MIT
SEA
GRANT
PROGRAM

Appendix to
PORT DESIGN AND ANALYSIS METHODOLOGY

E. Frankel, Editor
With Contributions By
B. Golden
P. Wilmes
R. Orner
K. Chelst
M. Creton
E. Frankel

Massachusetts Institute of Technology

Cambridge, Massachusetts 02139

Appendix to
Report No. MITSG 74-31
December 1, 1974

Appendix to
Port Design
and
Analysis Methodology

E. Frankel, Editor
With Contributions By
B. Golden
P. Wilmes
R. Orner
K. Chelst
M. Creton
E. Frankel

Appendix to
Report No. MITSG 74-31
Index No. 74-331-Nto

TABLE OF CONTENTS

APPENDIX A

User's Manual for the Linear Approximation Algorithm

User's Manual for the Linear Approximation Algorithm

Variables in Program:

N = the number of arcs in network

M = the number of nodes in network

NN = the number of fifj terms in S(f)

MM = the number of steps considered in shortest path problem

EPS = epsilon, used as a stopping rule

JK = the number of the arc corresponding to the first port arc

FI(700) = a vector with coefficients of Fi^2 in S(f)

FI2(700) = a vector with coefficients of Fi^2 in S(f)

FIFJ(25) = a vector with coefficient of fifj n S(f), 18 fifj terms in 9 U.S. port models

FIFJ(25,2) = a matrix containing arc i, arc j in each row corresponding to the crossterms in S(f)

XK(700) = a starting vector of flows

G(700) = a vector of arc distances, arcs numbered from 1 to N

IG(700,3) = a matrix of partial derivatives of cross terms, each row i representing the partial with respect to Si. The column tells us the variable.

D(60,50) = matrix of distances between node i and node j

F(50,60) = a matrix of shortest distances between source and other nodes in K steps where L = 1,2,...,5. (There are really two sources, one for imports and one for exports.)

Pred(60) = predecision of node i vector in least cost path

Flow(700) = shortest path flow vector

Demand(40,2) = row i of the matrix gives the node # of demand center, then actual demand.

XKP1(700) = new vector of flows.

A note on program dimensions: with 4 foreign ports, 9 U.S. ports, and 30 hinterland supersource and supersink, we have 54 nodes and 698 arcs.

Sections in program:

I. Dimension cards, read statements defining the variables.

II. Calculate partial derivatives of total cost in the network.

III. This section is used during first iteration only and gives us a starting feasible flow.

IV. Evaluate partial derivatives at a point X_k which yields distances associated to each arc.

V. Initialize all distances to either zero or infinity.

VI. Now associate distance G to the ordered pair (i,j) if and only if arc ij appears on graph.

VII. Find shortest route from node i, the supersource.

VIII. For all demand centers supplied from node, we place demand on each arc contained in the shortest path.

IX. Find shortest route from node m, the supersink.

X. For all demand centers supplied from node m, we place demand on each arc contained in the shortest path.

XI. This section is used to help find initial feasible solution.

XII. Golden Section Search Technique.

XIII. Stopping Criterion.

Inputs:

The input variables are:

FI is read in as 0. 0. 5. 5. 7. 7. 6. 5. 4. 5. 8. 9. 13. 13.
11. 10. 9. 9. 0. 0. 0. 0.

FI2 is read in as 0. 0. 0. 0. 0. 0. 3. 3. 4. 4. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0.

FIFJ is read in as -1. -1. -2. -1.

IFIFJ is read in as 7 8, 7 8, 9 10, 9 10

XK is read in as 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0.

IG is read in as 1 2 1, 2 1 2, 2 3 3, 3 2 4, 2 4 5, 4 2 6, 3 5 7, 5 3 8, 4 8 9, 6 4 10, 5 7 11, 17 5 12, 7 6 13, 6 7 14, 8 5 15, 5 8 16, 6 8 17, 8 6 16, 7 9 19, 9 7 20, 8 9 21, 9 8 22

Demand is read in as 7 30, 8 36, 2 32.

As is obvious from the preceeding graph, N = 22 and m = 9. The number of crossterms in S(f), NN, is equal to the number of P - P' arcs, here 4. There are three demand centers (ND = 3) which are nodes 2, 7, and 8. An EPS value of 0.1 is used. For this six-partite class of graphs, mm will equal 5. Concerning the numbering of arcs, our policy will be to number from left to right, up to down, in general. JK will be the number of the first arc to possess crossterms (the first U.S. port arc) and arcs JK through JK + NN - 1 are P - P' arcs with crossterms.

Let it be demonstrated how inputs are handled by referring to the example which has been solved using both algorithms. The graph following is the same graph as was mentioned in Section 4.4.4.3, excepting this graph's additional node (supersource) and tow arcs. Therefore, $C_i(f)$ from the previous graph represents $C_{i+2}(f)$ on this graph. Nodes and arcs are numbered as shown in Fig. A-1.

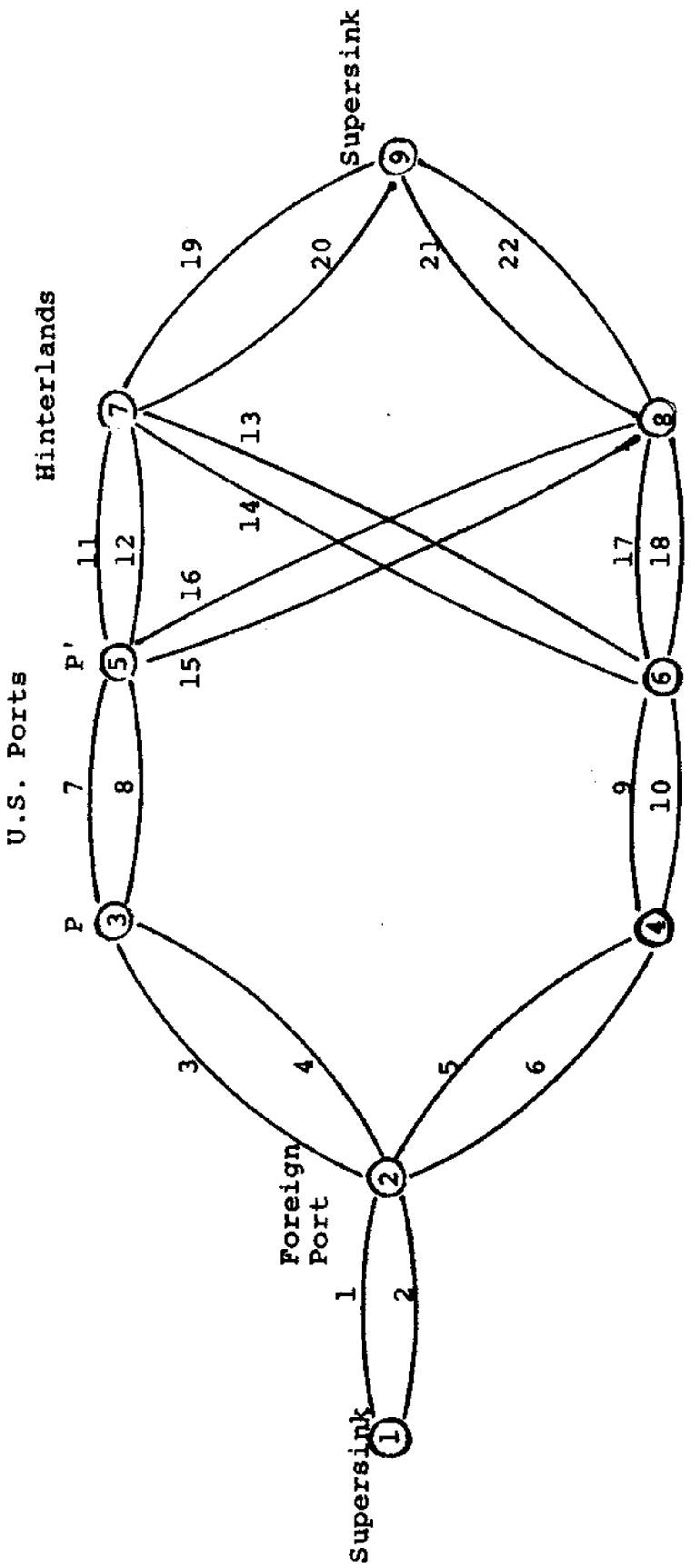
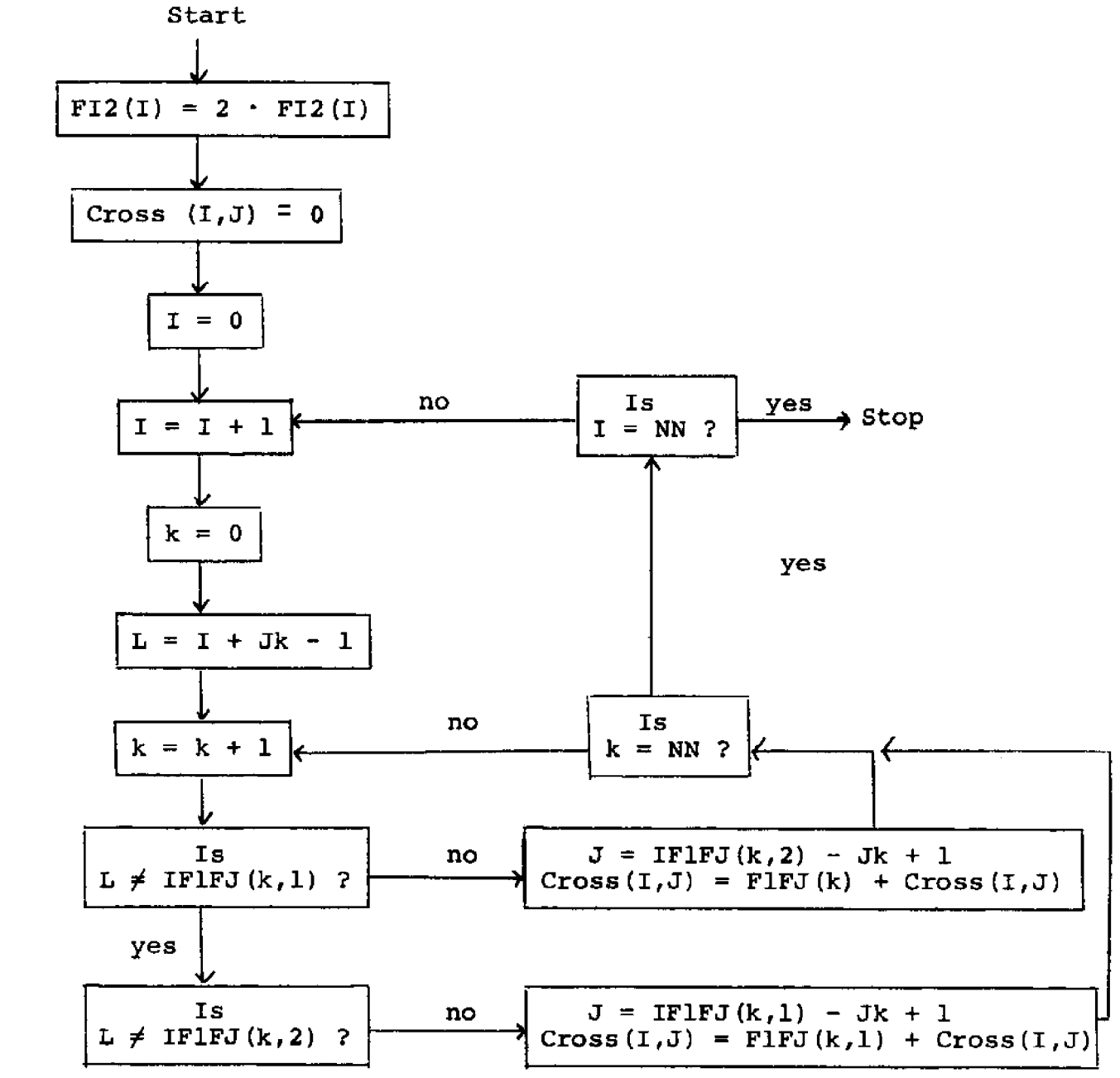
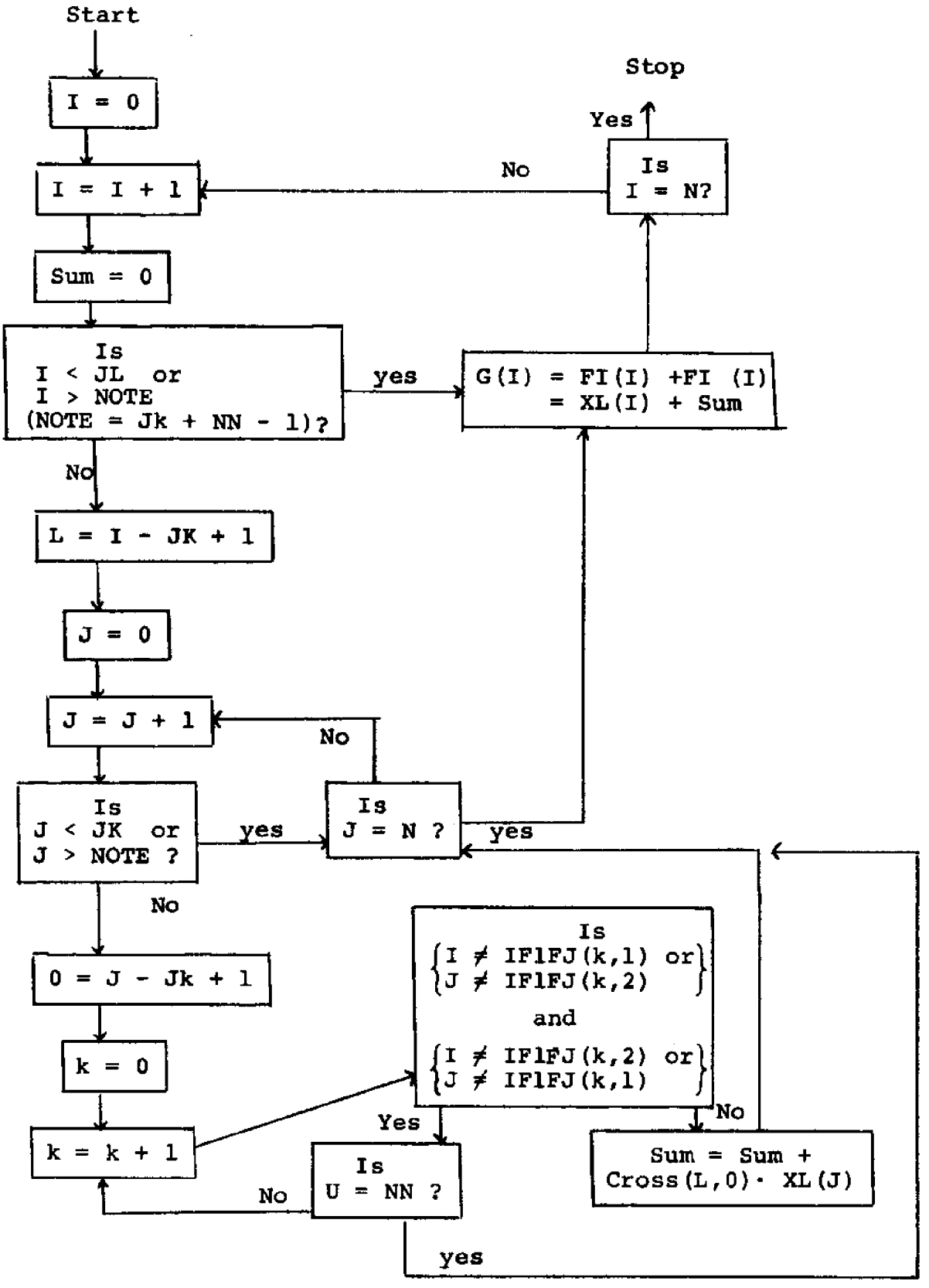


Fig.A1 Nodes and Arcs in Network

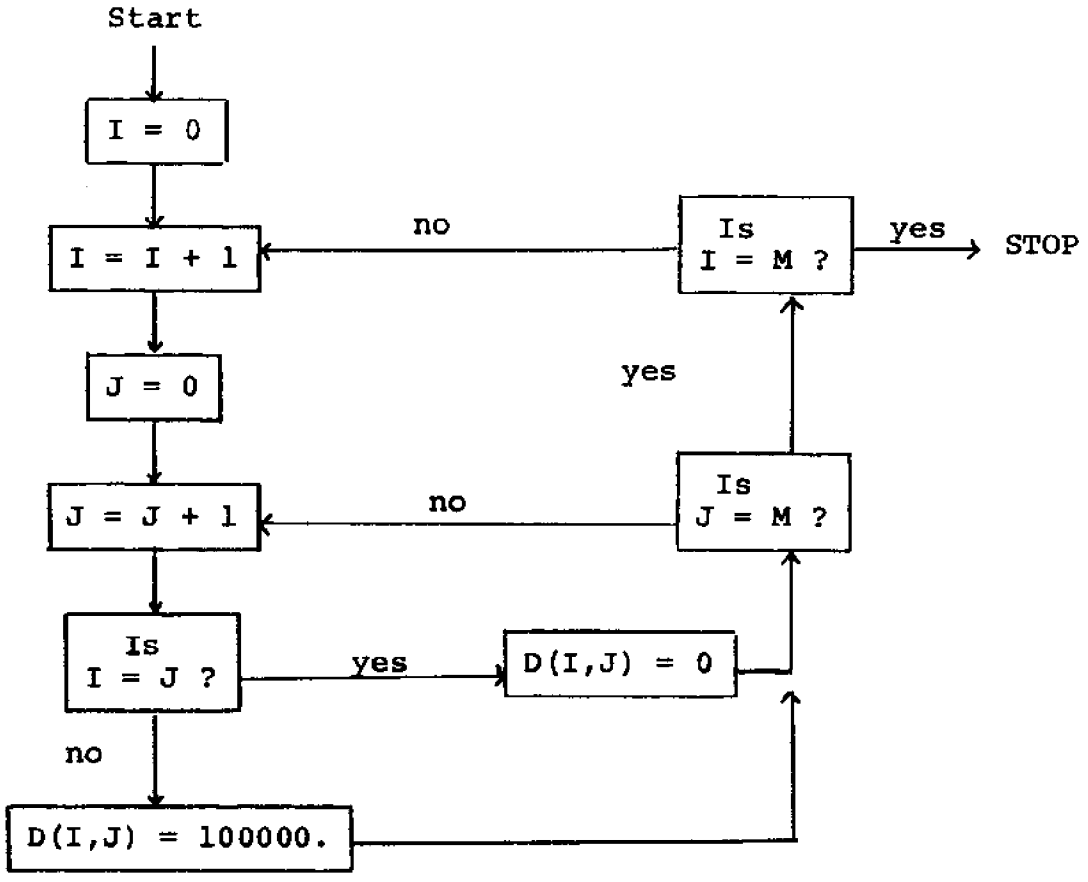
Flow Chart of Section II.



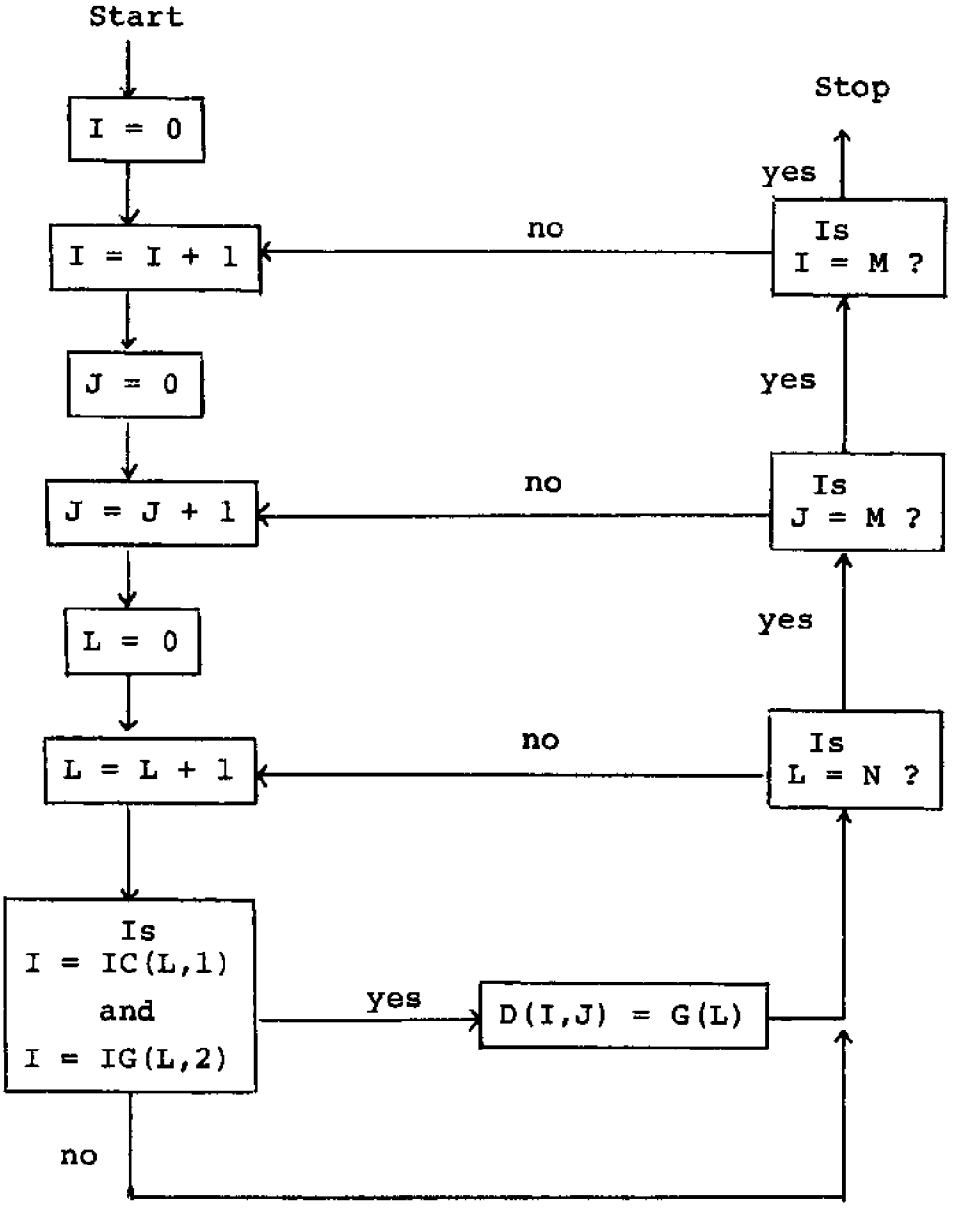
Flow Chart of Section IV.



