE
IDIFD	DIFFERENCE BETWEEN A D(I,J,K) VALUE AND MIND(J)
IINPT	OPTION TO LIST INPUT DATA
IMIN	MINIMUM OVER INDEX I FOR A GIVEN COLUMN J
INET	INDICATOR FOR ET
INIS	HAS THE SAME VALUE AS ISTEP
INS	INDEX FOR STACK OF VARIABLES STX
INSET	INDEX FOR DISPLAYING DETAILED STEPS IF OPTION ET IS SPECIFIED
INT	INTERMEDIATE CALL TO TIMET SUBROUTINE

ISTEP	OPTION TO LIST SUMMARY OF INTERMEDIATE STEPS, DETAILED STEPS, OR TO SKIP THE LISTING
IT0	INITIAL CALL TO TIMET SUBROUTINE
IT1	FINAL CALL TO TIMET SUBROUTINE
IUNCAP	OPTION TO INDICATE IF THE PROBLEM BEING SOLVED IS CAPACITATED OR UNCAPACITATED
IX	INDEX I OF VARIABLE X(I,J)
J	INDEX FOR ACTIVITIES
JX	INDEX J OF VARIABLE X(I,J)
K	INDEX FOR FACILITIES
KT2(J)	A COUNTER FOR CX(I,J) VALUES SET EQUAL TO 2 FOR COLUMN J
LK	A COUNTER
LOWB	LOWER BOUND
LQ1	A COUNTER FOR CX(I,J) VALUES SET EQUAL TO 1
LQ2	A COUNTER FOR CX(I,J) VALUES SET EQUAL TO 2 IN A CYCLE IN WHICH ALL THE FACILITIES ARE EXAMINED
LR2	A COUNTER SET EQUAL TO LQ2 AT THE END OF A CYCLE WHEN APPLYING THE CAPACITY RULE
P	NUMBER OF FACILITIES
S(K)	TOTAL CAPACITY OF FACILITY K
SOLX(J)	SOLUTION SHOWING INDEX 1 OF X(I,J) VARIABLES CORRESPONDING TO J=1, 2, ..., N
STX(INS)	STACK OF FIXED X(I,J) VARIABLES FOR BRANCHING AND BACKTRACKING. AN X(I,J) RECORDED IN CX(I,J) AS 1 DUE TO THE BRANCHING RULE IS RECORDED IN STX(INS) AS X * 100 + J . AN X(I,J) RECORDED IN CX(I,J) AS 1 DUE TO THE CAPACITY RULE OR THE BOUNDING RULE IS RECORDED IN STX(INS) AS (X * 100 + J) + 1,000,000. AN X(I,J) RECORDED IN CX(I,J) AS 2 IS RECORDED IN STX(INS) AS -(X * 100 + J) - 1,000,000
TIMET	A SUBROUTINE FROM THE PL1 LIBRARY TO RECORD THE EXECUTION TIME

UPB	UPPER BOUND
ZIPCAP	NAME OF THE PROGRAM, AN ACRONYM FOR ZERO-ONE INTEGER PROGRAM TO SOLVE MULTIACTIVITY MULTIFACILITY CAPACITY-CONSTRAINED ASSIGNMENT PROBLEMS

APPENDIX C

ANNOTATED OUTPUT FOR TEST PROBLEM 1

OPTIONS SELECTED : IINPT=1 ICAPR=1 ISTEP=1 IUNCAP=0 EPS= 0.0 ET= 0.0 (a)

INPUT DATA

NUMBER OF DESIGNS (M)= 3

NUMBER OF ACTIVITIES (N)= 4 (b)

NUMBER OF FACILITIES (P)= 5

VARIABLE COST MATRIX A(I,J)

I= 1	9847	15718	4450	8925
I= 2	10192	10542	4422	8118
I= 3	11019	9750	4609	8337

FIXED COST VECTOR B(K)

1750	2000	1750	1350	1000

CAPACITY LIMIT VECTOR S(K) (c)

400	400	1000	400	400

CAPACITY USAGE MATRIX D(I,J,K)