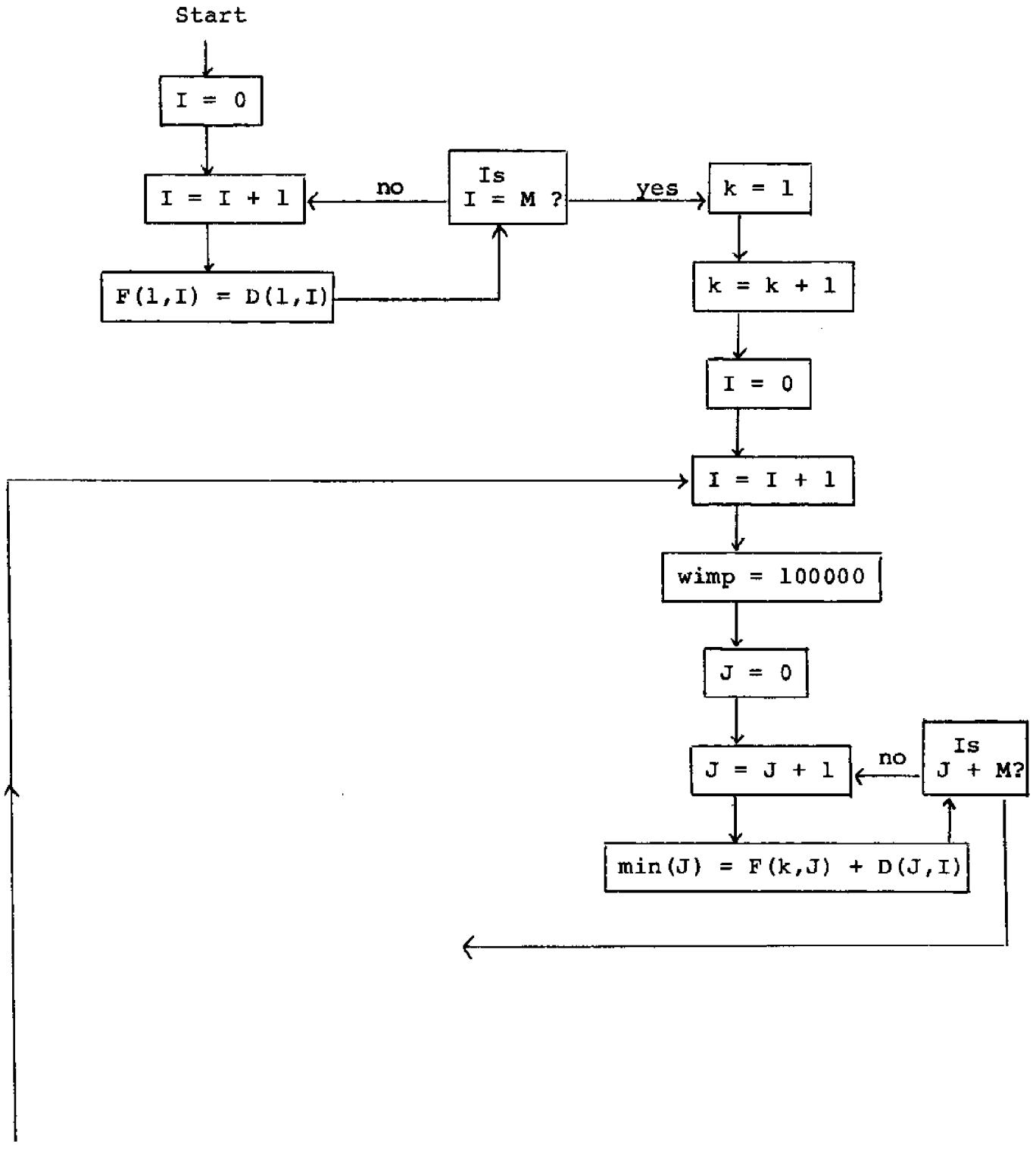
Flow Chart of Section V.



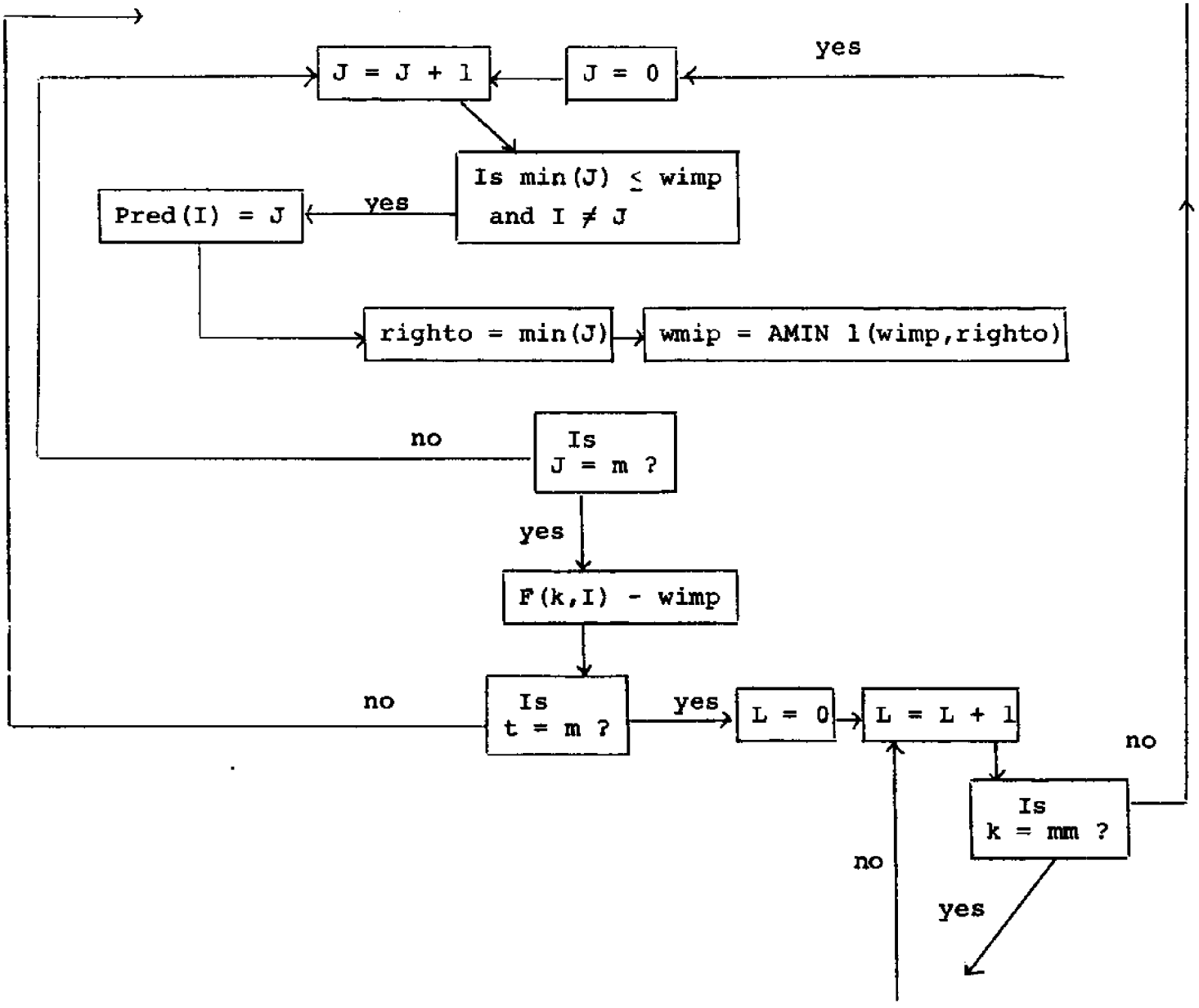
Flow Chart of Section VI.



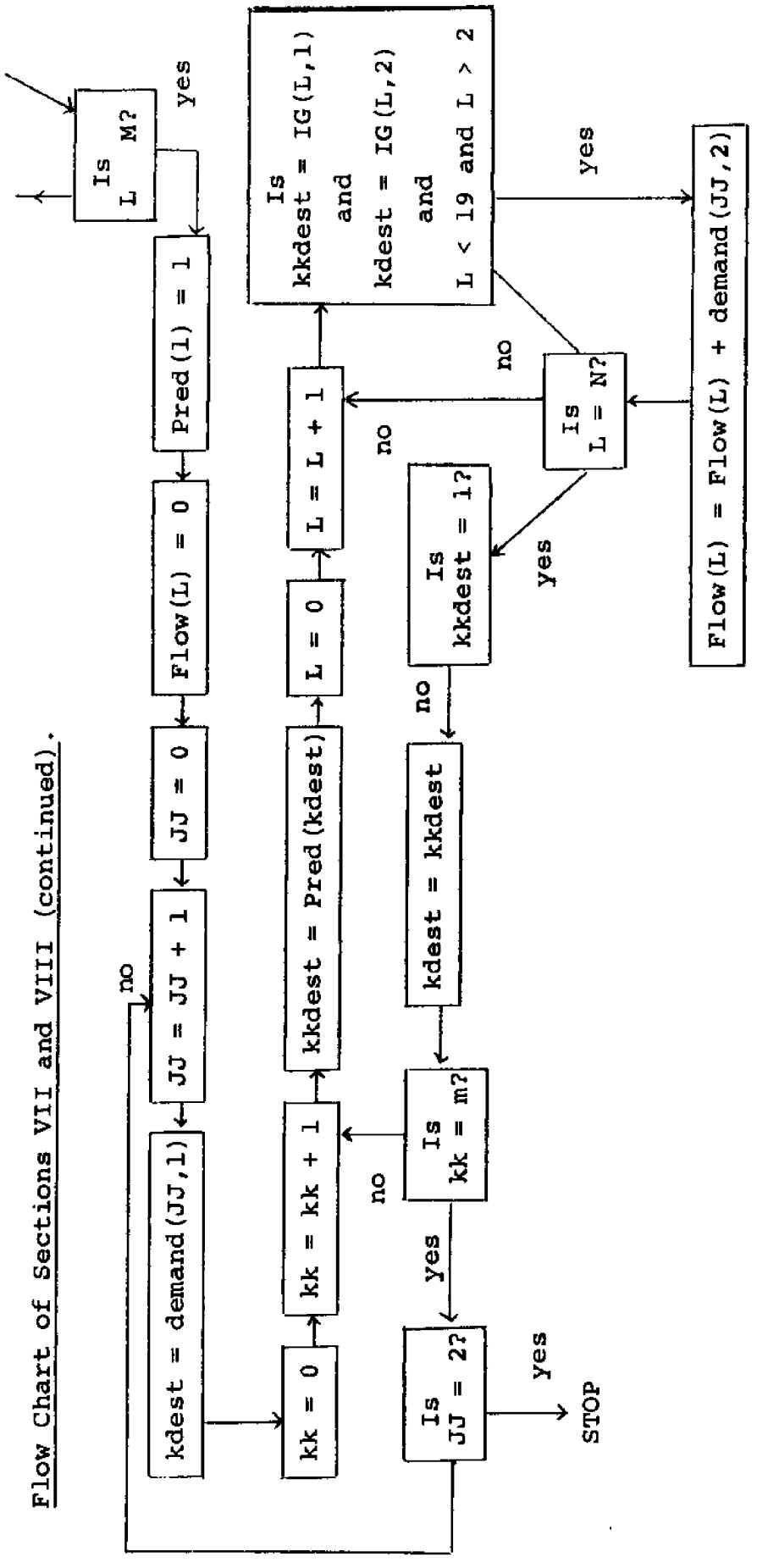
Flow Chart of Sections VII and VIII.



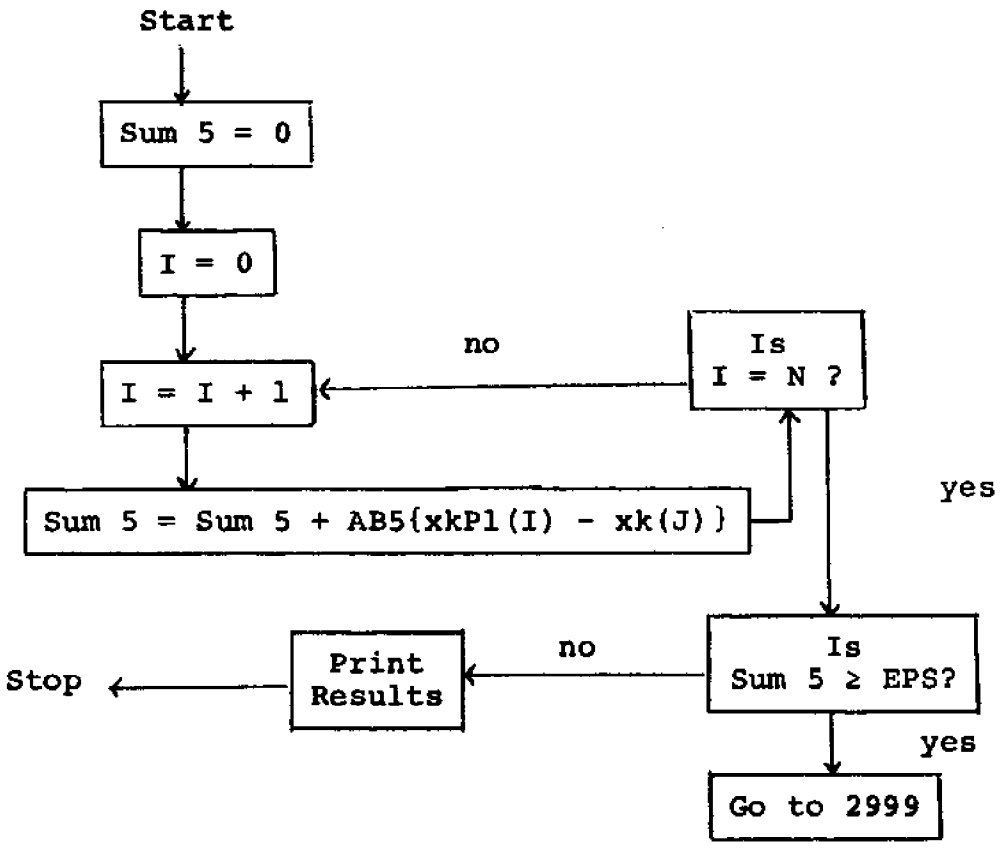
Flow Chart of Sections VII and VIII (continued).



Flow Chart of Sections VII and VIII (continued).



Flow Chart of Section XIII.



(The flow charts of Sections I, III, and XI have been omitted because they are trivial. Sections IX and X are very similar to Sections VII and VIII, and thus are also omitted.

APPENDIX B

The Ford-Bellman-Moore Shortest Path Algorithm

The Ford-Bellman-Moore Shortest Path Algorithm

The node label $f_i^{(k)}$ represents the length of the shortest path connecting node 1 and node i and that contains k + 1 or fewer arcs. The fundamental recursion is $f_i^{(k+1)} = \min_j [d_{ji} + f_j^{(k)}]$, $f_i^{(o)} \equiv d_{1i}$. In the case where all distances are positive or zero, convergence occurs wherever $f_i^{(k)} = f_i^{(k+1)}$. On the other hand, if negative distances exist either convergence occurs for $k \leq N - 1$ or a change in some f_i will occur on the (N - 1) iteration indicating a negative loop. [19]

Find minimum cost free from (a) in the graph below. Solution is darkened.

$f_i^{(k)}$ = the length of the shortest path that connects node 1 and node i and that contains k + 1 or fewer arcs.

$f_i^{(o)} = d_{1i}$

$f_i^{(k+1)} = \min_j [d_{ji} + f_j^{(k)}]$, j is right before i

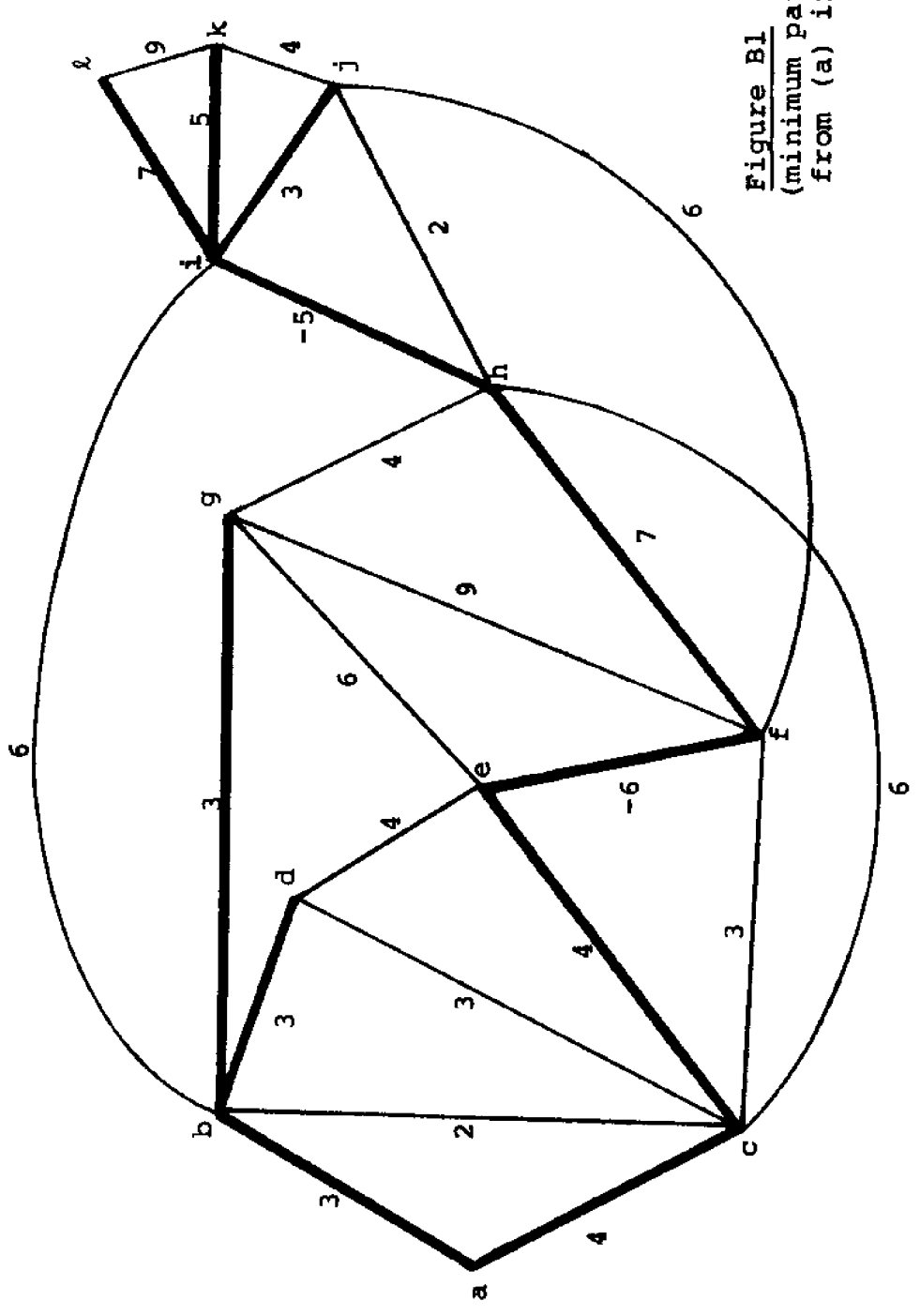


Figure B1
(minimum path free from (a) in dark)

Calculations

k		(0)	(1)	(2)	(3)	(4)	(5)	(6)
f_b	=	3	3	3	3	3	3	3
f_c	=	4	4	4	4	4	4	4
f_d	=		6	6	6	6	6	6
f_e	=		8	8	8	8	8	8
f_f	=		7	2	2	2	2	2
f_g	=		6	6	6	6	6	6
f_h	=			10	9	9	9	9
f_i	=		9	9	9	4	4	4
f_j	=			12	12	12	7	7
f_k	=			14	14	14	9	9
f_ℓ	=			16	16	16	11	11

Therefore, there are no negative loops, convergence has occurred.

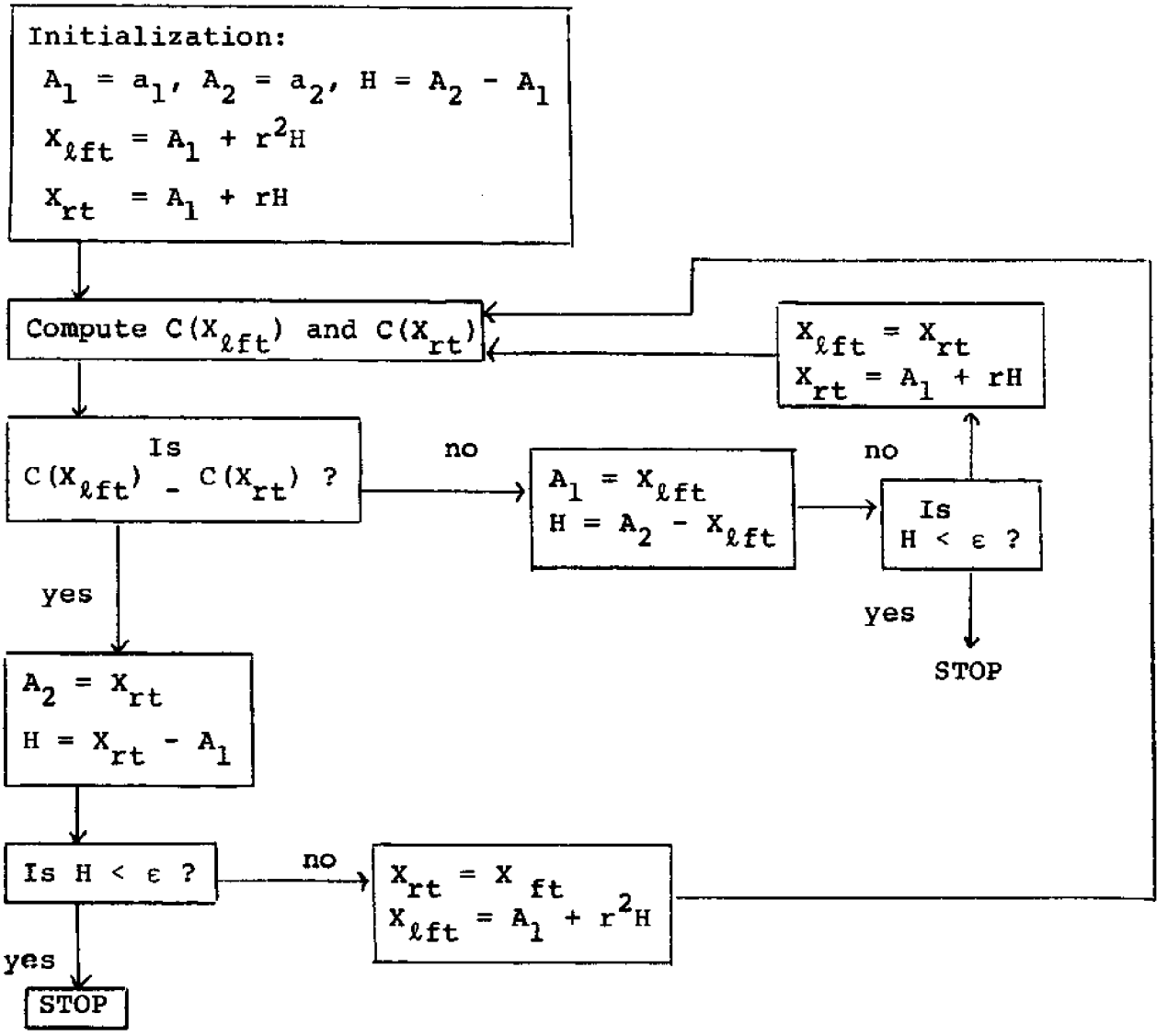
BLANK

APPENDIX C

Method of Golden Search

Method of Golden Search

Suppose we want to maximize C(x) a unimodal function, where x lies in the interval $I = [a_1, a_2]$ (min C(x) = -max -C(x) so no loss of generality). Define $\sigma = .5(\sqrt{5} - 1) = .618033$.



BLANK

APPENDIX D

Regional Port Model

Program Listings

I. Main Program
II. Minimum Cost Multicommodity Flow
III. Modified Steepest-Descent Method

```
0001          INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
             CTX,PR1,PR2,TREL*4,PARBER,NONE,ND,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
             CNIDI,WAIT,XIDLE,RANGE,TNARR
0002          COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
             C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPO,LEXPO,LSIZE,LBUIM,
             CLBUEX,LTYPI,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
             C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
             C5,20,11),NONE(50),ND(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5),NN
0003          COMMON/NEW/IERLY,TCAP(10),TAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
             1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
             2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0004          DIMENSION TARR(50),SAV2AX(50,15)
0005       9909 CONTINUE
0006          READ(5,150)NDAYS,N9,N8
0007          READ(5,130)IX1,XTIME,ITIME,NSYS,NN
0008          READ (5,130)IX,NSAM,NFEQ,NSPD,NP
0009          READ(5,120)((FPAR(I,J),J=1,NP),I=1,NFEQ)
0010          READ(5,900)(NPR(J),J=1,NSYS)
0011          READ(5,250)JCAT,JPAM
0012          READ(5,800)((PRMTX(I,J),J=1,JCAT),I=1,JPAM)
0013          READ(5,251)LTYPE,LIMPO,LEXPO,LSIZE,LAUIM,LBUEX,LTYPI,LTYPX
0014          READ(5,150)NPB
0015          READ(5,860)(NNB(J),J=1,NSYS)
0016          READ(5,870)(NIDI(J),J=1,NSYS)
0017          READ(5,130)ILAM
0018          NFAQ=NFEQ
0019          KNPB=NPB
0020       9900 ALAM=ILAM
0021        120 FORMAT(14I5)
0022        130 FORMAT(I6,4I3)
0023        900 FORMAT(3I3)
0024        800 FORMAT(5I3)
0025        250 FORMAT(2I3)
0026        251 FORMAT(8I2)
0027        150 FORMAT(3I3)
0028        860 FORMAT(3I3)
0029        870 FORMAT(11I5)
0030        870 FORMAT(10I3)
0031          CALL REDIN(NFEQ)
0032          NEWDAY=0
             C * * * * * * * * * * * * * * * * * * * * COMMENT * * * * * * * * * * * * * * * * * *
             C     DO-LOOP 93 SET INITIAL RELEASING TIMES AND DO-LOOP 45 SET ALL THE BERTHS*
             C    IN AVAILABLE STATUS. THE FIRST INDEX IS THE SYSTEM AND THE SECOND ONE IS  *
             C    THE NUMBER OF THE BERTHS.
             C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
0033          DO 8 J=1,NSYS
0034          WAIT(J)=0
0035          XIDLE(J)=0
0036        8 N(J)=0
0037          NIB=NNB(J)
0038          READ(5,850)((PARBER(J,I,K),K=1,NPB),I=1,NIB)
0039          DO 93 JJ=1,NIB
0040          TREL(J,JJ)=ITIME
```

```
0041            93 CONTINUE
0042             8 CONTINUE
0043               TIME=ITIME
0044               NP1=NP+1
0045               NP2=NP1
0046               LSZ=LSIZE+2
0047               LTS=LTYPE+2
0048               LTI=LTYPI+2
0049               DO 2 KK1=1,NDAYS
0050               DO 45 K1=1,NSYS
0051               NIB=NNB(K1)
0052               DO 45 K2=1,NIB
0053               KAVAL(K1,K2)=0
0054            45 CONTINUE
0055               WRITE(6,701)KK1
0056               CALL GENERA(IX,IX1,NSAM,NFEQ,NSPO,NEWDAY)
                C * * * * * * * * * * * * * * * * * * * * COMMENT * * * * * * * * * * * * * * * * * * *
                C *  DO-LOOP 7 TAKES EACH SYSTEM,CALCULATES THE TOTAL NUMBER OF SHIPS IN THE *
                C    SYSTEM (NTOT(J)),DISPLACES EVERY SHIP IN THE ARRIVAL LIST  AS MANY POSI-
                C    TIONS AS SHIPS ARE IN THE QUEUE, THIS DISPLACEMENT IS DONE BY THE       *
                C    STATEMENTS CONTAINED IN LOOP 5.THE SHIPS IN THE QUEUE ARE PLACED AT THE  *
                C    BEGINNING OF THE LIST BY MEANS OF THE STATEMENTS APPEARING IN DO-LOOP 10 *
                C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
0057               DO 7 J=1,NSYS
0058               NTOT(J)=N(J)+NARR(J)
0059               NAR=NARR(J)
0060               NTO=NTOT(J)
0061               LPR=N(J)
0062               IF(LPR)11,11,9
0063            9  DO 5 I=1,NAR
0064               I1=NTO-I+1
0065               I2=NAR-I+1
0066               DO 5 K=1,NP1
0067               PARMAX(J,I1,K)=PARMAX(J,I2,K)
0068            5  CONTINUE
0069               DO 10 I=1,LPR
0070               DO 10 K=1,NP2
0071               PARMAX(J,I,K)=SAVIAX(J,I,K)
0072           10  CONTINUE
0073           11  CONTINUE
0074            7  CONTINUE
                C * * * * * * * * * * * * * * * * * * * * COMMENT * * * * * * * * * * * * * * * * * * *
                C       THE STATEMENTS CONTEINED IN DO-LOOP 20 HAVE AS FUNCTION TO GENERATE    *
                C    DERIVATES PARAMETERS AS PRIORITY ,LENGHT  AND DRAUGHT                     *
                C       DO-LOOP NUMBER 35 ASSIGN PRIORITIES TO SHIPS JUST ARRIVED, THIS PRIORITY*
                C    IS ASSIGNED ACCORDINGLY TO SHIP-TYPE PARAMETER AND IMPORT CARGO-TYPE PARA-*
                C    METERS (KTS AND KTI) THE PRIORITY IS THE ONE CORRESPONDING TO THE HIGHER  *
                C    OF THE TWO.
                C     DO-LOOP 34 SET EQUAL TO 0 THE PRIORITIES IN THE SYSTEM THAT DON'T NEED  *
                C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
0075               DO 20 J=1,NSYS
0076               NTO=NTOT(J)
0077               LPR=N(J)+1
```

```
0078                    IF(NPR(J))36,36,30
0079                 30 DO 35 I=LPR,NTO
0080                    KTS=PARMAX(J,I,LTS)
0081                    KTI=PARMAX(J,I,LTS)
0082                    KTOT=0
0083                    DO 31 L=1,J
0084                    KTOT=NPR(L)+KTOT
0085                 31 CONTINUE
0086                    KTOT=KTOT*2-1
0087                    PR1=PRMTX(KTOT,KTS)
0088                    PR2=PRMTX(KTOT+1,KTI)
0089                    IF(PR1.GT.PR2)GO TO 32
0090                    PR=PR2
0091                    GO TO 33
0092                 32 PR=PR1
0093                 33 PARMAX(J,I,   14)=PR
0094                 35 CONTINUE
0095                    GO TO 38
0096                 36 DO 34 I=LPR,NTO
0097                    PARMAX(J,I,   14)=0
0098                 34 CONTINUE
0099                 38 CONTINUE
0100                 20 CONTINUE
                    C * * * * * * * * * * * * * * * * * * * COMMENT * * * * * * * * * * * * * * * * * *
                    C *     THIS PART OF THE PROGRAM THROUGH STATEMENT   143 ORDER THE SHIPS BY    *
                    C * DECREASING ORDER OF PRIORITIES. THE MATRIX SAV2AX IS USED FOR THE STORAGE *
                    C * OF SHIPS CHARACTERISTICS THAT ARE TRANSFERRED AGAIN TO THE MATRIX PARMAX  *
                    C * BY MEANS OF THE STATEMENTS APPEARING IN DO-LOOP 142                       *
                    C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
0101                    J=1
0102                 65 NTO=NTOT(J)
0103                    IF(NPR(J)) 86, 86,50
0104                 50 DO 80 I=1,NTO
0105                    PRA(I)=PARMAX(J,I,   14)
0106                 80 CONTINUE
0107                    CALL MAXO(LIM,MAX,NTO)
0108                    L=1
0109                 81 PRA(LIM)=0
0110                    DO 82 KK=1,NP2
0111                    SAV2AX(L,KK)=PARMAX(J,LIM,KK)
0112                 82 CONTINUE
0113                    IF(LIM.EQ.NTO)GO TO 85
0114                    I=LIM+1
0115                 83 TT=PRA(I)
0116                    IF(TT.NE.MAX)GO TO 84
0117                    LIM=I
0118                    L=L+1
0119                    GO TO 81
0120                 84 IF(I.GE.NTO)GO TO 85
0121                    I=I+1
0122                    GO TO 83
0123                 85 IF(MAX.EQ.1) GO TO 86
0124                    MAX=MAX-1
```

```
0125                    I=1
0126                    GO TO 83
0127                 86 DO 142 I=1,NTO
0128                    IBT=SAV2AX(I,7)
0129                    IF(IBT.EQ.0)GO TO 1143
0130                    IF(IBT.GT.0)GO TO 1144
0131                    IBT=-IBT
0132                    K=PARBER(J,IBT,10)
0133                    IIM(K)=IIM(K)+SAV2AX(I,4)
0134                    IEX(K)=IEX(K)+SAV2AX(I,5)
0135                    PARBER(J,IBT,7)=I
0136                    GO TO 1143
0137                1144 SAV2AX(I,7)=0
0138                1143 DO 142 KK=1,NP2
0139                    PARMAX(J,I,KK)=SAV2AX(I,KK)
0140                 142 CONTINUE
0141                 143 CONTINUE
0142                    NSHIP=NTOT(J)
0143                    WRITE(6,510)KK1,J
0144                    WRITE(6,511)
0145                    WRITE(6,512)
0146                    WRITE(6,71)((PARMAX(J,II,KK),KK=1,NP2),II=1,NSHIP)
0147                    CALL FEAS1(RANGE,J,KK1)
0148                    CALL ASSIGN(RANGE,J,KK1)
0149                    NQ=NNB(J)
0150                    DO 600 I=1,NQ
0151                 600 CALL NEXT(J,I)
0152                    CALL DEV(J)
0153                  55 IF(J-NSYS)144,146,146
0154                 144 J=J+1
0155                    GO TO 65
0156                 146 CONTINUE
0157                    CALL SIML
0158                    WRITE(6,507)
0159                    DO 87 J=1,NSYS
0160                    NPB=NNB(J)
0161                    WRITE(6,506)(PARBER(J,LL,2),TREL(J,LL),LL=1,NPB)
0162                  87 CONTINUE
0163                    WRITE(6,508)
0164                    DO 88 J=1,NSYS
0165                    NPB=NNB(J)
0166                    WRITE(6,506)(PARBER(J,LL,2),KAVAL(J,LL),LL=1,NPB)
0167                  88 CONTINUE
0168                   2 CONTINUE
0169                 701 FORMAT('1',20X,' LIST OF SHIPS ON ARRIVAL AT DAY ',I3)
0170                 511 FORMAT('0',10X,'ARRIVAL   SHIP    TYPE   IMPORT   EXPORT  SIZE
                        CMPORT  EXPORT   TYPE    TYPE   SYSTEM  LENGHT   DEPTH  PRIORITY')
0171                  71 FORMAT('0',10X,15I8)
0172                 510 FORMAT('1',20X,'SHIPS TO BE ASSIGNED AT DAY',I2,' IN SYSTEM',I2)
0173                 512 FORMAT(' ',10X,' TIME    NUMBER   SHIP    CARGO    CARGO  SHIP
                        CBULK    BULK   IMPORT  EXPORT ')
0174                 506 FORMAT('0',10X,10(' BERTH',I2,'=',I5))
0175                 508 FORMAT('-',20X,' NON- AVAILABILITY STATUS OF BERTHS')
```

```
0176          507 FORMAT('-',20X,' EXPECTED RELEASING TIME OF BERTHS')
0177              READ(5,120,END=9902)IX1,IX,ITIME,NN,ILAN
0178              IF(IX1.EQ.00)GO TO 9909
0179              NFEQ=NFAQ
0180              NPB=KNPB
0181              GO TO 9900
0182          9902 STOP
0183              END
```