K= 1

I= 1	0	0	0	0
I= 2	186	60	261	148
I= 3	154	38	175	100

K= 2

I= 1	0	0	0	0
I= 2	0	0	0	0
I= 3	154	38	175	100

K= 3

I= 1	525	144	1011	396
I= 2	186	60	261	148
I= 3	154	38	175	100

K= 4

I= 1	0	0	0	0
I= 2	0	0	0	0
I= 3	154	38	175	100

K= 5

I= 1	0	0	0	0
I= 2	186	60	261	148
I= 3	154	38	175	100

(c)

DESIGN-FACILITY MATRIX E(I,K)

I= 1	0	0	1	0	0
I= 2	1	0	1	0	1
I= 3	1	1	1	1	1

SUMMARY OF STEPS

MINSC, FC, LOWB 37229.3594 0 37229.3594

BR1 101

(d)

```
    NODE NUMBER    2

 MINSC, FC, LOWB      35528.3633          1750     37278.3633
 BR1       203

    NODE NUMBER    3

 MINSC, FC, LOWB      32356.0000          7850     40206.0000
NSUMA, FCUB, UPB, BUB, BUBS      32356      7850     40206.0000    9999999.0000    9999999.0000
BRO         203                                                                              (d)

    NODE NUMBER    4

 MINSC, FC, LOWB      32324.0000          7850     40174.0000
NSUMA, FCUB, UPB, BUB, BUBS      32324      7850     40174.0000     40206.0000     40206.0000
BRO         101

    NODE NUMBER    5

 MINSC, FC, LOWB      33476.0000          7850     41326.0000


ELAPSED TIME IN SECONDS=     0.06226163                                                      (e)
TOTAL NUMBER OF NODES EXPLORED =  5                                                          (f)
NOTE: 1.  FOLLOWING X(I,J) VARIABLES SHOW DESIGN I TO WHICH ACTIVITY J IS ASSIGNED FOR J=1 TO N.
      2.  IF EPSILON EPS WAS ASSIGNED A POSITIVE (NON-ZERO) VALUE, THE SOLUTION MAY BE SUBOPTIMAL.

                                   OPTIMAL SOLUTION
                                   ----------------
X(I,J) WITH VALUE 1:       1       3       3       2                                         (g)
Y(K) VALUES:               1       1       1       1       1
OPTIMAL VALUE OF OBJECTIVE FUNCTION:          40174


*****NORMAL END OF JOB*****
```

APPENDIX D

ANNOTATED OUTPUT FOR TEST PROBLEM 2

OPTIONS SELECTED : IINPT=1 ICAPR=1 ISTEP=0 IUNCAP=0 EPS= 0.00200 ET= 0.0 (a)

INPUT DATA

NUMBER OF DESIGNS (M)= 5

NUMBER OF ACTIVITIES (N)= 4 (b)

NUMBER OF FACILITIES (P)= 8

VARIABLE COST MATRIX A(I,J)

I= 1	194951	218871	155094	104056
I= 2	235277	272087	143264	138641
I= 3	196138	220718	160399	107481
I= 4	198751	224042	167046	112445
I= 5	190873	229169	128361	112498

FIXED COST VECTOR B(K) (c)

14000	14000	14000	14000	14000	28000	19000	31000

CAPACITY LIMIT VECTOR S(K)

350	350	350	200	200	700	500	800

CAPACITY USAGE MATRIX D(I,J,K)

K= 1

(c)

I= 1	80	90	100	180
I= 2	0	0	0	0
I= 3	80	90	100	180
I= 4	40	30	50	120
I= 5	0	0	0	0
K= 2				
I= 1	80	90	100	180
I= 2	0	0	0	0
I= 3	80	90	100	180
I= 4	40	30	50	120
I= 5	0	0	0	0
K= 3				
I= 1	80	90	100	180
I= 2	0	0	0	0
I= 3	80	90	100	180
I= 4	40	30	50	120
I= 5	0	0	0	0
K= 4				
I= 1	80	90	100	180
I= 2	0	0	0	0
I= 3	80	90	100	180
I= 4	40	30	50	120
I= 5	80	90	100	180
K= 5				
I= 1	80	90	100	180
I= 2	0	0	0	0
I= 3	80	90	100	180