Information Processing Center

```
      C     READ IN TERMINAL COMPLEX PARAMETERS AND INITIALIZE DATA
0001          SUBROUTINE REDIN(NFEQ)
0002          INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
     C1X,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
     CAIDI,WAIT,XIDLE,RANGE,TNARR
0003          COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NNL(5),NTOT(5),N(5),NARR(5),NPR(5)
     C,TIML,TARR(50),TREL(5,20),NUMB(50),LTYPE,LIMFO,LEXPO,LSIZE,LDUIM,
     CLBDEX,LTYPE,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
     C15),PRATX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
     C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004          COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
     1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
     2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005          READ(5,100)NCOMP, (SRATE1(K),RATE(K),CAP(K),K=1,NCOMP)
0006      100 FORMAT(I5/(3F6.0))
0007          READ(5,200)  ((MPLX(K,L),L=1,6),K=1,NCOMP)
0008      200 FORMAT(6I6 )
0009          READ(5,200)IRRD,IRRLIM,IX3,IX4
0010          DO 101 K=1,NCOMP
0011          J=K
0012          IBRTH(J)=0
0013          IMUN(J)=0
0014          IAUN(J)=0
0015          IIM(J)=0
0016          IEX(J)=0
0017          MPLX(K,2)=-100
0018          MPLX(K,3)=-100
0019          MPLX(K,4) = ITIME
0020          MPLX(K,6) = MPLX(K,1)
0021          SRATE(K)=SRATE1(K)
0022      101 TCAP(K)=0.5*CAP(K)
0023          DO 201 J=1,NSYS
0024          ITRND(J)=0
0025          NN=NNB(J)
0026          DO 201 I=1,NN
0027          TREL(J,I)=0
0028          PARBER(J,I,7)=0
0029          PARBER(J,I,8)=0
0030          PARBER(J,I,9)=0
0031      201 PARBER(J,I,11)=0
0032          DO 300 J=1,NFEQ
0033          FPAR(J,  14)=-10
0034      300 FPAR(J,  9)=-600
0035          JTIME=ITIME+XTIME
0036          RETURN
0037          END
```

```
0001          SUBROUTINE GENERA(IX,IX1,NSAM,NEEQ,NSPO,NEWDAY)
              C
              C * * * * * * * * * * * * * * * * * * * * * * COMMENT * * * * * * * * * * * * * * * * * * *
              C *    THE FUNCTION OF THIS SUBROUTINE IS TO GENERATE THE ARRIVAL OF SHIPS                  *
              C * AND THEIR CHARACTERISTICS.                                                              *
              C       INPUT-- CHARACTERISTIC FREQUENCY TABLE                                             *
              C               INTERARRIVAL DISTRIBUTION FUNCTION                                         *
              C *             CURRENT TIME                                                               *
              C *             NUMBER OF SHIPS IN PORT                                                    *
              C *             SEEDS FOR THE GENERATION OF RANDOM NUMBERS (IX AND IX1)                    *
              C *     OUTPUT- NUMBER OF SHIPS ARRIVED DURING PERIOD AND  CHARACTERISTICS OF              *
              C * THOSE SHIPS                                                                            *
              C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
              C
0002              INTEGER*2 FPAR,PARMAX,TAPR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
                 CTX,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
                 CNIDI,WAIT,XIDLE,RANGE,TNARR
0003              COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NNI(5),NTOT(5),N(5),NARR(5),NPR(5)
                 C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPC,LEXPC,LSIZE,LBUIM,
                 CLBUDX,LTYPI,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
                 C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
                 C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5),NN
0004              COMMON/NEW/IERLY,TCAP(10),IAUN(10),MFLX(10,6),ITRND(10),IBRTH(10),
                 1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
                 2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005              DIMENSION IARR(50),NDM(50)
0006              NTARR=0
0007              DO 45 J=1,NSYS
0008              NARR(J)=0
0009           45 CONTINUE
0010              NLAST=NN
0011              KTIME=TIME
0012              I=1
0013              IF(NEWDAY.LE.0)GO TO 3
0014              IARR(I)=NEWDAY
0015              GO TO 33
              C *                RANDOM NUMBER GENERATION FOR INTERARRIVAL TIMES
              C * YFL IS THE OUTPUT AND ITS RANGE IS 0-1
              C
0016            3 IY1=IX1*65539
0017              IF(IY1)12,13,13
0018           12 IY1=IY1+2147483647+1
0019           13 YFL=IY1
0020              YFL=YFL*.4656613E-9
              C * * * * * * * * * * * * * * * * * * * * COMMENT * * * * * * * * * * * * * * * * * * * *
              C      THE FOLLOWING FUNCTION HAS BEEN CHOSEN ARBITRARILY AND COULD BE OTHER               *
              C    MORE REALISTIC AS AN ERLANG OR EXPONENTIAL.                                           *
              C    THE CALL FOR ANY OTHER FUNCTION COULD BE                                              *
              C                                          IARR(I)=INTRA(YFL,GAMA)                         *
              C    THE FUNCTION COULD BE DEFINED AS AN EXPONENTIAL IN THAT CASE-                         *
              C                                          INTRA(YFL,MEAN)=-MEAN*LOG(YFL)                  *
              C                                                                                          *
              C    THIS FUNCTION DEFINITION MUST BE PLACED BEFORE THE FIRST EXEC. STATEMENT             *
              C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
0021                    RDM(I)=YFL
0022                    IARR(I)=-ALOG(     YFL)*ALAM
0023                    IX1=IY1
0024                 33 KTIME=KTIME+IARR(I)
0025                    NTARR=NTARR+IARR(I)
0026                    IF(NTARR-XTIME)10,10,20
0027                 10 TARR(I)=KTIME
0028                    I=I+1
0029                    GO TO 3
0030                 20 INARR=I-1
0031                    NEWDAY=NTAPR-XTIME
0032                    IF(INARR.EQ.0)RETURN
0033                    NN=NLAST+INARR
0034                    LL=1
                     C  * RANDOM NUMBER GENERATION FOR SHIP CHARACTERISTICS
0035                  1 IY=IX*65539
0036                    IF(IY)5,6,6
0037                  5 IY=IY+2147483647+1
0038                  6 YFL=IY
0039                    YFL=YFL*.4656613E-9
0040                    NFEQ=1+NSAM*YFL
0041                    IF(NSAM.LT.NFEQ)NFEQ=NSAM
0042                    IX=IY
0043                    I=1
                     C  * FPAR(I,1) IS THE UPPER LIMIT OF THE CUMULATIVE FREQUENCY FOR CLASS I      *
0044                  7 IF(NFEQ-FPAR(I,1))15,15,30
0045                 30 I=I+1
0046                    GO TO 7
0047                 15 MN=I
0048                    J=FPAR(MN,NSPO)
0049                    NARR(J)=NARR(J)+1
0050                    KK=NARR(J)
                     C      THIS PART ASSIGN ELEMENTS FROM FPAR TO PARMAX FROM POSITION 2 ON
                     C    NOTICE THAT ELEMENT K IS PLACED IN POSITION K+1 IN PARMAX
0051                    DO 40 K=2,NP
0052                    PARMAX(J,KK,K+1)=FPAR(MN,K)
0053                    PARMAX(J,KK,1)=TARR(LL)
0054                    PARMAX(J,KK,2)=NLAST+LL
0055                    PARMAX(J,KK,14)=1000*RDM(LL)
0056                 40 CONTINUE
0057                    IF(LL.GT.TNARR)GO TO 8
0058                    LL=LL+1
0059                    GO TO 1
0060                  8 CONTINUE
0061                    WRITE(N8,500)NN
0062                500 FORMAT('0',10X,'TOTAL NUMBER OF SHIPS =',I3)
0063                    DO 25 J=1,NSYS
0064                    IF(NARR(J).EQ.0)GO TO 25
0065                    NSHIP=NARR(J)
0066                    NP1=NP+1
0067                    WRITE(N8,200)J
0068                    WRITE(N8,210)((PARMAX(J,II,KK),KK=1,NP1),II=1,NSHIP)
0069                 25 CONTINUE
```

```
0070          200 FORMAT('0',10X,'SYSTEM NUMBER',I3//)
0071          210 FORMAT (1H,10X,15I8)
0072              RETURN
0073              END
```

```
0001          SUBROUTINE FEAS1(RANGE,J,KK1)
              C * * * * * * * * * * * * * * * * * * * * COMMENT * * * * * * * * * * * * * * * *
              C   THIS IS THE FEASIBILITY SUBROUTINE. ITS MAIN ROLE IS TO DEFINE THE MA-      *
              C  TRIX A(15,18) EVERY PERIOD. AFTER AL THE FEASIBILITY CONDITIONS HAS BEEN     *
              C  ACCOMPLISHED THEN THE NUMBER 1 IS ASSIGNED TO THE CORRESPONDENT POSITION    *
              C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
0002          INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
              CTX,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,XAVAL,NUMSH,TINQ,FINDLE,A,X,
              CNID1,WAIT,XIDLE,RANGE,TNARR
0003          COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
              C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPL,LIMPC,LEXPO,LSIZE,LBUIM,
              CLBUEX,LTYPI,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
              C15),PRMTX(6,5),PRA(50),XTIME,XAVAL(5,20),IDLE(50),A(50,20),PARBER(
              C5,20,11),NONE(50),NO(50),NUMSH(50),NID1(5),WAIT(5),XIDLE(5)
0004          COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRNO(10),IBRTH(10),
              1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SKATE(10),SRATE(10),
              2 NCOMP,IRRO,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005          DO 18 IS=1,50
0006          DO 18 IB=1,20
0007          A(IS,IB)=0
0008       18 CONTINUE
0009          INT=XTIME
0010          RANGE=0
0011          IB=0
0012          LIM=NNB(J)
0013          DO 10 LL=1,LIM
0014          IB=IB+1
0015          RANGE=RANGE+1
0016          IF(TREL(J,LL)-TIME-INT)5,5,15
0017       15 NO(IB)=-LL
0018          GO TO 10
0019        5 NO(IB)=LL
0020          XAVAL(J,   IB )=0
0021       10 CONTINUE
0022          IF(RANGE.EQ.0) GO TO 100
0023          NTO=NTOT(J)
0024          DO 90 IS=1,NTO
0025          NONE(IS)=0
0026          IB=RANGE
0027          DRAUGH=PARMAX(J,IS,   13)
0028       20 DEPTH=PARBER(J,   IB ,5)
0029          IF(DRAUGH-DEPTH)30,30,25
0030       25 IF(IB.EQ.1)GO TO 90
0031          IB=IB-1
0032          GO TO 20
0033       30 NONE(IS)=IB
0034          MAXB=IB
0035          LS=PARMAX(J,IS,   12)
0036          DO 80 IB=1,MAXB
0037          LB=PARBER(J,   IB ,4)
0038          IF(LS-LB)40,40,80
0039       40 IF(PARMAX(J,IS,LTYPE+2).EQ.1)GO TO 60
0040          IF(PARMAX(J,IS,LTYPE+2).NE.PARBER(J,   IB ,5))GO TO 80
```

```
0041          60 A(IS,IB)=1
0042          80 CONTINUE
0043          90 CONTINUE
0044             WRITE(6,508)
0045             WRITE(6,511)
0046             WRITE(6,512)
0047             WRITE(6,510)(PARBER(J,   IB  ,2),IB=1,LIM)
0048             DO 28 IS=1,NTO
0049             WRITE(6,505)PARMAX(J,IS,2),(A(IS,IB),IB=1,LIM)
0050          28 CONTINUE
0051             GO TO 110
0052         100 CONTINUE
0053             WRITE(6,509)
0054         110 CONTINUE
0055         510 FORMAT('0',20X,10I4)
0056         511 FORMAT('0',4X,'NUMBER OF SHIP  **')
0057         512 FORMAT('0',20X,'**  NUMBER OF BERTH')
0058         505 FORMAT('0',16X,I3,' *',10I4)
0059         508 FORMAT('-',20X,'FEASIBILITY MATRIX')
0060         509 FORMAT('0',10X,'NO BERTHS AVAILABLE',///)
0061             RETURN
0062             END
```

```
0001          SUBROUTINE ASSIGN(RANGE,J,KK1)
0002          INTEGER TRAL(20)
0003          INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
             C1X,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
             CNIDI,WAIT,XIDLE,RANGE,TNARR
0004          COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
             C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPU,LEXPU,LSIZE,LBUIM,
             CLBUEX,LTYP1,LTYPX,NNB(20),PARMAX(5,50,5),FPAR(50,15),SAV1AX(5,50,
             C15),PRNTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
             C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0005          COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRNO(10),IBRTH(10),
             1 IMUN(10),IINI(10),IEX(10),CAP(10),RATL(10),SRATE(10),SRATE1(10),
             2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0006          DIMENSION X(50,20)
             C * * * * * * * * * * * * * * * * * * * * COMMENT * * * * * * * * * * * * * * * * * * *
             C   SERVT(K1,KG) IS AN ARBITRARY FUNCTION USED FOR DETERMINE THE EXPECTED      *
             C  SERVICE TIME. ANY FUNCTION COULD BE USED TAKING CARE THAT THE PARAMETERS    *
             C  IN THE CALL STATEMENT WOULD BE PREVIOUSLY DEFINED.                          *
             C   THE STATEMENTS DEFINING THE PARAMETERS IN THIS CASE ARE THOSE CORRESPON-   *
             C  DING TO THE TYPE OF CARGO (LTYPI) AND AMOUNT OF CARGO (LIMPU).  THE DEFINI-*
             C  TION OF PARAMETERS HAS TO BE DONE  TAKING CARE OF THE POSITIONS  OF THOSE   *
             C  PARAMETERS IN THE PARMAX AND PARBER MATRIX.                                 *
             C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
0007          NN=NNB(J)
0008          DO 600 K=1,NN
0009     600  TRAL(K)=TREL(J,K)
0010          TINQ=0
0011          TINDLE=0
0012          NTAS=0
0013          NSHIP=NTOT(J)
0014          NP2=14
0015          NSAVE=1
0016          WRITE(6,702)KK1,J
0017     702  FORMAT('-',20X,' SHIPS ASSIGNED AT DAY',I3,' IN SYSTEM',I2//)
0018          DO 85 IS=1,NSHIP
0019          IF(PARMAX(J,IS,7).LT.0)GO TO 85
0020          TARR(IS)=PARMAX(J,IS,1)
0021          MINI=)
0022          INTER=TIME+XTIME-TARR(IS)
0023          MINQ=INTER
0024          IF(NTAS.EQ.RANGE)GO TO 75
0025          IF(NONE(IS).EQ.0)GO TO 75
0026          IB=NONE(IS)
0027     20   IF(KAVAL(J,   IB ).EQ.1)GO TO 50
0028          IF(A(IS,IB).LE.0)GO TO 50
0029          IDLE(IB)=TARR(IS)-TRAL(IB)
0030          IF(IDLE(IB))35,35,40
0031     35   NWAIT=-(IDLE(IB))
0032          IF((TARR(IS)+NWAIT).GE.(TIME+XTIME))GO TO 50
0033          IF(NWAIT.GE.MINQ)GO TO 50
0034          MINQ=NWAIT
0035          NASQ=IB
0036          GO TO 50
```

```
0037          40 IF(MINI.NE.0)GO TO 41
0038             MINI=IDLE(IB)
0039             NASO=IB
0040             GO TO 50
0041          41 NIDLE=IDLE(IB)
0042             IF(NIDLE.GT.MINI)GO TO 50
0043             MINI=NIDLE
0044             NASO=IB
0045          50 IF(IB.EQ.1)GO TO 65
0046             IB=IB-1
0047             GO TO 20
0048          65 IF(MINI.NE.0)GO TO 70
0049             IF(MINQ.EQ.INTER)GO TO 75
0050             M1=PARMAX(J,IS,LTYP1+2)
0051             M2=PARMAX(J,IS,LIMPO+2)
0052             IB=NASO
0053             PARMAX(J,IS,9)=TRAL(IB)
0054             PARMAX(J,IS,7)=IB
0055             X(IS,IB)=1
0056             TINQ=TINQ+MINQ
0057             IRLS=TRAL(IB)
0058             TRAL(IB)=IOPT(J,IB,IS,IRLS)
0059             GO TO 64
0060          70 IB=NASO
0061             M1=PARMAX(J,IS,LTYP1+2)
0062             M2=PARMAX(J,IS,LIMPO+2)
0063             X(IS,IB)=1
0064             IRLS=TARR(IS)
0065             TRAL(IB)=IOPT(J,IB,IS,IRLS)
0066             PARMAX(J,IS,9)=TARR(IS)
0067             PARMAX(J,IS,7)=IB
0068             TINDLE=TINDLE+MINI
0069          64 IF(TRAL(   IB   )-(TIME+XTIME))67,69,69
0070          69 KAVAL(J,   IB )=1
0071             NTAS=NTAS+1
0072             GO TO 80
0073          67 KAVAL(J,   IB )=0
0074          80 NUMSH(   IB )=IS
0075             NUMBL(IS)=   IB
                C    K10 AND K11 ARE THE NUMBER OF SHIPS AND BERTHS STORED IN COLUMN NUMBER 2  *
                C    OF THE PARMAX AND PARBER MATRIXS RESPECTIVELY                             *
0076             K10=PARMAX(J,IS,2)
0077             K11=PARBER(J,   IB ,2)
0078             NTEXPE=TRAL(IB)
0079             WRITE(6,513)K10,K11,NTEXPE
0080             GO TO 85
0081          75 DO 95 K=1,NP2
0082             SAVIAX(J,MSAVE,K)=PARMAX(J,IS,K)
0083          95 CONTINUE
                C    THIS STATEMENTS ARE PROVIDED FOR THE INCREMENT OF PRIORITY WITH TIME
                C    THE VARIABLE NIDI(J) DEFINES THE INCREMENT OF PRIORITY
0084             IF(NIDI(J).EQ.0)GO TO 96
0085             SAVIAX(J,MSAVE,NP2)=SAVIAX(J,MSAVE,NP2)+NIDI(J)
```

```
0086          96 CONTINUE
0087             PARMAX(J,IS,9)=0
0088             PARMAX(J,IS,7)=0
0089             MSAVE=MSAVE+1
0090          85 CONTINUE
0091             LSQ=MSAVE - 1
0092             IF(LSQ.EQ.0) GO TO 86
0093             WRITE(6,516)
0094             WRITE(6,517)
0095             WRITE(6,703)KK1,J
0096             WRITE(6,514)((SAVIAX(J,IS,K),K=1,NP2),IS=1,LSQ)
0097          86 CONTINUE
0098             WAIT(J)=WAIT(J)+TINQ
0099             XIDLE(J)=XIDLE(J)+TINDLE
0100             WRITE(6,802)
0101             WRITE(6,807)TINQ,TINDLE
0102             WRITE(6,803)
0103             WRITE(6,807)WAIT(J),XIDLE(J)
0104         517 FORMAT(' ',10X,' TIME   NUMBER   SHIP   CARGO   CARGO   SHIP
               CBULK   BULK   IMPORT   EXPORT ')
0105         516 FORMAT('0',10X,'ARRIVAL   SHIP   TYPE   IMPORT   EXPORT   SIZE   I
               CMPORT   EXPORT   TYPE   TYPE   SYSTEM   LENGHT   DEPTH   PRIORITY')
0106         514 FORMAT('0',10X,1518)
0107         513 FORMAT('0',10X,'SHIP NUMBER',I3,' ASSIGNED TO BERTH',I3,' ITS RELE
               CASING TIME IS',I4)
0108         703 FORMAT('-',20X,' CHARACTERISTICS OF SHIPS IN QUEUE AT THE END OF D
               CAY',I3,'  IN SYSTEM',I2)
0109         802 FORMAT('0',20X,'DAILY STATISTICS')
0110         803 FORMAT('0',20X,'TOTAL STATISTICS')
0111         807 FORMAT('-',10X,'TOTAL WAITING TIME   =',I5//'0',10X,'TOTAL BERTH I
               CDLE TIME=',I5)
0112             RETURN
0113             END
```


Information Processing Center

```
0001          SUBROUTINE MAXO(LIM,MAX,NTO)
              C *.* * * * * * * * * * * * * * * * * * * * COMMENT * * * * * * * * * * * * * * * * * *
              C  *   THIS SUBROUTINE LOOKS FOR THE SHIP WITH THE HIGHEST  PRIORITY IN THE     *
              C  * SYSTEM                                                                      *
              C
0002          INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAVIAX,SAV2AX,PRM
             CTX,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
             CNIDI,WAIT,XIDLE,RANGE,TNARR
0003          COMMON NSYS,NY,NP,NPR,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
             C,TIME,TARR(50),TREL(5,20),NJMBE(50),LTYPE,LIMPO,LEXPO,LSIZE,LBUIM,
             CLBUFX,LTYPI,LTYPX,ANB(20),PARMAX(5,50,15),FPAR(50,15),SAVIAX(5,50,
             C15),PRMIX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
             C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004          COMMON/NEW/ IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IORTH(10),
             1 IMUN(10),LIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
             2 NCOMP,IRRO,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005          MAX=PRA(1)
0006          LIM=1
0007          DO 30 I=2,NTO
0008          IF(MAX.GE.PRA(I))GO TO 30
0009          MAX=PRA(I)
0010          LIM=I
0011       30 CONTINUE
0012          RETURN
0013          END
```

```
      C     GENERATES NORMAL DIST RANDOM DEVIATIONS FROM SCHED ARRIVALS,
      C     ALSO IMPORT AND EXPORT VOLUMES
0001          SUBROUTINE DEV(J)
0002          INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
             GTX,PR1,PR2,TREL*4,PARDLP,ACRE,NC,IDLE,KAVAL,NUMSH,TINC,TINDLE,A,X,
             GN1DI,WAIT,XIDLE,RANGE,TNARR
              COMMON NDYS,KY,KP,NPB,JCAT,JPAR,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
             G,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPC,LEXPC,LSIZE,LPUIP,
             GLBUCK,LTYP1,LTYPX,NNd(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
             G15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
             G5,20,11),NUNE(50),NU(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004          COMMON/NEW/IEFLY,TCAP(10),IAUN(10),MFLX(10,6),ITRND(10),IRRTH(10),
             1 IMUN(10),IIM(10),IcX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
             2 NCUMP,IRRD,IFFLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005          NN=NTU(J)
0006          IF(NN.EQ.0)GO TO 100
0007          DO 101 I=1,NN
0008          IF(PARMAX(J,I,10).NE.-0001GO TO 101
0009          S=1RRD
0010          AM=PARMAX(J,I,1)
0011          CALL GAUSS(IX3,S,AP,V)
0012          PARMAX(J,I,1)=IRCUND(V)
0013          AM1 = PARMAX(J,I,4)
0014          AM2 = PARMAX(J,I,5)
0015          AM3=PARMAX(J,I,6)
0016          AMO=AM1-5.0
0017          S=AM3-AM1
0018          IF(AMO.LT.5)S=AM6
0019          CALL RANDU(IX4,IY4,V)
0020          IX4=IY4
0021          V=AM1-5+V*5*2.0
0022          PARMAX(J,I,4)=IRCUND(V)
0023          AMO=AM2-5.0
0024          S=AM3-AM2
0025          IF(AMO.LT.5)S=AM6
0026          CALL RANDU(IX4,IY4,V)
0027          IX4=IY4
0028          V=AM2-5+V*5*2.0
0029          PARMAX(J,I,5) = IRCUNJ(V)
0030      101 CONTINUE
0031      100 CONTINUE
0032          RETURN
0033          END
```

```
0001          SUBROUTINE NEXT(J,L)
0002          INTEGER*2 PAR,PARMAX,TARR(4),XTIME,TREL(4),PRA,PR,SAV1AA,SAV2AA,PRM
              CTX,PR1,PR2,TREL*4,PARBER,NORE,NO,IDLE,KAVAL,NUMSH,TIND,TINDLE,A,X,
              CNIDI,WAIT,XIDLE,RANGE,ENAME
0003          COMMON NSYS,NY,NP,NPR,JCAT,JPAR,NTJT(5),NIDT(5),N(5),NARR(5),NPR(5)
              C,TIME,TARR(50),TREL(5,2),NUMRL(50),LTYPL,LIMPO,LEXPO,LSIZE,LOUIM,
              CLBUEX,LTYP1,LTYPX,PNR(20),PARMAX(5,50,15),PPAR(50,15),SAVEAX(5,50,
              C15),PRMEX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
              C5,2),II),NUNE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004          COMMON/NEW/ IERLY,ICAP(10),IAUN(10),MPLX(10,6),ITRND(10),IGRTH(10),
              1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
              2 NCOMP,IRRD,IRRL(4,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005          PARBER(J,L,8)=0
0006          NN=NTJT(J)
0007          IF(NN.EQ.0)RETURN
0008          MERLY=JTIME
0009          IO=0
0010          DO 100 I=1,NN
0011          IF(PARMAX(J,I,7).NE.L.OR.PARMAX(J,I,9).GE.MERLY)GO TO 100
0012          MERLY=PARMAX(J,I,9)
0013          IO=I
0014      100 CONTINUE
0015          PARBER(J,L,8)=IO
0016          PARBER(J,L,9)=  MAXO(TIME,TREL(J,L),MERLY)
0017          RETURN
0018          END
```

```
C.SYSIN DD *
C         FINDS EXPECTED RELEASE TIME FOR SHIP M ASSIGNED TO BERTH L SYS J
C         GIVEN BERTHS RELEASE TIME IS IRLS
      FUNCTION IDPT(J,L,M,IRLS)
      INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
     CTX,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
     CNIDI,WAIT,XIDLE,RANGE,TNARR
      COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NNL(5),NTOT(5),N(5),NARR(5),NPR(5)
     C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPO,LEXPC,LSIZE,LBUIM,
     CLBUEX,LTYPI,LTYPX,KNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
     C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
     C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
      COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
     1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
     2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
      K=PARBER(J,L,10)
      IF(A(M,L).NE.0)GO TO 10
      IDPT=-10
      RETURN
   10 ANVX=PARMAX(J,M,5)
      ANVM=PARMAX(J,M,4)
      IUNT=PARMAX(J,M,8)
      IF(MPLX(K,6).LT.IUNT)IUNT=MPLX(K,6)
      IF(IUNT.EQ.0)GO TO 21
      IF(MPLX(K,2).EQ.000.OR.MPLX(K,3).EQ.000)GO TO 12
      IF(ANVM.GT.(CAP(K)-TCAP(K)).OR.ANVX.GT.TCAP(K))GO TO 12
      IDPT=IRLS + (ANVX + ANVM)/(IUNT*SRATE1(K))
      RETURN
   12 TRATE = RATE(K)/(FLOW(J,K) + IUNT*(ANVM-ANVX)/(ANVM+ANVX))
      IF(TRATE.LT.0.0)TRATE=-TRATE
      IF(TRATE.GT.SRATE1(K))TRATE=SRATE1(K)
      IF(MPLX(K,3).EQ.000.OR.ANVX.GT.TCAP(K))GO TO 14
      IDPT=IRLS + (ANVM+ANVX)/(TRATE*(1.0-(CAP(K)-TCAP(K)) /ANVM)
     1   +     SRATE1(K)*IUNT*(CAP(K)-TCAP(K))/ANVM)
      RETURN
   14 IDPT=IRLS + (ANVM+ANVX)/(TRATE*(1.0-TCAP(K)/ANVX)  +
     1  SRATE1(K)*IUNT*(TCAP(K)/ANVX))
      RETURN
   21 IDPT=32000 + IRLS - TIME
      IF(IDPT.GT.32600)IDPT=32600
      RETURN
      END
```

```
      C     FINDS ROUNDED INTEGER FROM REAL
0001          FUNCTION IROUND(A)
0002          I=A
0003          B=I
0004          IF((2.0*A - 2.0*B).LT.1.0)GO TO 100
0005          I=I+1
0006      100 IROUND =I
0007          RETURN
0008          END
```

```
      C     SIMULATES DAYS ACTIVITY BASED ON TENTATIVE SCHEDUKE
0001        SUBRCUTINE SIML
0002        INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
           CTX,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
           CNIDI,WAIT,XIDLE,RANGE,TNARR
0003        COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTCT(5),N(5),NARR(5),NPR(5)
           C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPO,LEXPO,LSIZE,LBUIM,
           CLBUEX,LTYPI,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
           C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
           C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004        COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
           1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
           2 NCOMP,IRRD,IRRLIP,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005    300 CALL WHENI      IFUL,JFUL,KFUL      )
0006        WRITE(N8,200) IERLY,IFUL,JFUL,KFUL
0007        IF(IFUL.EQ.0.OR.IFUL.EQ.1) GO TO 100
0008        IF(IFUL.EQ.5) GO TO 105
0009    200 FORMAT('0',5X,'TIME = ',I10,5X,'CODES = ',3I5)
0010        IF (IFUL.EQ.4)GO TO 201
0011        WRITE(N8,202) (PARMAX(JFUL,KFUL,K),K=1,15)
0012    202 FORMAT('0SHIP PARAMETERS ',I10,I5,I3,3I10,I4,2I3,I10,I3,I5,I4,I3,
           1 I10)
0013        GO TO 203
0014    201 WRITE(N8,204) (PARBER(JFUL,KFUL,K),K=1,11)
0015    204 FORMAT('0BERTH PARAMETERS ',11I10)
0016    203 CALL BERTRL(JFUL,KFUL,IFUL)
0017        GO TO 300
0018    100 WRITE(N8,250)  (MPLX(KFUL,K),K=1,6)
0019    250 FORMAT('0COMPLX PARAMETERS ',6I12)
0020        CALL TERM(JFUL,KFUL,IFUL)
0021        GO TC 300
0022    105 CALL END
0023        RETURN
0024        END
```

```
      C     FINDS TIME FOR NEXT EVENT CODES FOR NEXT EVENT
0001        SUBROUTINE  WHENT      (IFUL,JFUL,KFUL      )
0002        INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PPM
     1(X,PK1,PK2,TREL*4,PARBER,NCNE,AC,IDLE,KAVAL,NUMSH,TINC,TINDLE,A,X,
     2NIDI,WAIT,XIDLE,RANGE,TNARR
0003        COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NAPR(5),NPR(5)
     1,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPO,LEXPO,LSIZE,LBUFM,
     2LBUFX,LTYPI,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
     315),PRMTX(5,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
     45,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004        COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
     1 IMUN(10),IIM(10),ICX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
     2 NCOMP,IKKO,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005        IERLY=JTIME
0006        IFUL=5
0007        JFUL=0
0008        KFUL=0
0009        DO 200 J=1,NSYS
0010        NN=NNB(J)
0011        DO 100 K=1,NN
0012        IF(TREL(J,K).GE.IERLY.OR.PARBER(J,K,7).EQ.0)GO TO 100
0013        IERLY=TREL(J,K)
0014        IFUL=4
0015        JFUL=J
0016        KFUL=K
0017    100 CONTINUE
0018        NN=NTOT(J)
0019        IF(NN.EQ.0)GO TO 201
0020        DO 200 K=1,NN
0021        IF(PARMAX(J,K,1).GE.IERLY.OR.PARMAX(J,K,10).NE.-600)GO TO 202
0022        IERLY=PARMAX(J,K,1)
0023        IF(IERLY.LT.TIME)IERLY=TIME
0024        IFUL=3
0025        JFUL=J
0026        KFUL=K
0027    202 IF((PARMAX(J,K,9)+IRRLIM).EE.IERLY.OR.PARMAX(J,K,10).NE.-600)GO TO
     1 200
0028        IF(PARMAX(J,K,7).LE.0)GO TO 200
0029        IERLY=  PARMAX(J,K,9)
0030        IFUL=2
0031        JFUL=J
0032        KFUL=K
0033    200 CONTINUE
0034    201 DO 300 K=1,NCOMP
0035        IF(MPLX(K,2).GT.IERLY.OR.MPLX(K,2).LE.0)GO TO 301
0036        IERLY= MPLX(K,2)
0037        IFUL=1
0038        JFUL=  MPLX(K,5)
0039        KFUL=K
0040        GO TO 300
0041    301 IF(MPLX(K,3).GT.IERLY.OR.MPLX(K,3).LE.0)GO TO 300
0042        IERLY=MPLX(K,3)
0043        IFUL=0
```

```
0044                JPUL=MPLX(K,5)
0045                KPUL=K
0046          300   CONTINUE
0047                RETURN
0048                END
```

```
0001                C     FINDS CARGO VOLUMES REMAINING FOR ONE OR ALL SHIPS OF COMPLX K, SYS J
                    C     INPUT VARIABLES L & M SAME AS FOR ROUTINE RELEAS
0001                      SUBROUTINE CARGO(J,K,L,M,IEND,IERR)
0002                      INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,FRA,PR,SAV1AX,SAV2AX,PRM
                    C,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
                    CNIDI,WAIT,XIDLE,RANGE,TNARR
0003                      COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
                    C,TIME,TARR(50),TREL(5,20),NUMBE(5)1,LTYPE,LIMPO,LEXPO,LSIZE,LBUIM,
                    CLBUFX,LTYP1,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
                    C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(5J,20),PARBER(
                    C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004                      COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
                    1 IMUN(10),ITM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
                    2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005                      IERR = 0
0006                      IF(L.EQ.0.AND.M.EQ.0)GO TO 100
0007                      IF(L.EQ.0)L=-PARMAX(J,M,7)
0008                      IF(M.EQ.0)M= PARBER(J,L,7)
0009                      IF(L.LE.0.OR.M.LE.0)GO TO 101
0010                      ANVM= PARMAX(J,M,4)
0011                      ANVX= PARMAX(J,M,5)
0012                      IUNT= PARMAX(J,M,9)
0013                      IRLS = TREL(J,L)
0014                      AA = (IRLS - TIME)*IUNT*SRATE(K)/(ANVM + ANVX)
0015                      IF(AA.LT.0.0)AA=0.0
0016                      BB=ANVM*AA
0017                      PARMAX(J,M,4)=IROUND(BB)
0018                      BB=ANVX*AA
0019                      PARMAX(J,M,5)=IROUND(BB)
0020                  200 RETURN
0021                  101 IERR=1
0022                      RETURN
0023                  100 NN=NNB(J)
0024                      DO 10 I=1,NN
0025                      MM=PARBER(J,I,7)
0026                      IF(PARBER(J,I,10).NE.K.OR.MM.EQ.0)GO TO 10
0027                      ANVM = PARMAX(J,MM,4)
0028                      ANVX = PARMAX(J,MM,5)
0029                      IUNT = PARMAX(J,MM,9)
0030                      IRLS=TREL(J,I)
0031                      WRITE(6,505) I,IRLS,IUNT,SRATE(K),ANVM,ANVX,TCAP(K)
0032                  505 FORMAT(35X,'******* CARGO',3I10,4F10.1)
0033                      AA=(IRLS-TIME)*IUNT*SRATE(K)/(ANVM + ANVX)
0034                      IF(AA.LE.0.0)GO TO 201
0035                      BB=ANVM*AA
0036                      PARMAX(J,MM,4)=IROUND(BB)
0037                      BB=ANVX*AA
0038                      PARMAX(J,MM,5)=IROUND(BB)
0039                  201 IF(IEND.EQ.1)GO TO 10
0040                      IBTIM = TIME - PARMAX(J,MM,10)
0041                      IBRTH(K)=IBRTH(K)+IBTIM
0042                      IMUN(K)=IMUN(K)+IBTIM*PARMAX(J,MM,8)
0043                      IAUN(K)=IAUN(K)+(TIME-PARBER(J,I,11))*PARMAX(J,MM,9)
```


Information Processing Center

```
0044                IIM(K)=IIM(K)-PARMAX(J,MM,4)
0045                IEX(K)=IEX(K)-PARMAX(J,MM,5)
0046                PARMAX(J,MM,10)=TIME
0047                PARBER(J,I, 11)=TIME
0048             10 CONTINUE
0049                RETURN
0050                END
```

```
      C     FINDS CAPACITY COMPLX K, SYSTEM J AT TIME LIT GIVEN THAT
      C     AT TIME LITO ITS CAPACITY WAS TO
0001        FUNCTION TRIG(J,K,LIT,LITO,TO,IERR)
0002        INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
           CTX,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
           CNIDI,WAIT,XIDLE,RANGE,TNARR
0003        COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NM1(5),NTOT(5),N(5),NARR(5),NPR(5)
           C,TIME,TARR(50),TREL(5,20),NUMBF(50),LTYPF,LIMPO,LEXPO,LSIZE,LBUFM,
           CUBUFX,LTYPI,LTYPX,NNR(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
           C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
           C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004        COMMON/NEW/IERLY,TCAP(10),IAUN(10),MFLX(10,6),ITRND(10),IBRTH(10),
           1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
           2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005        IERR=0
0006        TRIG=TO
0007        IF(MFLX(K,2).EQ.0.OR.MFLX(K,3).EQ.0)RETURN
0008        B=2.0*RATE(K)
0009        AB= (FLOW(J,K)*SRATE(K) + RATE(K))*CAP(K)/B
0010        C= -(LIT - LITO)*B/CAP(K)
0011        IF(C.GT.0.0)IERR=1
0012        TRIG=(TO-AB)*EXP(C) + AB
0013        IF(TRIG.GT.CAP(K))GO TO 100
0014        IF(TRIG.LT.0.0)   GO TO 101
0015        RETURN
0016      100 WRITE(6,200)J,K,TRIG
0017        TRIG=CAP(K)
0018        GO TO 102
0019      101 WRITE(6,200)J,K,TRIG
0020        TRIG=0.0
0021      102 IERR=IERR+2
0022      200 FORMAT(*  **********   THIS ERROR    **********',2I10,F12.1)
0023        RETURN
0024        END
```

```
C     RESET LOADING RATES FOR TERMINALS FULL OR EMPTY
      SUBROUTINE TERR(J,K,IFUL)
      INTEGER*2 FPAR,PARMAX,TERR=4,XTIME,TIMF=4,PRA,PR,SAV1AX,SAV2AX,PRM
     CTX,PR1,PR2,TREL=4,PARMER,NCRE,NC,IDLE,KAVAL,NUMSH,TINC,TIDLE,A,X,
     GNID1,WAIT,XIDLE,RANGE,TABR
      COMMON NSYS,NY,NP,NPJ,JCAT,JFAP,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
     C,TIME,TARR(50),TREL(5,20),NUMPE(50),LTYPE,LIMPO,LEXPO,LSIZE,LBUIP,
     CLBUCA,LTYP1,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
     C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARMER(
     C5,20,11),NCRE(50),NC(50),NUMSH(50),NID1(5),WAIT(5),XIDLE(5)
      COMMON/NEW/ TERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRAC(10),IBRTH(10),
     1 AMUN(10),TIP(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
     2 NCOMP,IERR,IRRLIP,IX3,IX4,ITIME,JTIME,ALAM,A9,A8
      TIME=TERLY
      CALL CARGO(J,K,0,0,1,IE)
      FL=FLOW(J,K)
      IF(FL.EQ.0.0)GO TO 800
      SRATE(K) = RATE(K)/FL
      GO TO 801
  800 SRATE(K)=SRATE1(K)
  801 CONTINUE
      IF((IFUL.EQ.1.AND.SRATE(K).GE.0.AND.SRATE(K).LE.SRATE1(K))GO TO 10
      IF((IFUL.EQ.0.AND.SRATE(K).LE.0.AND.SRATE(K).GE.(-SRATE1(K)))GO TO
     1 11
      WRITE(6,100)K
  100 FORMAT('0',16,'  COMPLEX SRATE ERROR')
      SRATE(K)=SRATE1(K)
      GO TO 10
   11 SRATE(K)=-SRATE(K)
   10 CONTINUE
      IF(IFUL.EQ.0)GO TO 20
      MPLX(K,2)=0
      MPLX(K,3)=-100
      TCAP(K)=CAP(K)
      GO TO 30
   20 MPLX(K,2) = -100
      MPLX(K,3) = 0
      TCAP(K)=0.0
   30 MPLX(K,4) = TIME
      CALL RELEAS(J,K,0,0,IE)
      RETURN
      END
```

```
                C       CALCULATIONS FOR BERTH RELEASE - BERTH L OF SYSTEM J COMPLEX K
                C       K1 IS SHIP FOR IFUL=2 OR 3,    BERTH FOR IFUL=4
                C       DETERMINES NEXT SHIP SCHEDULED - BERTH L SYSTEM J
0001                    SUBROUTINE BERTRL(J,K1,IFUL)
0002                    INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
                       CTX,PR1,PR2,TREL*4,PARBER,NUNF,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
                       CNIDI,WAIT,XIDLL,RANGE,TNARR
0003                    COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
                       C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPC,LEXPC,LSIZE,LBUIM,
                       CLBUEX,LTYPI,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
                       C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,2C),PARBER(
                       C5,20,11),NUNF(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004                    COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
                       1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
                       2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAP,N9,N8
0005                    TIME=IERLY
0006                    IF(IFUL.EQ.4)GO TO 100
0007                    M=K1
0008                    L=PARMAX(J,K1,7)
0009                    IF(IFUL.EQ.3)GO TO 600
                C          RELEASE BERTH FROM DEFAULTED SHIP
0010                    K=PARBER(J,L,10)
0011                    PARMAX(J,K1,9) =0
0012                    PARMAX(J,K1,7) =0
0013                    IF(K1.NE.PARBER(J,L,8))GO TO 290
0014                    CALL NEXT(J,L)
0015                290 IF(PARBER(J,L,7).GT.0)RETURN
0016                    CALL CARGO(J,K,0,0,1,IE)
0017                    TO=TCAP(K)
0018                    LITO=MPLX(K,4)
0019                    TCAP(K)=TBIG(J,K,TIME,LITO,TO,IE)
0020                    MPLX(K,4)=TIME
0021                    GO TO 200
0022                100 M=PARBER(J,K1,7)
0023                    K=PARBER(J,K1,10)
                C          RELEASE SHIP,AND REASSIGN CRANES
0024                    TO=TCAP(K)
0025                    LITO=MPLX(K,4)
0026                    TCAP(K)=TBIG(J,K,TIME,LITO,TO,IE)
0027                    MPLX(K,4)=TIME
0028                    PARBER(J,K1,7)=0
0029                    L=K1
0030                    CALL CARGO(J,K,0,0,1,IE)
0031                    PARMAX(J,M,7)=-500
0032                    IF(MPLX(K,6).EQ.0)GO TO 301
0033                    MPLX(K,6)=MPLX(K,6) + PARMAX(J,M,9)
0034                    GO TO 302
0035                301 MPLX(K,6)=PARMAX(J,M,9)
0036                302 NN=NNB(J)
0037                    DO 303 I=1,NN
0038                    IF(PARBER(J,I,10).NE.K)GO TO 303
0039                    JS = PARBER(J,I,7)
0040                    IF(JS.EQ.0)GO TO 303
```

```
0041                MISS = PARMAX(J,JS,8) - PARMAX(J,JS,9)
0042                IF(MISS.LE.0)GO TO 303
0043                IAUN(K)=IAUN(K) + (IERLY - PARBER(J,I,11))*PARMAX(J,JS,9)
0044                PARBER(J,I,11) = IERLY
0045                IF(MISS.GE.MPLX(K,6))GO TO 310
0046                PARMAX(J,JS,9)=PARMAX(J,JS,8)
0047                MPLX(K,6)=MPLX(K,6) - MISS
0048            303 CONTINUE
0049                GO TO 302
0050            310 PARMAX(J,JS,9) = PARMAX(J,JS,9) + MPLX(K,6)
0051                MPLX(K,6)= 0
0052            302 CONTINUE
0053                ITRND(J)=ITRND(J) + (IERLY - PARMAX(J,M,1))
0054                JER=IERLY - PARMAX(J,M,10)
0055                IBRTH(K) = IBRTH(K) + JER
0056                IMUN(K) = IMUN(K) + JER*PARMAX(J,M,8)
0057                IAUN(K) = IAUN(K) + PARMAX(J,M,9)*(IERLY-PARBER(J,L,11))
0058                PARBER(J,L,11) = -10
                C     *********
                C     SEARCH FOR NEXT SHIP
                C     **********
0059                ISSN=PARBER(J,L,8)
0060                IF(ISSN.EQ.0)GO TO 200
0061                IF(PARMAX(J,ISSN,10).NE.-600.AND.PARMAX(J,ISSN,7).GE.0)GO TO 401
0062            200 NN=NTOT(J)
0063                IF(NN.EQ.0)GO TO 404
0064                IRLS=TREE(J,L)
0065                DO 405 I=1,NN
0066                JDP=0
0067                IIS=PARMAX(J,I,7)
0068                IF(PARMAX(J,I,10).EQ.-600.OR.IIS.LT.0)GO TO 405
0069                IF(IIS.EQ.0)GO TO 440
0070                JRLS=TREE(J,IIS)
0071                IF(JRLS.LT.TIME)JRLS=TIME
0072                JDP=IDPT(J,IIS,1,JRLS)
0073            440 IDP=IDPT(J,L,I,IRLS)
0074                IF(ISSN.NE.0.AND.IDP.GT.PARBER(J,L,9))GO TO 405
0075                IF((IDP.LT.JDP.AND.IDP.GT.0).OR.(IDP.GT.0.AND.JDP.LE.0))GO TO 420
0076            405 CONTINUE
0077            404 IF(IFUL.NE.4) RETURN
0078                GO TO 421
0079            401 I=ISSN
0080            420 IIS=PARMAX(J,I,7)
0081            451 PARMAX(J,I,7)= -L
0082                PARMAX(J,I,10)=IERLY
0083                PARBER(J,L,7) = I
0084                PARBER(J,L,11)= IERLY
0085                IF(IIS.EQ.0)GO TO 1003
0086                IF(I.EQ.PARBER(J,IIS,8))CALL NEXT(J,IIS)
0087            1003 WRITE(N8,1005)TIME,I,L
0088                INITS = PARMAX(J,I,8)
0089                IF(MPLX(K,6).GE.INITS)GO TO 450
0090                PARMAX(J,I,9)=MPLX(K,6)
```

-D28-

```
0091                    MPLX(K,6)=0
0092                    GO TO 481
0093                450 PARMAX(J,I,9)=INITS
0094                    MPLX(K,6)=MPLX(K,6) - INITS
0095                481 IIM(K) = IIM(K) + PARMAX(J,I,4)
0096                    IEX(K) = IEX(K) + PARMAX(J,I,5)
                C       SET NEW TIMES FOR TERMINAL CAPACITIES AND BERTH RELEASES
0097                421 TO=TCAP(K)
0098                    IF(MPLX(K,2).EQ.0.OR.MPLX(K,3).EQ.0)GO TO 500
0099                    T=CAP(K)
0100                    MPLX(K,2)=LITT(J,K,T,TIME,TO)
0101                    T=0.0
0102                    MPLX(K,3)=LITT(J,K,T,TIME,TO)
0103                    GO TO 501
0104                530 FL=FLOW(J,K)
0105                    IF(FL.EQ.0.0)GO TO 502
0106                    TRATE=RATE(K)/FL
0107                    GO TO 503
0108                502 TRATE=2.0*SRATE1(K)
0109                503 IF(MPLX(K,2).EQ.0)GO TO 504
0110                    IF(TRATE.GE.0.0)GO TO 505
0111                    TRATE=-TRATE
0112                    IF(TRATE.LE.SRATE1(K))GO TO 506
0113                505 SRATE(K)=SRATE1(K)
0114                    MPLX(K,3)=-100
0115                    T=CAP(K)
0116                    MPLX(K,2)=LITT(J,K,T,TIME,TO)
0117                    GO TO 501
0118                506 SRATE(K)=TRATE
0119                    GO TO 501
0120                504 IF(TRATE.LT.0.0)GO TO 507
0121                    IF(TRATE.LE.SRATE1(K))GO TO 508
0122                507 SRATE(K)=SRATE1(K)
0123                    MPLX(K,2)=-100
0124                    T=0.0
0125                    MPLX(K,3)=LITT(J,K,T,TIME,TO)
0126                    GO TO 501
0127                508 SRATE(K)=TRATE
0128                501 CALL RELEAS(J,K,0,0,IE)
0129                    RETURN
0130                600 PARMAX(J,K1,10)=0
0131                    ITS=L
0132                    NN=NNB(J)
0133                    JDP=0
0134                    DO 601 I=1,NN
0135                    IRLS=TREL(J,I)
0136                    IF(TIME.GT.IRLS)IRLS=TIME
0137                    IDP=IOPT(J,I,M,IRLS)
0138                    ISSN=PARBER(J,I,8)
0139                    IF(IDP.GT.PARBER(J,I,9).AND.ISSN.NE.0.AND.ISSN.NE.M)GO TO 601
0140                    IF((IDP.GE.JDP.OR.IDP.LE.0).AND.(IDP.LE.0.OR.JDP.GT.0))GO TO 601
0141                    L=I
0142                    JDP=IDP
```

```
0143          601 CONTINUE
0144         1005 FORMAT('0',110,'  AT BERTH  SHIP',I7,'  BERTH',I7,I15)
0145              WRITE(NB,1001)TIME,M,L,JDP
0146         1001 FORMAT('0',I10,'  IN QUEUE  SHIP',I7,'  BERTH',I7,I15)
0147              IF(JDP.EQ.0)RETURN
0148              I=M
0149              PARMAX(J,M,9)=JDP
0150              IF(PARBER(J,L,7).LE.0)GO TO 461
0151              WRITE(NB,1001)TIME,M,L
0152              IF(L.EQ.115)RETURN
0153              IF(IIS.EQ.0)GO TO 1002
0154              IF(L.EQ.PARBER(J,IIS,8))CALL NEXT(J,IIS)
0155         1002 PARMAX(J,M,7)=L
0156              PARBER(J,L,8)=M
0157              PARBER(J,L,9)=TREL(J,L)
0158              RETURN
0159          461 K=PARBER(J,L,10)
0160              CALL CARGU(J,K,J,0,1,IE)
0161              TO=TCAP(K)
0162              LITO=MPLX(K,4)
0163              TCAP(K)=TBIG(J,K,TIME,LITO,TO,IE)
0164              MPLX(K,4)=TIME
0165              GO TO 451
0166              END
```

```
C     FINDS BERTH RELEASE TIMES FO ONE OR ALL BERTHS OF A COMPLEX - K OF J
C     INPUT VARIABLE L - BERTH NO. OR IS ZERO FOR ALL BERTHS OF COMPLX K
C                    M - SHIP NO - IF ZERO WILL BE FOUND FROM L
0001      SUBROUTINE RELEAS(J,K,L,M,IERR)
0002      INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,P4,SAV1AX,SAV2AX,PRM
         CTX,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
         CNIDI,WAIT,XIDLE,RANGL,TNARR
0003      COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
         C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPC,LEXPC,LSIZE,LBUIM,
         CLBUEX,LTYPI,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
         C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(5),A(50,20),PARBER(
         C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004      COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
         1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
         2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005      IERR = 0
C        IF L & M ARE ZERO DO ALL BERTHS - IF ONLY ONE IS ZERO CALC IT FROM OTHR
0006      IF(L.EQ.0.AND.M.EQ.0)GO TO 100
0007      IF(L.EQ.0)L=-PARMAX(J,M,7)
0008      IF(M.EQ.0)M= PARBER(J,L,7)
0009      IF(L.LE.0.OR.M.LE.0)GO TO 101
0010      ANVM= PARMAX(J,M,4)
0011      ANVX= PARMAX(J,M,5)
0012      IUNT= PARMAX(J,M,9)
0013      AA = (ANVM+ANVX)/(SRATE(K)*IUNT)
0014      TREL(J,L)=IROUND(AA)+TIME
0015      RETURN
0016    101 IERR=1
0017      RETURN
0018    100 NN=NNB(J)
0019      DO 10 I=1,NN
0020      MM=PARBER(J,I,7)
0021      IF(PARBER(J,I,10).NE.K.OR.MM.EQ.0)GO TO 10
0022      ANVM = PARMAX(J,MM,4)
0023      ANVX = PARMAX(J,MM,5)
0024      IUNT = PARMAX(J,MM,9)
0025      IF(IUNT.NE.0)GO TO 506
0026      TREL(J,I)=32000
0027      GO TO 507
0028    506 AA = (ANVM+ANVX)/(SRATE(K)*IUNT)
0029      TREL(J,I)=IROUND(AA)+TIME
0030    507 WRITE(N8,505) I,TREL(J,I),IUNT,SRATE(K),ANVM,ANVX,TCAP(K)
0031    505 FORMAT(35X,'******* RELES',3I10,4F10.1)
0032     10 CONTINUE
0033      RETURN
0034      END
```

```
C       FINDS NET FLOW OF CARGO FROM SHIPS AT COMPLEX K OF SYS J
0001    FUNCTION FLOW(J,K)
0002    INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
        C IX,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
        CNIDI,WAIT,XIDLE,RANGE,TMARK
0003    COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
        C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPC,LEXPO,LSIZE,LBUIM,
        CLBUCX,LTYPI,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
        C15),PRATX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
        C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004    COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
        1 SMUNE(10),TIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
        2 NCOMP,IR(8),IRRLIN,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005      SUM=0.)
0006      NN=NTOT(J)
0007      IF(NN.EQ.0)GO TO 11
0008      DO 10 I=1,NN
0009      IF(PARMAX(J,I,7).GE.0)GO TO 10
0010      IB=-PARMAX(J,I,7)
0011      IF(IB.EQ.500)GO TO 10
0012      IF(PARBER(J,IB,10).NE.K) GO TO 10
0013      ANVH=PARMAX(J,I,4)
0014      ANVX=PARMAX(J,I,5)
0015      INTS=PARMAX(J,I,9)
0016      SUM = INTS*(ANVH-ANVX)/(ANVH+ANVX) + SUM
0017    10 CONTINUE
0018    11 FLOW=SUM
0019      RETURN
0020      END
```

```
      C       FINDS TIME WHEN CPLX K OF SYS J REACHES CAPACITY T GIVEN THAT
      C       AT TIME LITO TIS CAP WAS TO
0001     FUNCTION LITT(J,K,T,LITO,TO)
0002     INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAVIAX,SAV2AX,PRM
        CTX,PR1,PR2,TREL*4,PARBER,NONE,NO,IDLE,KAVAL,NUMSH,TINQ,TINDLE,A,X,
        CNIDI,WAIT,XIDLE,RANGE,TNARR
0003     COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
        C,TIME,TARR(50),TREL(5,20),NUMBER(50),LTYPE,LIMPO,LEXPO,LSIZE,LBUIM,
        CLBUEX,LTYPI,LTYPX,NNB(20),PARMAX(5,20,15),FPAR(50,15),SAVIAX(5,50,
        C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
        C5,20,11),NONE(50),NO(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004     COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
        1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),XIDLE(5)
        2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N9,N8
0005     AB=FLOW(J,K)*SRATE(K)
0006     DENOM=AB+RATE(K)*(1.0 - 2.0*T/CAP(K))
0007     IF(DENOM.LT.0.005.AND.DENOM.GT.-0.005)GO TO 100
0008     DENUM=AB+RATE(K)*(1.0 - 2.0*TO/CAP(K))
0009     DENOM=DENUM/DENOM
0010     IF(DENOM.01.1.001)GO TO 100
0011     DENOM=ALOO(DENOM)*CAP(K)/(2.0*RATE(K))
0012     LITT=LITO+IROUND(DENOM)
0013     RETURN
0014   100 LITT=-100
0015     RETURN
0016     END
```

```
0045                KTIM = KTIM + IIM(J)
0046                KTEX = KTEX + IEX(J)
0047                KTTL = KTTL + ITTL(J)
0048                UAV(J) = FLOAT(ITTL(J))/FLOAT(IAUN(J))
0049              501 AAR(J) = FLOAT(ITTL(J))/FLOAT(IBRTH(J))
0050                TTL = FLOAT(KTTL)
0051                UUAV = TTL / FLOAT(KAUN)
0052                AABR = TTL / FLOAT(KBRTH)
0053                AATR = TTL / FLOAT(ITURN)
0054                WRITE(N8,100) (J,ITRND(J),J=1,NSYS)
0055                WRITE(N8,101) (J,IBRTH(J),INUN(J),IAUN(J),IIM(J),IEX(J),
                   1 ITTL(J),UAV(J),AAR(J),TCAP(J),J=1,NCOMP    )
0056              100 FORMAT(1H1,20X,'CARGO THROUGHPUT STATISTICS'///' SYSTEM      TURNARO
                   1UND TIME'///(' ',16,I20))
0057              101 FORMAT(/////'0COMPLX     AT BERTH  MAX UNT HRS  TOT UNT HRS       IN
                   1PORT       EXPORT        TOTAL       UN AVG     BRTH AVG     CAPACITY
                   2   TRN AVG'                        //(' ',I6,I12,2I13,3I12,3F12.1))
0058                WRITE(N8,102)KBRTH,KHUN,KAUN,KTIM,KTEX,KTTL,UUAV,AABR,AATR
0059              102 FORMAT('0',6X,I12,2I13,3I12,2F12.1,12X,F12.1)
0060                RETURN
0061                END
```

```
                C       CALCULATES STATISTICS AT END OF DAY  -  REINITIALIZES PARAMETERS
0001                    SUBROUTINE END
0002                    INTEGER*2 FPAR,PARMAX,TARR*4,XTIME,TIME*4,PRA,PR,SAV1AX,SAV2AX,PRM
                       CTX,PRI,PR2,TREL*4,PARBER,NONE,ND,IDLE,KAVAL,NUMSH,TIND,TINDLE,A,X,
                       CNIDI,WAIT,XIDLE,RANGE,TNARR
0003                    COMMON NSYS,NY,NP,NPB,JCAT,JPAM,NN1(5),NTOT(5),N(5),NARR(5),NPR(5)
                       C,TIME,TARR(50),TREL(5,20),NUMBE(50),LTYPE,LIMPO,LEXPO,LSIZE,LBUIM,
                       CLBUEX,LTYPI,LTYPX,NNB(20),PARMAX(5,50,15),FPAR(50,15),SAV1AX(5,50,
                       C15),PRMTX(6,5),PRA(50),XTIME,KAVAL(5,20),IDLE(50),A(50,20),PARBER(
                       C5,20,11),NONE(50),ND(50),NUMSH(50),NIDI(5),WAIT(5),XIDLE(5)
0004                    COMMON/NEW/IERLY,TCAP(10),IAUN(10),MPLX(10,6),ITRND(10),IBRTH(10),
                       1 IMUN(10),IIM(10),IEX(10),CAP(10),RATE(10),SRATE(10),SRATE1(10),
                       2 NCOMP,IRRD,IRRLIM,IX3,IX4,ITIME,JTIME,ALAM,N8,N9
0005                    DIMENSION  ITTL(10),UAV(10),AUR(10)
0006                    TIME = IERLY
0007                    DO 10 I=1,NCOMP
0008                    J=MPLX(I,5)
0009                    TO=TCAP(I)
0010                    LITO=MPLX(I,4)
0011                    TCAP(I)=TBIG(J,I,TIME,LITO,TO,IE)
0012                    MPLX(I,4)=TIME
0013                 10 CALL CARGO(J,I,0,0,0,IE)
0014                    DO 11 J=1,NSYS
0015                    NN=NTOT(J)
0016                    IF(NN.EQ.0)GO TO 11
0017                    IN=0
0018                    DO 12 I=1,NN
0019                    IF(PARMAX(J,I,7).EQ.-500)GO TO 12
0020                    IN = IN + 1
0021                    DO 13 K=1,15
0022                 13 SAV1AX(J,IN,K)=PARMAX(J,I,K)
0023                    IRT = TIME - PARMAX(J,I,1)
0024                    IF(IRT.EQ.0)GO TO 12
0025                    SAV1AX(J,IN,1) =TIME
0026                    ITRND(J)=ITRND(J)+IRT
0027                 12 CONTINUE
0028                    N(J)=IN
0029                 11 CONTINUE
0030                    JTIME=JTIME+XTIME
0031                    ITURN=0
0032                    DO 500 J=1,NSYS
0033                500 ITURN=ITURN+ITRND(J)
0034                    KBRTH = 0
0035                    KMUN  = 0
0036                    KAUN  = 0
0037                    KIIM  = 0
0038                    KLEX  = 0
0039                    KTTL  = 0
0040                    DO 501 J=1,NCOMP
0041                    ITTL(J) = IIM(J) + IEX(J)
0042                    KBRTH = KBRTH + IBRTH(J)
0043                    KMUN = KMUN + IMUN(J)
0044                    KAUN = KAUN + IAUN(J)
```