I= 4	40	30	50	120
I= 5	80	90	100	180

K= 6

I= 1	240	180	300	450
I= 2	0	0	0	0
I= 3	120	90	150	360
I= 4	0	0	0	0
I= 5	40	30	50	120

K= 7

I= 1	160	180	200	360
I= 2	0	0	0	0
I= 3	80	60	100	240
I= 4	0	0	0	0
I= 5	160	180	200	360

K= 8

I= 1	200	150	250	600
I= 2	40	30	50	180
I= 3	0	0	0	0
I= 4	0	0	0	0
I= 5	120	90	150	300

DESIGN-FACILITY MATRIX E(I,K)

I= 1	1	1	1	1	1	1	1	1
I= 2	0	0	0	0	0	0	0	1
I= 3	1	1	1	1	1	1	1	0
I= 4	1	1	1	1	1	0	0	0

```
I= 5          0        0        0        1        1        1        1        1

ELAPSED TIME IN SECONDS=     0.09392625                                        (e)

TOTAL NUMBER OF NODES EXPLORED = 11                                            (f)

NOTE: 1.  FOLLOWING X(I,J) VARIABLES SHOW DESIGN I TO WHICH ACTIVITY J IS ASSIGNED FOR J=1 TO N.
      2.  IF EPSILON EPS WAS ASSIGNED A POSITIVE (NON-ZERO) VALUE, THE SOLUTION MAY BE SUBOPTIMAL.

                                 OPTIMAL SOLUTION

X(I,J) WITH VALUE 1:      4     4     2     4

Y(K) VALUES:              1     1     1     1     1     0     0     1           (g)

OPTIMAL VALUE OF OBJECTIVE FUNCTION:       779502


*****NORMAL END OF JOB*****
```

APPENDIX E

ANNOTATED OUTPUT FOR TEST PROBLEM 3

OPTIONS SELECTED : IINPT=1 ICAPR=0 ISTEP=0 IUNCAP=1 EPS= 0.0 ET= 30.000

(a)

INPUT DATA

NUMBER OF DESIGNS (M)= 10

NUMBER OF ACTIVITIES (N)= 8

NUMBER OF FACILITIES (P)= 8

(b)

VARIABLE COST MATRIX A(I,J)

I= 1	21	13	20	8	15	10	35	20
I= 2	26	13	27	14	17	13	40	25
I= 3	25	12	26	13	16	12	20	23
I= 4	23	9	22	20	19	20	50	19
I= 5	23	20	24	13	20	15	40	17
I= 6	23	20	24	13	20	15	40	17
I= 7	26	13	26	20	15	15	35	20
I= 8	30	20	40	23	30	20	55	25
I= 9	26	13	26	20	20	15	40	15
I= 10	36	46	46	50	19	20	40	20

(c)

FIXED COST VECTOR B(K)

10	8	4	10	14	30	20	50

CAPACITY LIMIT VECTOR S(K)

8	8	8	8	8	8	8	8

CAPACITY USAGE MATRIX D(I,J,K)

K= 1								
I= 1	1	1	1	1	1	1	1	1
I= 2	1	1	1	1	1	1	1	1
I= 3	1	1	1	1	1	1	1	1
I= 4	1	1	1	1	1	1	1	1
I= 5	1	1	1	1	1	1	1	1
I= 6	1	1	1	1	1	1	1	1
I= 7	1	1	1	1	1	1	1	1
I= 8	0	0	0	0	0	0	0	0
I= 9	1	1	1	1	1	1	1	1
I= 10	1	1	1	1	1	1	1	1
K= 2								
I= 1	1	1	1	1	1	1	1	1
I= 2	1	1	1	1	1	1	1	1
I= 3	1	1	1	1	1	1	1	1
I= 4	1	1	1	1	1	1	1	1

C

c

I= 5	1	1	1	1	1	1	1	1
I= 6	1	1	1	1	1	1	1	1
I= 7	1	1	1	1	1	1	1	1
I= 8	0	0	0	0	0	0	0	0
I= 9	1	1	1	1	1	1	1	1
I= 10	1	1	1	1	1	1	1	1
K= 3								
I= 1	1	1	1	1	1	1	1	1
I= 2	1	1	1	1	1	1	1	1
I= 3	1	1	1	1	1	1	1	1
I= 4	0	0	0	0	0	0	0	0
I= 5	1	1	1	1	1	1	1	1
I= 6	1	1	1	1	1	1	1	1
I= 7	1	1	1	1	1	1	1	1
I= 8	0	0	0	0	0	0	0	0
I= 9	1	1	1	1	1	1	1	1
I= 10	1	1	1	1	1	1	1	1
K= 4								
I= 1	1	1	1	1	1	1	1	1
I= 2	1	1	1	1	1	1	1	1
I= 3	0	0	0	0	0	0	0	0

c

I= 4	1	1	1	1	1	1	1	1
I= 5	0	0	0	0	0	0	0	0
I= 6	0	0	0	0	0	0	0	0
I= 7	1	1	1	1	1	1	1	1
I= 8	0	0	0	0	0	0	0	0
I= 9	1	1	1	1	1	1	1	1
I= 10	1	1	1	1	1	1	1	1
K= 5								
I= 1	1	1	1	1	1	1	1	1
I= 2	1	1	1	1	1	1	1	1
I= 3	1	1	1	1	1	1	1	1
I= 4	0	0	0	0	0	0	0	0
I= 5	0	0	0	0	0	0	0	0
I= 6	0	0	0	0	0	0	0	0
I= 7	1	1	1	1	1	1	1	1
I= 8	0	0	0	0	0	0	0	0
I= 9	1	1	1	1	1	1	1	1
I= 10	1	1	1	1	1	1	1	1
K= 6								
I= 1	1	1	1	1	1	1	1	1

(c)

I= 2	1	1	1	1	1	1	1	1
I= 3	1	1	1	1	1	1	1	1
I= 4	1	1	1	1	1	1	1	1
I= 5	1	1	1	1	1	1	1	1
I= 6	1	1	1	1	1	1	1	1
I= 7	0	0	0	0	0	0	0	0
I= 8	0	0	0	0	0	0	0	0
I= 9	1	1	1	1	1	1	1	1
I= 10	0	0	0	0	0	0	0	0

K= 7

I= 1	1	1	1	1	1	1	1	1
I= 2	0	0	0	0	0	0	0	0
I= 3	0	0	0	0	0	0	0	0
I= 4	1	1	1	1	1	1	1	1
I= 5	1	1	1	1	1	1	1	1
I= 6	0	0	0	0	0	0	0	0
I= 7	0	0	0	0	0	0	0	0
I= 8	0	0	0	0	0	0	0	0
I= 9	1	1	1	1	1	1	1	1
I= 10	0	0	0	0	0	0	0	0

K= 8

I= 1	1	1	1	1	1	1	1	1
I= 2	1	1	1	1	1	1	1	1
I= 3	1	1	1	1	1	1	1	1
I= 4	1	1	1	1	1	1	1	1
I= 5	1	1	1	1	1	1	1	1
I= 6	1	1	1	1	1	1	1	1
I= 7	1	1	1	1	1	1	1	1
I= 8	1	1	1	1	1	1	1	1
I= 9	0	0	0	0	0	0	0	0
I= 10	0	0	0	0	0	0	0	0

DESIGN-FACILITY MATRIX E(I,K)