II. Minimum Cost Multicommodity Flows

This program relates to the section in the report devoted to Minimum Cost Multicommodity Network Flows. In particular, we might be concerned with an optimal simultaneous routing of imports and exports. This computer program provides the opportunity to address such questions as it handles more than one commodity (imports and exports can be considered distinct commodities).

```
$JOB          RGOLDEN
C     DIMENSION CARDS, READ STATEMENTS, DEFINING TYPE VARIABLES
      DIMENSION FI(700),FI2(700),FIFJ(25),XK(700),S(700),CROSS(25,25),
     1F(5 ,60),FLOW(700),ALPHA(2),XKPL(700),D(60,60),MIN(60)
      INTEGER IFIFJ(25,2),IG(700,3),PRED(60),DEMAND(40,2),O
      REAL LCOST
      READ,N,M,NN,ND,MM,EPS,JK
      READ,(FI(I),I=1,N),(FI2(I),I=1,N)
      READ,(IFIFJ(I),I=1,NN),((IFIFJ(I,J),J=1,2),I=1,NN)
      READ,(XK(I),I=1,N),((IG(I,J),J=1,3),I=1,N)
      READ,((DEMAND(I,J),J=1,2),I=1,ND)
      KOUNT=0
      NOTE=JK+NN-1
C     CALCULATE PARTIAL DERIVATIVES OF TOTAL COST IN THE NETWORK WHICH
C     IS REPRESENTED BY A QUADRATIC COST FUNCTION
      DO 1 I=1,N
      FI2(I)=2.*FI2(I)
    1 CONTINUE
      DO 10 I=1,NN
      DO 11 J=1,NN
      CROSS(I,J)=0.
   11 CONTINUE
   10 CONTINUE
      DO 2 I=1,NN
      L=I+JK-1
      DO 4 K=1,NN
      IF(L.NE.IFIFJ(K,1))GO TO 3
      J=IFIFJ(K,2)-JK+1
      CROSS(I,J)=FIFJ(K)+CROSS(I,J)
      GO TO 4
    3 IF(L.NE.IFIFJ(K,2))GO TO 4
      J=IFIFJ(K,1)-JK+1
      CROSS(I,J)=FIFJ(K)+CROSS(I,J)
    4 CONTINUE
    2 CONTINUE
C     THIS SECTION IS USED DURING FIRST ITERATION ONLY AND GIVES US A
C     STARTING FEASIBLE FLOW
      GO TO 9998
 2999 CONTINUE
      DO 9999 I=1,N
 9999 XK(I)=XKPL(I)
C     EVALUATE PARTIAL DERIVATIVES AT A POINT XK WHICH YIELDS DISTANCES
C     ASSOCIATED TO EACH ARC
 9998 DO 111 I=1,N
      SUM=0.
      IF(I.LT.JK.OR.I.GT.NOTE)GO TO 113
      L=I-JK+1
      DO 112 J=1,N
      IF(J.LT.JK.OR.J.GT.NOTE)GO TO 112
      O=J-JK+1
      DO 6 K=1,NN
      IF((I.NE.IFIFJ(K,1).OR.J.NE.IFIFJ(K,2)).AND.(I.NE.IFIFJ(K,2).OR.J.
     1NE.IFIFJ(K,1)))GO TO 6
      SUM=SUM+CROSS(L,O)*XK(J)
      GO TO 7
    6 CONTINUE
```

```
      7     CONTINUE
    112     CONTINUE
    113     G(I)=F1(I)+F12(I)*XK(I)+SUM
    111     CONTINUE
      C     INITIALIZE ALL DISTANCES TO EITHER ZERO OR INFINITY
            DO 311 I=1,M
            DO 12 J=1,M
            IF(I.EQ.J)GO TO 1000
            D(I,J)=100000.
            GO TO 12
   1000     D(I,J)=0.
     12     CONTINUE
    311     CONTINUE
      C     NOW ASSOCIATE DISTANCE G TO THE ORDERED PAIR I,J IF AND ONLY IF
      C     ARC I,J APPEARS ON GRAPH
            DO 13 I=1,M
            DO 14 J=1,M
            DO 15 L=1,N
            IF(I.EQ.IG(L,1).AND.J.EQ.IG(L,2))D(I,J)=G(L)
     15     CONTINUE
     14     CONTINUE
     13     CONTINUE
      C     FIND SHORTEST ROUTE FROM NODE 1, THE SUPERSOURCE
            DO 21 I=1,M
     21     F(1,I)=D(1,I)
            DO 22 K=2,MM
            DO 23 I=1,M
            WIMP=100000.
            DO 24 J=1,M
            MIN(J)=F(K-1,J)+D(J,I)
     24     CONTINUE
            DO 101 J=1,M
            IF(MIN(J).LE.WIMP.AND.I.NE.J)PRED(I)=J
            RIGHTO=MIN(J)
            WIMP=AMIN1(WIMP,RIGHTO)
    101     CONTINUE
            F(K,I)=WIMP
     23     CONTINUE
            DO 5 L=1,M
            IF(K.EQ.MM)GO TO 5
            GO TO 22
      5     CONTINUE
            PRED(1)=1
      C     FOR ALL DEMAND CENTERS SUPPLIED FROM NODE 1 WE PLACE DEMAND ON
      C     EACH ARC CONTAINED IN THE SHORTEST PATH
            DO 202 L=1,N
            FLOW(L)=0.
     202    CONTINUE
            GO TO 99
     22     CONTINUE
     99     DO 103 JJ=1,2
            KDEST=DEMAND(JJ,I)
            DO 104 KK=1,M
            KKDEST=PRED(KDEST)
            DO 115 L=1,N
            IF(KKDEST.EQ.IG(L,1).AND.KDEST.EQ.IG(L,2).AND.L.LT.19.AND.L.GT.2)F
```

```
      ILOW(L)=FLOW(L)+DEMAND(JJ,2)
  115 CONTINUE
   98 IF(KKDEST.EQ.1)GO TO 103
   99 KDEST=KKDEST
  100 104 CONTINUE
  101 103 CONTINUE
      C   FIND SHORTEST ROUTE FROM NODE M, THE SUPERSINK
  102     DO 201 I=1,M
  103 201 F(1,I)=D(M,I)
  104     DO 2202 K=2,MM
  105     DO 203 I=1,M
  106     WIMP=100000.
  107     DO 204 J=1,M
  108     MIN(J)=F(K-1,J)+D(J,I)
  109 204 CONTINUE
  110     DO 2101 J=1,M
  111     IF(MIN(J).LE.WIMP.AND.I.NE.J)PRED(I)=J
  112     RIGHTO=MIN(J)
  113     WIMP=AMIN1(WIMP,RIGHTO)
  114 2101 CONTINUE
  115     F(K,I)=WIMP
  116 203 CONTINUE
  117     DO 205 L=1,M
  118     IF(K.EQ.MM)GO TO 205
  119     GO TO 2202
  120 205 CONTINUE
  121     PRED(M)=M
      C   FOR ALL DEMAND CENTERS SUPPLIED FROM NODE M PLACE DEMAND ON EACH
      C   ARC CONTAINED IN THE SHORTEST PATH
  122     GO TO 299
  123 2202 CONTINUE
  124 299 JJ=1
  125     KDEST=DEMAND(2*JJ,1)
  126     DO 2104 KK=1,M
  127     KKDEST=PRED(KDEST)
  128     DO 2115 L=1,N
  129     IF(KKDEST.EQ.IG(L,1).AND.KDEST.EQ.IG(L,2).AND.L.LT.19.AND.L.GT.2)F
     1LOW(L)=FLOW(L)+DEMAND(JJ+2,2)
  130 2115 CONTINUE
  131     IF(KKDEST.EQ.M)GO TO 2103
  132     KDEST=KKDEST
  133 2104 CONTINUE
  134 2103 CONTINUE
      C   USED TO HELP FIND INITIAL FEASIBLE SOLUTION
  135     KOUNT=KOUNT+1
  136     IF(KOUNT.GT.1)GO TO 1234
  137     DO 1235 LL=1,N
  138 1235 XK(LL)=FLOW(LL)
  139     GO TO 9998
      C   GOLDEN SECTION SEARCH TECHNIQUE
  140 1234 CONTINUE
  141     A1=0.
  142     A2=1.
  143     H=A2-A1
  144     R=.618033
  145     ALPHA(1)=A1+H*(R**2)
```

```
      ALPHA(2)=A1+R*H
  536 DO 532 J=1,2
      DO 533 I=1,N
  533 XKP1(I)=ALPHA(J)*XK(I)+(1.-ALPHA(J))*FLOW(I)
 1037 SUM21=0.
      COST=0.
      DO 5006 K=1,NN
      SUM21=SUM21+FIFJ(K)*XKP1(IFIFJ(K,1))*XKP1(IFIFJ(K,2))
 5006 CONTINUE
      DO 5001 I=1,N
      COST=COST+FI(I)*XKP1(I)+.5*FI2(I)*XKP1(I)*XKP1(I)
 5001 CONTINUE
      COST=COST+SUM21
      IF(J.EQ.2)GO TO 532
      LCOST=COST
  532 CONTINUE
      IF(LCOST.LT.COST)GO TO 534
      A1=ALPHA(1)
      H=A2-ALPHA(1)
      IF(H.LT..0001)GO TO 535
      ALPHA(1)=ALPHA(2)
      ALPHA(2)=A1+R*H
      GO TO 536
  534 A2=ALPHA(2)
      H=ALPHA(2)-A1
      IF(H.LT..0001)GO TO 535
      ALPHA(2)=ALPHA(1)
      ALPHA(1)=A1+H*(R**2)
      GO TO 536
  535 CONTINUE
C     STOPPING CRITERION
      SUM5=0.
      DO 50 I=1,N
      SUM5=SUM5+ABS(XKP1(I)-XK(I))
   50 CONTINUE
      IF(SUM5.GE.EPS)GO TO 2999
      PRINT270
  270 FORMAT('0','TOTAL SYSTEM COST=$')
      PRINT271,COST
  271 FORMAT('+',20X,F8.3///)
      PRINT272
  272 FORMAT('0','ARC FLOWS'//)
      DO 8878 II=1,N
      PRINT273,II,XKP1(II)
  273 FORMAT('0',I2,3X,F6.3)
 8878 CONTINUE
      STOP
      END
```

Modified Steepest-Descent Method

The modified steepest-descent computer code is designed to minimize (suboptimally) the total cost (fixed + variable) of a sea land transportation network. The code uses an out-of-kilter network flow algorithm to minimize (optimally) the variable costs (e.g. land transport, sea transport, port handling) for a particular configuration of port capital investments and proceeds to search through a series of predetermined capital investment options for each port in order to reduce the total cost. Reduction in capital investments are weighed against increased transportation costs and the p first stage of the code proceeds until no further reductions in capital investment can be justified economically. In the second stage the code proceeds to determine whether any trade route between a foreign port and Eastern seaboard port can be economically eliminated (saving the fixed cost associated with maintaining a trade route). When no further capital savings can be accrued the program comes to a halt.

At each step in the procedure the program outputs which capital reductions are justified, the resultant total cost, the throughput for each port and which hinterland(s) each port feeds.

```
/*EOJ ********
// *CHELST*,REGION=250K,CLASS=A
//C.SYSIN DD *                                                    OKF00030
      SUBROUTINE MAINE
      DIMENSION KL(3500),KC(3500),KU(3500),KX(3500),JJ(3500),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(3500),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9),KE(101)
       DIMENSION NOPORT(9),TRAN(18,30),F(9,3),P(9),NFIX(9,9),NF(9),NCOST
     CR(9,9),PY(9,3),NCOST1(7,3,9),NCAP1(7,3,9),KCLOSE(9,3),NTRACK(9)
      DIMENSION  MTRACK(9),NOPT9(10)
      DIMENSION MAXOPT(9)
      COMMON/KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      OKF00080
      COMMON /JL/JL/JJ/JJ/M/M/N/N/LFR/LER/KAT/KAT/KOR/KOR/KTER/KTER    OKF00090
      COMMON /MINF/MINF/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KQ/KQ/KQ/KQ/K/K    OKF00100
      COMMON/KPORT/KPORT/TRAN/TRAN/KCUM/KCUM/RUN/RUN/CAPARC/CAPARC
      COMMON /NOCOST/NOCOST/NCOSTR/NCOSTR/NOCOST/NOCOST/LOCOST/LOCOST/M
     CAX/MAX/NOPORT/NOPORT/NCHECK/NCHECK/NOARCS/NOARCS
      COMMON/NOHELP/NOHELP
      INTEGER TRAN,TALTER,P,SOURC1,SOURC2,P,ZERO,TCOMPU,UTE,CAPARC,TR,F,
     -PY,OUT1,OUT2,OPEN
      DATA NOPT9/10*3/
      DATA MAXOPT/9*1/
      DATA F/1500000,4700000,3000000,1500000,5000000,7800000,17000000,15
     C000000,4200000,750000,2350000,1500000,750000,2500000,3900000,85000
     C00,7500000,2100000,9*0/
      DATA NTRACK,MTRACK/18*0/
      DATA NCOST1/10,15,15,20,45,125,380,10,15,15,20,45,125,380,10,15,15
     C,20,45,125,380,5*3,14,60,5*3,14,60,5*3,14,60,10,3*13,18,70,175,10,
     C3*13,18,70,175,10,3*13,18,70,175,10,15,15,20,45,125,380,10,15,15,2
     C0,45,125,380,10,15,15,20,45,125,380,2,3,3*4,10,69,2,3,3*4,10,69,
     C 2,3,3*4,10,69,2,5,8,8,9,15,50,2,5,8,8,9,15,50,4*3
     C,7,29,40,4*3,7,29,40,4*3,7,29,40,2*3,4,4,8,32,85,2*3,4,4,8,32,85,
     C2*3,4,4,4,32,85,5,10,10,20,40,150,167,5,10,10,20,40,150,167,5,10,
     C10,20,40,150,167/
      DATA NCAP1/5*100000,40000,50000,5*50000,20000,25000,7*0,7*200000,
     C7*100000,7*0,5*200000,100000,40000,5*100000,50000,20000,7*0,100000
     C,4*100000,40000,50000,5*50000,20000,25000,7*0,2*4000000,2*2000000,
     C2*4000000,800000,2*2000000,2*1000000,2*2000000,400000,7*0,400000,3
     C00000,100000,2*200000,400000,100000,200000,150000,50000,2*100000,2
     C00000,50000,7*0,6*500000,300000,6*250000,150000,7*0,4*500000,2*250
     C000,200000,4*250000,2*125000,100000,7*0,400000,100000,300000,10000
     C0,300000,40000,60000,200000,50000,150000,50000,150000,20000,30000,
     C7*0/
      DATA KCLOSE/27*11/
      DATA P/4HP 1 ,4HP 2 ,4HP 3 ,4HP 4 ,4HP 5 ,4HP 6 ,4HP 7 ,4HP 8 ,4HP
     C 9 /
      DATA TALTER,R,SOURC1,SOURC2,ZERO/4HALTE,2HR ,4HS   ,2*4H    /
      DATA TCOMPU,UTE/4HCOMP,4HUTE /
      DATA L,NALTER,NF,PY,NFIX/11*0,9*1,18*0,81*200000/
      NOHELP=0
      LCOUNT=1
      LOCOST=400000000
      DO 731 IJK=1,9
  731 NOPORT(IJK)=11
      NOCOST=LOCOST
      RUN=1
      NOARCS=7
      KPORT=1
      NOPT1=1
      DO 9=1
```