I= 1	1	1	1	1	1	1	1	1
I= 2	1	1	1	1	1	1	0	1
I= 3	1	1	1	0	1	1	0	1
I= 4	1	1	0	1	0	1	1	1
I= 5	1	1	1	0	0	1	1	1
I= 6	1	1	1	0	0	1	0	1
I= 7	1	1	1	1	1	0	0	1
I= 8	0	0	0	0	0	0	0	1
I= 9	1	1	1	1	1	1	1	0
I= 10	1	1	1	1	1	0	0	0

```
ELAPSED TIME IN SECONDS=     0.57446837                                                    e
TOTAL NUMBER OF NODES EXPLORED = 35                                                        f

NOTE: 1.  FOLLOWING X(I,J) VARIABLES SHOW DESIGN I TO WHICH ACTIVITY J IS ASSIGNED FOR J=1 TO N.
      2.  IF EPSILON EPS WAS ASSIGNED A POSITIVE (NON-ZERO) VALUE, THE SOLUTION MAY BE SUBOPTIMAL.

                                        OPTIMAL SOLUTION
                                        ----------------

X(I,J) WITH VALUF 1:        6      3      6      3      3      3      3      6                g

Y(K) VALUES:                1      1      1      0      1      1      0      1

OPTIMAL VALUE OF OBJECTIVE FUNCTION:          253


*****NORMAL END OF JOB*****
```

THE GEORGE WASHINGTON UNIVERSITY
Program in Logistics
Distribution List for Technical Papers

The George Washington University
   Office of Sponsored Research
   Library
   Vice President H. F. Bright
   Dean Harold Liebowitz
   Dean Henry Solomon

ONR
   Chief of Naval Research
      (Codes 200, 434)
   Resident Representative

OPNAV
   OP-40
   DCNO, Logistics
   Navy Dept Library
   NAVDATA Automation Cmd
   OP-964

Naval Aviation Integrated Log Support

NARDAC Tech Library

Naval Electronics Lab Library

Naval Facilities Eng Cmd Tech Library

Naval Ordnance Station
   Louisville, Ky.
   Indian Head, Md.

Naval Ordnance Sys Cmd Library

Naval Research Branch Office
   Boston
   Chicago
   New York
   Pasadena
   San Francisco

Naval Ship Eng Center
   Philadelphia, Pa.
   Washington, DC

Naval Ship Res & Dev Center

Naval Sea Systems Command
   PMS 30611
   Tech Library
   Code 073

Naval Supply Systems Command
   Library
   Operations and Inventory Analysis

Naval War College Library
   Newport

BUPERS Tech Library

FMSO

Integrated Sea Lift Study

USN Ammo Depot Earle

USN Postgrad School Monterey
   Library
   Dr Jack R. Borsting
   Prof C. R. Jones

US Marine Corps
   Commandant
   Deputy Chief of Staff, R&D

Marine Corps School Quantico
   Landing Force Dev Ctr
   Logistics Officer
   Commanding Officer
      USS Francis Marion (LPA-249)

Armed Forces Industrial College

Armed Forces Staff College

Army War College Library
   Carlisle Barracks

Army Cmd & Gen Staff College

Army Logistics Mgt Center
   Fort Lee

Commanding Officer, USALDSRA
   New Cumberland Army Depot

Army Inventory Res Ofc
   Philadelphia

Air Force Headquarters
   AFADS-3
   LEXY
   SAF/ALG

Griffiss Air Force Base
   Reliability Analysis Center

Gunter Air Force Base
   AFLMC/XR

Maxwell Air Force Base Library

Wright-Patterson Air Force Base
   Log Command
   Research Sch Log
   AFALD/XR

Defense Documentation Center

National Academy of Sciences
   Maritime Transportation Res Board Library

National Bureau of Standards
   Dr B. H. Colvin
   Dr Joan Rosenblatt

National Science Foundation

National Security Agency

Weapon Systems Evaluation Group

British Navy Staff

National Defense Hdqtrs, Ottawa
   Logistics, OR Analysis Establishment

American Power Jet Co
   George Chernowitz

General Dynamics, Pomona

General Research Corp
   Dr Hugh Cole
   Library

Logistics Management Institute
   Dr Murray A. Geisler

MATHTEC
   Dr Eliot Feldman

Rand Corporation
   Library

Carnegie-Mellon University
   Dean H. A. Simon
   Prof G. Thompson

Case Western Reserve University
   Prof B. V. Dean
   Prof M. Mesarovic
   Prof S. Zacks

Cornell University
   Prof R. E. Bechhofer
   Prof R. W. Conway
   Prof Andrew Schultz, Jr.