Information Processing Center

```
C   SYSTEM INPUT TAPE                                                   OKF00110
      KI=5
C   SYSTEM OUTPUT TAPE                                                  OKF00130
      KO=6
C   CARD PUNCH                                                          OKF00150
      KO(1)=7                                                           OKF00160
C   RESERVED OUTPUT TAPE                                                OKF00170
      KO(2)=3                                                           OKF00180
C   RESERVED INPUT TAPE                                                 OKF00190
      KO(3)=2                                                           OKF00200
C   MAXIMUM ARCS                                                        OKF00210
      KO(4)=3500
C   MAXIMUM NODES                                                       OKF00230
      KO(5)=500
      KO(6)=0                                                           OKF00250
      KO(9)=0                                                           OKF00260
      IFIN=32767                                                        OKF00270
  100 CALL PREDAT(KS)                                                   OKF00280
  101 IF(KS.NE.0) GO TO 1                                               OKF00290
      CALL ARCASY(LO)
      CALL MAKEJL                                                       OKF00310
      IF(LER.GE.KO(4)) GO TO 88
      LER=LER*KO(2)                                                     OK60  0
      IF(LO.EQ.0) GO TO 1
      CALL NODASY
    1 CALL READER
      CALL TRANSL
      IF(LER.NE.0) GO TO 88
      I=0
      KUP=1
      KE(101)=0
      DO 26 K=1,N
      IF(K.LT.KUP) GO TO 38
      I=I+1
      KUP=LOFCR(IC(I+1))
   38 CALL KILTER(I)
      IF(LER.EQ.0) GO TO 26
      IF(LER.NE.107) GO TO 24
      KE(101)=KE(101)+1
      KE=KE(101)
      IF(KE(101).GT.100) GO TO 26
      KE(KE)=K
   24 CONTINUE
C   COMPLETED CHECKING ALL ARCS
      LER=0
   99 CALL OUTPUT(KE)
      WRITE(6,999) KCUM,((IKCLOSE(IJKL,IJKLM),IJKL=1,9),IJKLM=1,3)
  999 FORMAT(I10,27I3)
      NOCOST=KCUM
      DO 997 IJK=1,9
      NOIOPT=NOPT9(IJK)
      DO 998 IJKL=1,NOIOPT
  998 NOCOST=NOCOST+PY(IJK,IJKL)*(F(IJK,IJKL)+NF(IJK))
  997 CONTINUE
      WRITE(6,57) NOCOST,(NOPORT(IJKL),IJKL=1,9),NOCOST
   57 FORMAT(20H0 THE TOTAL COST IS ,I10,11H AND PORTS ,8(I3,1H,),4HAND
     C,I5,12H ARE CLOSED.,I10)
      CALL TIMING(ICPU,IEXCP)
      WRITE(6,8888) ICPU,IEXCP
 8888 FORMAT(1H0,2I10)
```

```
      IF (LCOUNT.EQ.3) GO TO 792
      IF(KPORT.EQ.10) GO TO 700
      IF(NOHELP.EQ.1)  GO TO 10
      IF(NOHELP.EQ.1)  KCLOSE(KPORT,NOPT1)=1
      IF(MOCOST.GE.NOCOST) KCLOSE(KPORT,NOPT1)=1
      IF(MOCOST.GE.LOCOST) GO TO 10
      MPORT=KPORT
      MOPT1=NOPT1
      LOCOST=MOCOST
   10 NALTER=0
 6000 IF(KPORT.EQ.1.AND.NOPT1.EQ.1) NOCOST=LOCOST
      IF(L.EQ.10.AND.LCOUNT.EQ.1) GO TO 78
      IF(L.EQ.10.AND.LCOUNT.EQ.2) GO TO 79
    61 IF(NOPT1.NE.NOPT9(KPORT)) GO TO 11
   610 KPORT=KPORT+1
      NOPT1=1
    11 NOPT1=NOPT1+1
      IF (KPORT.EQ.10) GO TO 65
      IF(NOPORT(KPORT).NE.11.AND.NCOUNT.EQ.2) GO TO 610
      IF(NOPT1.GT.NOPT9(KPORT).AND.NCOUNT.EQ.2) GO TO 610
      IF(NOPT1.GT.NOPT9(KPORT)) GO TO 62
      IF(NOPORT(KPORT).NE.11) GO TO 62
      IF(KCLOSE(KPORT,NOPT1).NE.11) GO TO 62
      NALTER= NALTER+1
      PY(KPORT,MAXOPT(KPORT))=0
      PY(KPORT,NOPT1)=1
      IN1= NOARCS*(NALTER-1)+2
      IN2=NOARCS*NALTER+1
      DO 250 NCLOSE=IN1,IN2
      TRAN(1,NCLOSE)=TALTER
      TRAN(2,NCLOSE)=P
      TRAN(3,NCLOSE)=SOURC1
      TRAN(4,NCLOSE)=SOURC2
      TRAN(5,NCLOSE)=P(KPORT)
      TRAN(6,NCLOSE)= ZERO
      TRAN(7,NCLOSE)=NCLOSE-IN1+1
      TRAN(8,NCLOSE)=NCOST1(NCLOSE-IN1+1,NOPT1,KPORT)
  250 TRAN(9,NCLOSE)=NCAP1(NCLOSE-IN1+1,NOPT1,KPORT)
      JUMP=0
      GO TO 63
   62 JUMP=1
   63 IF(MCOUNT.EQ.2) GO TO 65
      IF(NOPT1.NE.1) PY(KPORT,NOPT1-1)=0
      MCOUNT=2
   65 IF(NOPT1.NE.2) GO TO 66
      IF(KPORT.EQ.1) NCOUNT=2
      IF(NCOUNT.EQ.2) GO TO 660
      NCOUNT=2
C     OPEN PORT KPORT-1
      NALTER=NALTER +1
      PY(KPORT-1,NOPT9(KPORT-1))=0
      PY(KPORT-1,MAXOPT(KPORT-1))=1
      OUT1=NOARCS*(NALTER-1)+2
      OUT2=NOARCS*NALTER+1
      DO 300 OPEN=OUT1,OUT2
      TRAN(1,OPEN)=TALTER
      TRAN(2,OPEN)=P
      TRAN(3,OPEN)=SOURC1
      TRAN(4,OPEN)=SOURC2
      TRAN(5,OPEN)=P(KPORT-1)
```

```
      TRAN(6,OPEN)=ZERO
      TRAN(7,OPEN)=OPEN-OUT1+1
      TRAN(8,OPEN)=NCOST1(OPEN-OUT1+1,MAXOPT(KPORT-1),KPORT-1)
  300 TRAN(9,OPEN)= NCAP1(OPEN-OUT1+1,MAXOPT(KPORT-1),KPORT-1)
      IF(KPORT.EQ.10) GO TO 661
  660 IF (JUMP.EQ.1) GO TO 61
  661 CONTINUE
      TRAN(1,NOARCS*NALTER+2)=TCOMPU
      TRAN(2,NOARCS*NALTER+2)=UTE
      KS=1
      MCOUNT=0
      NCOUNT=0
      GO TO 101
  700 KPORT=1
      NOPT1=1
      KS=1
      L=L+1
      IF(NOCOST.EQ.LOCOST.AND.LCOUNT.EQ.1) GO TO 78
      IF(NOCOST.EQ.LOCOST.AND.LCOUNT.EQ.2) GO TO 79
C     CLOSE PORT MPORT PERMANENTLY
      NCPORT(MPORT)=MPORT
      IF(MOPT1.NE.NOPTP(MPORT)) PY(MPORT,MOPT1)=1
      PY(MPORT,MAXOPT(MPORT))=0
      KCLOSE(MPORT,MOPT1)=2
      MAXOPT(MPORT)=MOPT1
      LIN1=2
      LIN2=NOARCS+1
      DO 400 NCLOSE=2,LIN2
      TRAN(1,NCLOSE)=TALTER
      TRAN(2,NCLOSE)=R
      TRAN(3,NCLOSE)=SOURC1
      TRAN(4,NCLOSE)=SOURC2
      TRAN(5,NCLOSE)=P(MPORT)
      TRAN(6,NCLOSE)=ZERO
      TRAN(7,NCLOSE)=NCLOSE-1
      TRAN(8,NCLOSE)=NCOST1(NCLOSE-1,MOPT1,MPORT)
  400 TRAN(9,NCLOSE)=NCAP1(NCLOSE-1,MOPT1,MPORT)
      JUMP=0
      NALTER=1
      NCOUNT=2
      MCOUNT=2
      GO TO 6000
   24 WRITE(KO,54)
      LL=LADDR(IJ(K))
      WRITE(KO,55) NN(2*I-1),NN(2*I),NN(2*LL-1),NN(2*LL)                OKF00610
      GO TO 99
   88 WRITE(KO,56)
      STOP
   51 FORMAT(A4)
   54 FORMAT(24HOOVERFLOW IN NODE PRICES)                               OKF00660
   55 FORMAT(23HORUN TERMINATED AT ARC ,4A4)                            OKF00670
   56 FORMAT(37HORUN TERMINATED DUE TO ERRORS IN DATA)                  OKF00680
   79 L=0
      WRITE(6,800)
  800 FORMAT(12H1 STAGE TWO )
      LCOUNT=2
      KPORT=1
      NOPT1=1
      MCOUNT=0
      NCOUNT=0
```


Information Processing Center

```
      DO 783 IJKL=1,9
      DO 782 IJK=1,MAX
  782 NF(IJKL)=NF(IJKL)*NFIX(IJKL,IJK)
      NO1OPT=NOPT9(IJKL)
      DO 784 IJKLM=1,NO1OPT
      IF(KCLOSE(IJKL,IJKLM).NE.2) KCLOSE(IJKL,IJKLM)=11
      MTRACK(IJKL)=MTRACK(IJKL)*PY(IJKL,IJKLM)
  784 NOCOST=NOCOST+PY(IJKL,IJKLM)*NF(IJKL)
  783 IF(MTRACK(IJKL).NE.0) NOPORT(IJKL)=11
      LOCOST=NOCOST
      GO TO 61
   79 DO 791 IJKL=1,9
      NO1OPT=NOPT9(IJKL)-1
      DO 790 IJKLM=1,NO1OPT
  790 MOCOST=MOCOST-PY(IJKL,IJKLM)*NF(IJKL)
      NF(IJKL)=0
  791 CONTINUE
      DO 796 IJKL=1,9
      NO1OPT=NOPT9(IJKL)
      DO 794 IJK=1,NO1OPT
      WRITE(6,803) MTRACK(IJKL)
  803 FORMAT(1H0,I10)
  794 MTRACK(IJKL)=MTRACK(IJKL)*PY(IJKL,IJK)
      IF(MTRACK(IJKL).NE.0) NOPORT(IJKL)=11
      DO 795 IJK=1,MAX
  795 NCOSTR(IJKL,IJK)=MTRACK(IJKL)
  796 CONTINUE
      WRITE(6,801)
  801 FORMAT(13H1 STAGE THREE)
      WRITE(6,802) MOCOST,(MTRACK(IJKL),IJKL=1,9)
  802 FORMAT(1H0,I11,9I8)
      LCOUNT=3
  792 CALL TRADER
      IF (NCHECK.EQ.1) GO TO 80
      GO TO 101
   80 RETURN
      END
      SUBROUTINE TRADER
      DIMENSION NCOSTR(9,9),NFIX(9,9),NOPORT(9),NOPOR2(9,9),KOPEN3(9,9),
     CTRAN(18,30),KOPEN1(9),NCOST2(9,9,2)
      DIMENSION NCAPAR(9,9,2),ND(9),NDO(9),NR(9),NRO(9)
      COMMON /NCAPAR/NCAPAR/KCUM/KCUM/NCOST2/NCOST2
      COMMON /MOCOST/MOCOST/NCOSTR/NCOSTR/NOCOST/NOCOST/LOCOST/LOCOST/M
     CAX/MAX/NOPORT/NOPORT/NCHECK/NCHECK/TRAN/TRAN/NOARCS/NOARCS
      COMMON/NOHELP/NOHELP
      INTEGER TALTER,R,TCOMPU,UTE,TRAN,OPEN
      DATA KLPORT,L,KOUNT,KROOT1,KROOT2,NALTER/2*1,4*0/
      DATA TALTER,R,TCOMPU,UTE/4HALTE,2HR ,4HCOMP,4HUTE /
      DATA HR/4HR 1 ,4HR 2 ,4HR 3 ,4HR 4 ,4HR 5 ,4HR 6 ,4HR 7 ,4HR 8 ,4H
     CR 9 /
      DATA ND/4HD 1 ,4HD 2 ,4HD 3 ,4HD 4 ,4HD 5 ,4HD 6 ,4HD 7 ,4HD 8 ,4H
     CD 9 /
      DATA KOPEN3,NOPOR2,KOPEN1/81*11,81*11,9*11/
      DATA NRO/2H1 ,2H2 ,2H3 ,2H4 ,2H5 ,2H6 ,2H7 ,2H8 ,2H9 /
      DATA NDO/2H1 ,2H2 ,2H3 ,2H4 ,2H5 ,2H6 ,2H7 ,2H8 ,2H9 /
      DATA NFIX/81*200000/
      NCOUNT=0
      DO 2 ILN=1,MAX
      DO 2 ILM=1,9
    2 MOCOST=MOCOST+NCOSTR(ILM,ILN)*NFIX(ILM,ILN)
```

```
      2 CONTINUE
        WRITE(6,900) KCUM,MOCOST,KLPORT,KROOT1,L
  900 FORMAT(9H0KCUM IS ,I10,11H MOCOST IS ,I10,7H KLPORT,I3,7H KROOT1,
     C I3,2H L,I3)
        KOUNT=KOUNT+1
        IF (KOUNT.EQ.1) GO TO 11
        IF (KLPORT.EQ.10) GO TO 700
        IF(NOHELP.EQ.1)  KOPEN3(KLPORT,KROOT1)=1
        IF(NOHELP.EQ.1)  GO TO 10
        IF(MOCOST.GE.NOCOST) KOPEN3(KLPORT,KROOT1)=1
        IF (LOCOST .LE. MOCOST) GO TO 10
        MPORT=KLPORT
        MROOT1=KROOT1
        LOCOST=MOCOST
   10 NALTER=0
 6000 IF(KLPORT.EQ.1.AND.KROOT1.EQ.0) NOCOST=LOCOST
        IF (L.EQ.81) GO TO  88
   61 IF (KROOT1.NE.MAX) GO TO 11
  610 KLPORT=KLPORT+1
        KROOT1=0
   11 KROOT1=KROOT1+1
        IF(KLPORT.EQ.10) GO TO 65
        IF(NOPORT(KLPORT).NE.11.AND.NCOUNT.EQ.2) GO TO 610
        IF(KOPEN1(KLPORT).NE.11.AND.NCOUNT.EQ.2) GO TO 610
        IF(NOPORT(KLPORT).NE.11.OR.KOPEN1(KLPORT).NE.11) GO TO 62
        IF(NOPOR2(KLPORT,KROOT1).NE.11.OR.KOPEN3(KLPORT,KROOT1).NE.11) GO
     C TO  62
C       CLOSE PORT TRADE ROUTE TEMPORARILY
        NALTER= NALTER+1
        NCOSTR(KLPORT,KROOT1)=0
        NCLOS1=NALTER*2
        NCLOS2=NALTER*2+1
        DO 250 NCLOSE=NCLOS1,NCLOS2
        TRAN(1,NCLOSE)=TALTER
        TRAN(2,NCLOSE)=R
        TRAN(3,NCLOSE)=ND(KLPORT)
        TRAN(4,NCLOSE)=NDO(KROOT1)
        TRAN(5,NCLOSE)=NR(KLPORT)
        TRAN(6,NCLOSE)=NRO(KROOT1)
        TRAN(7,NCLOSE)=NCLOSE-NCLOS1+1
        TRAN(8,NCLOSE)=0
  250 TRAN(9,NCLOSE)=0
        JUMP=0
        GO TO 65
   62 JUMP=1
   65 IF(NCOUNT.EQ.2) GO TO 66
        IF(KROOT1-1.EQ.0.AND.KLPORT-1.EQ.0) NCOUNT=2
        IF(KROOT1-1.EQ.0.AND.KLPORT-1.EQ.0)GO TO 66
        IF(KROOT1-1.NE.0) GO TO 720
        KLP=KLPORT-1
        KRO1=MAX
        GO  TO 725
  720 KLP=KLPORT
        KRO1=KROOT1-1
C       REOPEN PORT-TRADE ROUTE COMBINATION
  725 NCOUNT=2
        NALTER=NALTER+1
        NCOSTR(KLP,KRO1)=1
        NOPEN1=2*NALTER
        NOPEN2=2*NALTER+1
```


Information Processing Center

```
      DO 300 OPEN=NOPEN1,NOPEN2
      TRAN(1,OPEN)=TALTER
      TRAN(2,OPEN)=R
      TRAN(3,OPEN)=ND(KLP)
      TRAN(4,OPEN)=NDO(KRO1)
      TRAN(5,OPEN)=NP(KLP)
      TRAN(6,OPEN)=NRO(KRO1)
      TRAN(7,OPEN)=OPEN-NOPEN1+1
      TRAN(8,OPEN)=NCOST2(KLP,KRO1,OPEN-NOPEN1+1)
  300 TRAN(9,OPEN)=NCAPAR(KLP,KRO1,OPEN-NOPEN1+1)
      GO TO 66
      IF (KLPORT.EQ.10) GO TO 661
      IF (JUMP .EQ. 1) GO TO 61
  661 CONTINUE
      TRAN(1,2*NALTER+2)=TCOMPU
      TRAN(2,2*NALTER+2)=UTF
      KS=1
      RETURN
  700 KLPORT=1
      KS=1
      KROOT1=0
      KROOT2=0
      IF (NOCOST.EQ.LOCOST) GO TO 80
      NOPOR2(MPORT,MROOT1)=1
C       CLOSE PORT-TRADE ROUTE PERMANENTLY
      NOCOST=LOCOST
      NCOST2(MPORT,MROOT1)=0
      DO 850 NCLOSE=2,3
      TRAN(1,NCLOSE)=TALTER
      TRAN(2,NCLOSE)=R
      TRAN(3,NCLOSE)=ND(MPORT)
      TRAN(4,NCLOSE)=NDO(MROOT1)
      TRAN(5,NCLOSE)=NP(MPORT)
      TRAN(6,NCLOSE)=NRO(MROOT1)
      TRAN(7,NCLOSE)=NCLOSE-1
      TRAN(8,NCLOSE)=0
  850 TRAN(9,NCLOSE)=0
      L=L+1
      JUMP=0
      NALTER=1
      NCOUNT=2
      GO TO 5000
   80 NCHECK=1
      RETURN
      END
      SUBROUTINE OUTPUT(KZ)                                          OKF05000
      DIMENSION KL(3500),KC(3500),KU(3500),KK(3500),JI(3500),
     =NP(1000),NP(500),IJ(7000),IL(501),JL(3500),NL(500)
      DIMENSION LC(9),KA(19,2),KQ(9)                                 OKF05030
      DIMENSION K7(101)                                              OKF05040
      DIMENSION L7(3500)
      INTEGER SUMY(100),SUMC(100)
      COMMON /KL/KL/KC/KC/KU/KU/KK/KK/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL  OKF05050
      COMMON /JL/JL/JI/JI/M/M/N/N/LFR/LFR/KAT/KAT/KOR/KOR/KTER/KTFR  OKF05060
      COMMON /MINE/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K  OKF05070
      COMMON /KCUM/KCUM
      COMMON/NOHELP/NOHELP
      COMMON /KPORT/KPORT
      DATA KILT,BLANK,IEN/1HK,1H ,1HN/                               OKF05080
      NOHELP=0
      KQ1=KQ(1)                                                      OKF05090
```