Cowles Foundation for Research in Economics
   Prof Herbert Scarf
   Prof Martin Shubik

Florida State University
   Prof R. A. Bradley

Harvard University
   Prof K. J. Arrow
   Prof W. G. Cochran
   Prof Arthur Schleifer, Jr.

Princeton University
   Prof A. W. Tucker
   Prof J. W. Tukey
   Prof Geoffrey S. Watson

Purdue University
Prof S. S. Gupta
Prof H. Rubin
Prof Andrew Whinston

Stanford University
Prof T. W. Anderson
Prof G. B. Dantzig
Prof F. S. Hillier
Prof D. L. Iglehart
Prof Samuel Karlin
Prof G. J. Lieberman
Prof Herbert Solomon
Prof A. F. Veinott, Jr.

University of California, Berkeley
Prof R. E. Barlow
Prof D. Gale
Prof Jack Kiefer
Prof Rosedith Sitgreaves

University of California, Los Angeles
Prof J. R. Jackson
Prof R. R. O'Neill

University of North Carolina
Prof W. L. Smith
Prof M. R. Leadbetter

University of Pennsylvania
Prof Russell Ackoff
Prof Thomas L. Saaty

University of Texas
Prof A. Charnes

Yale University
Prof F. J. Anscombe
Prof I. R. Savage

Prof Z. W. Birnbaum
University of Washington

Prof B. H. Bissinger
The Pennsylvania State University

Prof Seth Bonder
University of Michigan

Prof G. E. P. Box
University of Wisconsin

Dr Jerome Bracken
Institute for Defense Analyses

Prof H. Chernoff
Mass. Institute of Technology

Prof Arthur Cohen
Rutgers - The State University

Mr Wallace M. Cohen
US General Accounting Office

Prof C. Derman
Columbia University

Prof Masao Fukushima
Kyoto University

Prof Saul I. Gass
University of Maryland

Dr Donald P. Gaver
Carmel, California

Prof Amrit L. Goel
Syracuse University

Prof J. F. Hannan
Michigan State University

Prof H. O. Hartley
Texas A & M Foundation

Mr Gerald F. Hein
NASA, Lewis Research Center

Prof W. M. Hirsch
Courant Institute

Dr Alan '. Hoffman
IBM, Yorktown Heights

Prof John R. Isbell
State University of New York, Amherst

Dr J. L. Jain
University of Delhi

Prof J. H. K. Kao
Polytech Institute of New York

Prof W. Kruskal
University of Chicago

Mr S. Kumar
University of Madras .

Prof C. E. Lemke
Rensselaer Polytech Institute

Prof Loynes
University of Sheffield, England

Prof Steven Nahmias
University of Pittsburgh

Prof D. B. Owen
Southern Methodist University

Prof E. Parzen
Texas A & M University

Prof H. O. Posten
University of Connecticut

Prof R. Remage, Jr.
University of Delaware

Prof Hans Riedwyl
University of Bern

Dr Fred Rigby
Texas Tech College

Mr David Rosenblatt
Washington, D. C.

Prof M. Rosenblatt
University of California, San Diego

Prof Alan J. Rowe
University of Southern California

Prof A. H. Rubenstein
Northwestern University

Dr M. E. Salveson
West Los Angeles

Prof Edward A. Silver
University of Waterloo, Canada

Prof M. J. Sobel
Georgia Inst of Technology

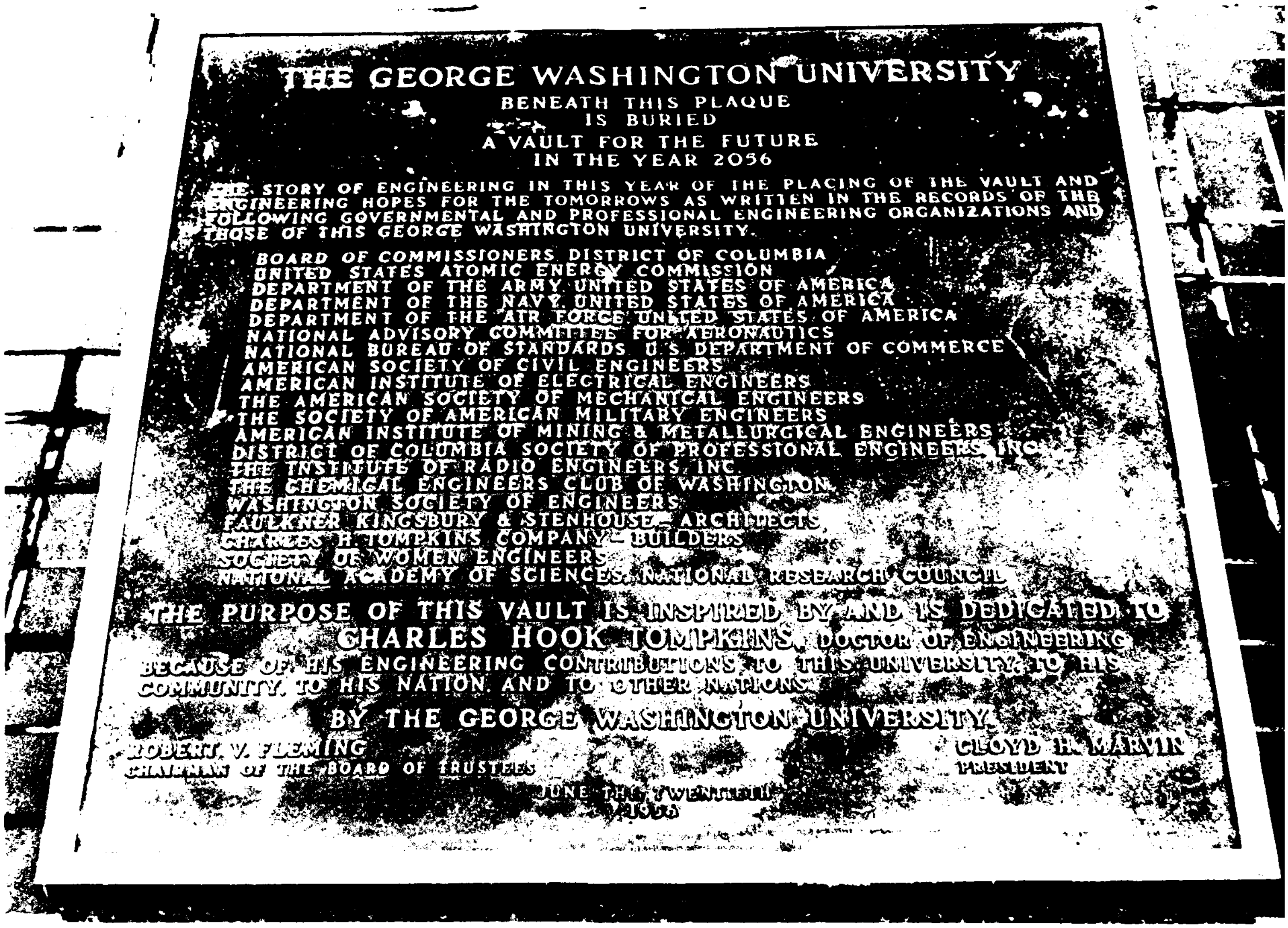
Prof R. M. Thrall
Rice University

Dr S. Vajda
University of Sussex, England

Prof T. M. Whitin
Wesleyan University

Prof Jacob Wolfowitz
University of South Florida

Prof Max A. Woodbury
Duke University



To cope with the expanding technology, our society must be assured of a continuing supply of rigorously trained and educated engineers. The School of Engineering and Applied Science is completely committed to this objective.


8