Information Processing Center

```
      KCUM=0                                                            OKF05100
      IF(KZ(101).NE.0) GO TO 10                                         OKF05110
      IF(LFF.NE.0) GO TO 30                                             OKF05120
      MZ=KILT                                                           OKF05130
      GO TO 100                                                         OKF05140
   10 IF(LFF.NE.0) GO TO 18                                             OKF05150
      NOHELP=1
      WRITE(KO,99) KZ(101)
   18 KZ(101)=MIN0(KZ(101),100)                                         OKF05170
   30 MZ=BLANK                                                          OKF05180
  100 KZ=KU(2)                                                          OKF05190
      IF(LC(2).EQ.0) GO TO 12                                           OKF05200
      WRITE(KO,90) (KA(I,2),I=1,18)                                     OKF05210
   12 IF(LC(1).EQ.0) GO TO 41                                           OKF05230
      IF(KU(6).NE.0) GO TO 24                                           OKF05240
      KU(6)=1                                                           OKF05250
      REWIND KZ                                                         OKF05260
   24 WRITE(KZ,90) (KA(I,2),I=1,18)                                     OKF05270
   41 IF(KU(7).EQ.0) GO TO 7                                            OKF05290
    7 L=1                                                               OKF05310
      LL=1                                                              OKF05320
      DO 3 I=1,M                                                        OKF05330
      LUP=LOFCP(IL(I))                                                  OKF05340
  302 IF(L.GE.LUP) GO TO 3
      LU=LADDR(IL(L))                                                   OKF05360
      LLC=KC(L)                                                         OKF05370
      KC(L)=KC(L)-NP(I)+NP(LU)                                          OKF05380
      KX(L)=KX(L)*KL(L)                                                 OKF05390
      KU(L)=KU(L)*KL(L)                                                 OKF05400
      LZ(L)=KX(L)*KC(L)
      KCUM=KCUM+LZ(L)
      MX=MZ
      IF(KZ(101).LE.0) GO TO 16                                         OKF05430
      IF(KZ(LL).NE.L) GO TO 16                                          OKF05440
      KZ(101)=KZ(101)-1                                                 OKF05450
      LL=LL+1                                                           OKF05460
      MX=IFN                                                            OKF05470
   16 IF(LC(1).EQ.0) GO TO 51                                           OKF05480
      WRITE(KZ,93) NN(2*I-1),NN(2*I),NN(2*LU-1),NN(2*LU),KC(L),KU(L),   OKF05490
     1KL(L),KX(L),LZ(L),MX                                              OKF05500
   51 IF(KU(7).EQ.0) GO TO 56
   56 IF(LC(2).EQ.0) GO TO 333                                          OKF05550
      WRITE(KO,93)  NN(2*I-1),NN(2*I),NN(2*LU-1),NN(2*LU),KC(L),        OKF05560
     1KU(L),KL(L),KX(L),LZ(L)
  333 L=L+1                                                             OKF05580
      GO TO 302                                                         OKF05590
    3 CONTINUE                                                          OKF05600
      IF(LC(3).EQ.0) GO TO 15                                           OKF05610
      IF(LC(1)+LC(2).EQ.0) GO TO 27                                     OKF05620
      IF(LC(1).EQ.0) GO TO 203                                          OKF05630
      WRITE(KZ,96)                                                      OKF05640
  203 IF(LC(2).EQ.0) GO TO 115                                          OKF05650
      WRITE(KO,96)                                                      OKF05660
  115 DO 200 I=1,M                                                      OKF05670
      IF(LC(1).EQ.0) GO TO 95                                           OKF05680
      WRITE(KZ,95) NN(2*I-1),NN(2*I),NP(I)                              OKF05690
   95 IF(LC(2).EQ.0) GO TO 200                                          OKF05700
      WRITE(KO,95) NN(2*I-1),NN(2*I),NP(I)                              OKF05710
  200 CONTINUE                                                          OKF05720
   15 IF(LC(3).EQ.0) GO TO 27                                           OKF05730
```

```
      WRITE(K2,97)                                                      OKF05740
   27 IF(KQ(7).EQ.0) GO TO 57                                           OKF05750
   57 IF(LC(2).EQ.0) GO TO 77                                           OKF05780
      WRITE(KQ1,97)                                                     OKF05790
   77 WRITE(KQ,92) (LC(I),I=5,8)
      RETURN
   88 FORMAT(24HOTHIS RUN OUTPUT TO TAPE )                             OKF05820
   99 FORMAT(25HOTHIS RUN OUTPUT TO PUNCH)                             OKF05830
   90 FORMAT(1HA4/4HARCS22X,4HCOST5X,5HUPPER5X,5HLOWER6X,4HFLOW7X,
     15HTCOST)
   91 FORMAT(1H1]A4/5H ARCS1AX,4HCOST6X,5HUPPER6X,5HLOWER7X,4HFLOW7X,
     15HTCOST9X,3HPI19X,3HPI2AX,4HCRAR/1X)
   92 FORMAT(1AHNO OF BREAKTHRUS=I12,22H, NO OF NONBREAKTHRUS=I12,18H, OKF05870
     1NO OF X CHANGES=I12,/42H NO OF NODES FROM WHICH LABELING WAS DONE=OKF05880
     2I12)                                                              OKF05890
   93 FORMAT(6X,2(A4,A2),2X,4I10,I12,1X,A1)                             OKF05900
   94 FORMAT(2(1X,A4,A2),4(1X,I10),4I12,1X,A1)                          OKF05910
   95 FORMAT(6X,A4,A2,6X,I12)                                           OKF05920
   96 FORMAT(6HNODES )                                                  OKF05930
   97 FORMAT(3HEND)                                                     OKF05940
   98 FORMAT(4HOEND)                                                    OKF05950
   99 FORMAT(1HO15,23H ARCS ARE OUT OF KILTER)                          OKF05960
      END                                                               OKF05980
      SUBROUTINE READER
      DIMENSION KL(3500),KC(3500),KU(3500),KX(3500),JI(3500),
     -NU(1000),NP(500),IJ(7000),IL(501),JL(3500),NL(500)
      DIMENSION TRAN(18,30)
      DIMENSION LC(9),KA(18,2),KQ(9)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER
      COMMON /MINE/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K
      COMMON /RUN/RUN
      COMMON /TRAN/TRAN
      INTEGER TRAN,TALTER,R,SOURC1,SOURC2,P,ZERO,TCOMPU,UTE,CAPARC,TR
       INTEGER TAPE1,PUNCH1,NODES1,PRINT1
       INTEGER ALTER,OUTPUT,COMPUT,TAPE,PUNCH,NODES,PRINT
      DATA TAPE1,PUNCH1,NODES1,PRINT1/4HAPE ,4HUNCH,4HODES,4HRINT/
      DATA ALTER,OUTPUT,COMPUT/4HALTE,4HOUTP,4HCOMP/
      DATA TAPE,PUNCH,NODES,PRINT/4H TAP,4H PUN,4H NOD,4H PR1/
      DATA NCOUNT/0/
      KCARD=1
      IF (RUN.GT.1) GO TO  51
    5 READ(KI,95) (KA(I,1),I=1,18)
      GO TO  52
   51 DO 1 I=1,18
    1 KA(I,1)=TRAN(I,KCARD)
      KCARD=KCARD+1
   52 IF(KA(1,1).EQ.ALTER) GO TO 18
      IF(KA(1,1).EQ.OUTPUT) GO TO 119
      IF(KA(1,1).EQ.COMPUT) GO TO 18
      WRITE(KO,96) (KA(I,1),I=1,18)
      DO 15 I=1,18
   15 KA(I,2)=KA(I,1)
      GO TO 5
   18 WRITE(KO,97)
      STOP
      WRITE(KO,96) (KA(I,1),I=1,18)
      LER=1
      IF(KA(1,1).EQ.COMPUT) GO TO 111
   20 IF (RUN.GT.1) GO TO  53
```

```
      READ(KI,90) (KA(I,1),I=1,11)
      GO TO 54
   53 DO 2 I=1,11
    2 KA(I,1)=TRAN(I,KCARD)
      KCARD=KCARD+1
   54 IF(KA(1,1).EQ.ALTER) GO TO 140
      IF(KA(1,1).EQ.OUTPUT) GO TO 121
      IF(KA(1,1).EQ.COMPUT) GO TO 111
      GO TO 200
C          COMPUTE
  111 CONTINUE
      RUN=RUN+1
  999 RETURN
C          SET OUTPUT CONTROL
  119 CONTINUE
      L=5
      IF(KA(3,1).EQ.TAPE1) L=1
      IF(KA(3,1).EQ.PUNCH1) L=2
      IF(KA(3,1).EQ.NODES1) L=3
      IF(KA(3,1).EQ.PRINT1) L=4
      GO TO 80
  121 WRITE(KO,88) (KA(I,1),I=1,6)
  120 L=5
      IF(KA(3,1).EQ.TAPE) L=1
      IF(KA(3,1).EQ.PUNCH) L=2
      IF(KA(3,1).EQ.NODES) L=3
      IF(KA(3,1).EQ.PRINT) L=4
   80 IF(L-4) 81,85,200
   81 LC(L)=1
      GO TO 20
   85 KC(7)=1
      GO TO 20
C          ALTER
  140 NCOUNT=NCOUNT+1
      IF(KA(7,1).GT.0) GO TO 142
      KA(7,1)=1
  142 IF(NCOUNT.LE.22) WRITE(KO,91) (KA(I,1),I=1,11)
      N1=NODENO(KA(3,1),KA(4,1))
      N2=NODENO(KA(5,1),KA(6,1))
      IF(N1.GT.M) GO TO 144
      IF(N2.LE.M) GO TO 145
  144 WRITE(KO,92)
      STOP
      LER=1
      GO TO 20
  145 L1=LOFOR(IL(N1))
      L2=LOFOR(IL(N1+1))-1
      IF(L2.LT.L1) GO TO 144
      DO 147 LL=L1,L2
      IF(LABOR(IJ(LL)).NE.N2) GO TO 147
      KA(7,1)=KA(7,1)-1
      IF(KA(7,1).EQ.0) GO TO 149
  147 CONTINUE
      GO TO 144
  149 KU(LL)=KA(9,1)
      KC(LL)=KA(8,1)
      GO TO 20
C          CARD PUNCHING ERROR
  200 LER=1
      WRITE(KO,87) (KA(I,1),I=1,6)
```

```
      STOP
      GO TO 20
   87 FORMAT(23H ILLEGAL CONTROL CARD (3(A4,A2),1H))
   88 FORMAT(1H03(A4,A2))
   90 FORMAT(3(A4,A2),I2,4I10)
   91 FORMAT(1H0,3(A4,A2),I2,4I10)
   92 FORMAT(47H0THE ARC ON THE ABOVE ALTER CARD IS NOT IN CORE)
   95 FORMAT(18A4)
   96 FORMAT(1H01BA4)
   97 FORMAT(47H0OUTPUT CONTROL CARD MISSING OR OUT OF SEQUENCE)        OKF03970
      END
      SUBROUTINE APCASY(LL)
      DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)
      DIMENSION NCAPAR(9,9,2),NCOST2(9,9,2)
      DIMENSION ND(9)
      DIMENSION KE(2),KF(2),KD(2)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      OKF01510
      COMMON /JL/JL/JI/JI/M/M/N/N/LEP/LER/KAT/KAT/KOR/KOR/KTER/KTER     OKF01520
      COMMON /MINF/MINF/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K     OKF01530
      COMMON /CAPARC/CAPARC/NOARCS/NOARCS/NCOST2/NCOST2
      COMMON /MAX/MAX/NCAPAR/NCAPAR
      DATA MPORT,MPOOT,MULAR1,NCOUNT,ND/2*1,2*0,4HD 1 /
       INTEGER END,NODES,BLANK
      DATA END,NODES,BLANK/4HEND ,4HNODE,4H    /
      LER=0
      KF(1)=0
      KF(2)=0
      M=0
      N=1
    6 READ(KOR,90) KD(1),KD(2),KE(1),KE(2),IJ(2*N-1),IJ(2*N),(KA(I,1),I=OKF01610
     1,4)
      IF(KD(1).EQ.END) GO TO 1
      IF(KD(1).EQ.NODES) GO TO 2
      IF(KD(1).EQ.BLANK) GO TO 3
      GO TO 4
C         NO NODES TO DO
    1 LL=0
      GO TO 5
C         NODES TO DO
    2 LL=2
      WRITE(KO,94) KD(1),KD(2)
    5 N=N-1
   99 IF(LER.EQ.0) GO TO 101
      KQ(8)=2
  101 RETURN
C         ARC TO FILE
    3 IF(KE(1).EQ.BLANK) GO TO 6
      IF(KE(1).EQ.ND(1)) NCOUNT=2
      IF(NCOUNT.EQ.2) GO TO 800
  950 CONTINUE
      KC(N)=KA(1,1)
      KU(N)=KA(2,1)
      KL(N)=KA(3,1)
      KX(N)=KA(4,1)
      IF(KE(1).EQ.KF(1).AND.KE(2).EQ.KF(2)) GO TO 9
      IF(NODENO(KF(1),KF(2)).EQ.M+1) GO TO 11
      WRITE(KO,91) KF(1),KF(2),IJ(2*N-1),IJ(2*N)
      GO TO 12
```


Information Processing Center

```
   11 KE(1)=KE(1)
      KE(2)=KE(2)
      IF(M.GT.KQ(5)) GO TO 23
      M=M+1
      NN(2*M-1)=KE(1)
      NN(2*M)=KE(2)
      NL(M)=N
    9 IF(N.GT.KQ(4)) GO TO 20
      N=N+1
      GO TO 6
    4 WRITE(KO,92) N
   12 WRITE(KO,93) KQ(1),KQ(2),KE(1),KE(2),IJ(2*N-1),IJ(2*N),(KA(I,1),I=OKF01980
     1 1,4)
      LER=LER+1                                                          OKF01990
      GO TO 6                                                            OKF02000
   20 WRITE(KO,89)                                                       OKF02010
   25 LER=10000                                                          OKF02020
      GO TO 99                                                           OKF02030
   23 WRITE(KO,88)                                                       OKF02040
      GO TO 25                                                           OKF02050
   88 FORMAT(27H0TOO MANY NODES IN THIS RUN)                             OKF02060
   89 FORMAT(26H0TOO MANY ARCS IN THIS RUN)                              OKF02070
   90 FORMAT(3(A4,A2),2X,4I10)                                           OKF02080
   91 FORMAT(36H0SOURCE NODES ARE NOT ADJACENT, ARC 4A4)                 OKF02090
   92 FORMAT(36H0CARD PUNCHING ERROR IN ARC CARD NO.,I6)                 OKF02100
   93 FORMAT(1H03(A4,A2),2X,4I10)                                        OKF02110
   94 FORMAT(1H0A4,A2)                                                   OKF02120
  800 IF(MPORT.GT.9) GO TO 849                                           OKF02130
      MULAR1=MULAR1+1                                                    OKF02140
      NCOST2(MPORT,MROOT,MULAR1)=KA(1,1)                                 OKF02150
      NCAPAR(MPORT,MROOT,MULAR1)=KA(2,1)                                 OKF02160
      IF(MULAR1.EQ.2) GO TO 801                                          OKF02170
      GO TO 950                                                          OKF02180
  801 MULAR1=0                                                           OKF02190
      IF(MROOT.EQ.MAX) GO TO 802                                         OKF02200
      MROOT=MROOT+1                                                      OKF02210
      GO TO 950                                                          OKF02220
  802 MROOT=1                                                            OKF02230
      MPORT=MPORT+1                                                      OKF02240
      GO TO 950                                                          OKF02250
  849 NCOUNT=0                                                           OKF02260
      GO TO 950                                                          OKF02270
      END                                                                OKF02280
      SUBROUTINE PREDAT(KS)                                              OKF02290
      DIMENSION KL(6100),KC(6100),KU(6100),KK(6100),JI(6100),            OKF02300
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)                 OKF02310
      DIMENSION LC(9),KA(18,2),KQ(9)                                     OKF02320
      COMMON /KL/KL/KC/KC/KU/KU/KK/KK/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      OKF00770
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER      OKF00780
      COMMON /MINF/MINF/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KO/KO/K/K      OKF00790
       INTEGER PAUSE,SAVE,READY,CARDS,TAPE,SKIP,TRANSP,ARCS,END          OKF00800
      DATA PAUSE,SAVE,READY/4HPAUS,4HSAVE,4HREAD/                        OKF00810
      DATA CARDS,TAPE,SKIP,TRANSP/4HCARD,4HTAPE,4HSKIP,4HTRAN/           OKF00820
      DATA ARCS/4HARCS/                                                  OKF00830
      DATA END/4HEND /                                                   OKF00840
      DO 200 I=1,4                                                       OKF00850
  200 LC(I)=0                                                            OKF00860
      KOR=KQ(3)                                                          OKF00870
      KQ(7)=0                                                            OKF00880
      KS=0                                                               OKF00890
```

```
   21 READ(KI,90) (KA(I,1),I=1,18)
      WRITE(KO,91) (KA(I,1),I=1,18)
      IF(KA(1,1).EQ.PAUSE) GO TO 180
      IF(KA(1,1).EQ.SAVE)  GO TO 50
      IF(KA(1,1).EQ.READY) GO TO 100
      GO TO 21
C          END JOB
  180 IF(KQ(5).EQ.0) GO TO 182
      K2=KQ(2)
      WRITE(KO,98)
      END FILE K2
      GO TO 183
  182 WRITE(KO,99)
  183 STOP
C          SAVE
   50 KS=1
    8 RETURN
C          READY
  100 IE=KQ(4)
      DO 101 I=1,IE
      KL(I)=0
      KQ(I)=0
      KU(I)=0
      KX(I)=0
      IJ(2*I-1)=0
      IJ(2*I)=0
  101 JT(I)=0
      IE=KQ(5)
      DO 102 I=1,IE
      NL(I)=0
      NP(I)=0
      IL(I)=0
      NN(2*I-1)=0
  102 NN(2*I)=0
      IL(IE+1)=0
      IE=MAXO(IE+1,KQ(4)-2*IE-1)
      DO 103 I=1,IE
  103 JL(I)=0
      M=0
      N=0
      LEP=0
      KQ(8)=1
    3 READ(KI,90) (KA(I,1),I=1,18)
      WRITE(KO,91) (KA(I,1),I=1,18)
      IF(KA(1,1).EQ.CARDS) GO TO 1
      IF(KA(1,1).EQ.TAPE) GO TO 6
      IF(KA(1,1).EQ.SKIP) GO TO 6
      IF(KA(1,1).EQ.TRANSP) GO TO 14
      GO TO 3
   14 KQ(8)=0
      GO TO 3
    6 IF(KQ(9).NE.0) GO TO 7
      KQ(9)=1
      REWIND KOR
    7 IF(KA(1,1).EQ.TAPE) GO TO 4
      GO TO 13
   1  KOR=KI
    4 READ(KOR,90) (KA(I,1),I=1,18)
      WRITE(KO,91) (KA(I,1),I=1,18)
```


Information Processing Center

```
C       TITLE
        DO 10 I=1,18
   10 KA(I,2)=KA(I,1)
        GO TO 4
   13 READ(KOR,92) KA(I,1)
        IF(KA(I,1).EQ.END) GO TO 3
        GO TO 13
   90 FORMAT(18A4)
   91 FORMAT(1H0,18A4)
   92 FORMAT(A4)
   98 FORMAT(31H0RESERVED TAPE HAS BEEN WRITTEN///1H0)
   99 FORMAT(34H0NO RESERVED TAPE HAS BEEN WRITTEN)
        END
        SUBROUTINE MAKEJL
        DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
       -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
        DIMENSION LC(9),KA(18,2),KQ(9)
        COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      OKF02220
        COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER      OKF02230
        COMMON /MINE/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K      OKF02240
C       NUMBERS TO IJ LIST
        I=1
        DO 1 L=1,N
    3 K=NODENO(IJ(2*L-1),IJ(2*L))
        IF(K.LE.M) GO TO 6
        IF(K.GE.KQ(5)) GO TO 9
        M=M+1
        NN(2*M-1)=IJ(2*L-1)
        NN(2*M)=IJ(2*L)
        IJ(2*L-1)=K
        NL(M)=N+1
        LER=LER+1
   19 IF(NL(I+1).GT.L) GO TO 18
        I=I+1
        IF(I.LT.M) GO TO 19
   18 WRITE(KO,90) NN(2*I-1),NN(2*I),NN(2*M-1),NN(2*M)
        GO TO 1
    6 IJ(2*L-1)=K
    1 CONTINUE
        DO 30 L=1,N
   30 IJ(L)=IJ(2*L-1)
C       FIX IL LIST
        DO 8 I=1,M
        CALL PLACE(NL(I),IL(I))
    8 NL(I)=0
        CALL PLACE(N+1,IL(M+1))
C       COUNT J-S
        DO 10 J=1,N
        I=LADDR(IJ(J))
   10 NL(I)=NL(I)+1
C       FORM JL LIST
        KK=1
        CALL PLACE(KK,JL(1))
        DO 20 I=1,M
        IF(NL(I).NE.0) GO TO 23
        WRITE(KO,91) NN(2*I-1),NN(2*I)
        LER=1
   23 KK=KK+NL(I)
   20 CALL PLACE(KK,JL(I+1))
  100 RETURN
```

```
    9 LER=100000
      WRITE(KO,92)
      GO TO 100
   90 FORMAT(5H0ARC ,4A4,18H IS A DEAD END ARC)
   91 FORMAT(21H0ND ARC ENDS AT NODE ,2A4)
   92 FORMAT(27H0TOO MANY NODES IN THIS RUN)
      END
      SUBROUTINE NODASY
      DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)
      DIMENSION KE(2),KD(2)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL     OKF02800
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER     OKF02810
      COMMON /MINF/MINF/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K
       INTEGER END,BLANK
      DATA END,BLANK/4HEND ,4H    /
      I=0
    3 I=I+1
      READ(KOR,90) KD(1),KD(2),KE(1),KE(2),KA(I,1)
      IF(KD(1).EQ.END) GO TO 99
      IF(KD(1).NE.BLANK) GO TO 2
      IF(KE(1).EQ.BLANK) GO TO 3
      K=NODENO(KE(1),KE(2))
      IF(K.GT.M) GO TO 6
      NP(K)=KA(I,1)
      GO TO 3
    6 WRITE(KO,91) I,KE(1),KE(2)
   10 LER=LER+1
      GO TO 3
    2 WRITE (KO,92) I,KD(1),KD(2),KE(1),KE(2),KA(I,1)
      GO TO 10
   99 RETURN
   90 FORMAT(2(A4,A2),8X,I10)
   91 FORMAT(5H0CARD I6,6H NODE A4,A2,12H NOT IN ARCS)
   92 FORMAT(37H0CARD PUNCHING ERROR IN NODE CARD NO.I6/1H 2(A4,A2),8X,IOKF03030
     -110)
      END
      SUBROUTINE TRANSL
      DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL     OKF04060
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER     OKF04070
      COMMON /MINF/MINF/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K
C     CLEAR NL STORAGE
      DO 11 I=1,M
   11 NL(I)=0
C     CALCULATE CIRCULATION AND C-BAR
      I=0
      LUP=1
      DO 2 L=1,N
      IF(L.LT.LUP) GO TO 13
      I=I+1
      LUP=LDECR(IL(I+1))
   13 LU=LADDR(IJ(L))
      NL(I)=NL(I)-KX(L)
      NL(LU)=NL(LU)+KX(L)
    2 KC(L)=KC(L)+NP(I)-NP(LU)
C     CIRCULATION MESSAGE FOR NON-ZERO CIRCULATION AND MOVE JL LIST       OKF  20
```

```
      DO 5 I=1,M
      IF(NL(I).EQ.0) GO TO 5
      WRITE(KO,90) NN(2*I-1),NN(2*I),NL(I)
    5 NL(I)=LDECR(JL(I))
C     COMPUTE EXCESS OF X AND UPPER BOUND OVER LOWER BOUND
      DO 1 J=1,N
      KU(J)=KU(J)-KL(J)
      IF(KU(J).GE.0) GO TO 1
      WRITE(KO,51) J
      LER=LER+1
    1 KX(J)=KX(J)-KL(J)
C     COMPUTE JI LISTS
      I=0
      LUP=1
      DO 22 L=1,N
      IF (L.LT.LUP) GO TO 25
      I=I+1
      LUP=LDECR(IL(I+1))
   25 K=LADDR(IJ(L))
      J=NL(K)
      JI(J)=I
      CALL PLACE(L,JI(J))
   22 NL(K)=NL(K)+1
C     CLEAR NL STORAGE
      DO 60 I=1,M
   60 NL(I)=0
      RETURN
   51 FORMAT(4H0A9C,I6,42H HAS LOWER BOUND GREATER THAN UPPER BOUND.)   OK6  10
   90 FORMAT(6H0NODE 2A4,28H NON-CONSERVATIVE, NET FLOW=I12)
      END
      SUBROUTINE KILTER(I)
      DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER    0
      COMMON /MINF/MINF/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K
      IF(LDECR(IJ(I)).EQ.0) GO TO 70
      CALL PLACE(0,IJ(I))
      DO 69 J=1,M
   69 NL(J)=0
   70 LER=0
    5 IF(KC(K)) 10,20,30
   10 IF(KX(K)-KU(K)) 13,40,14
   13 MINF=-KX(K)+KU(K)
      GO TO 50
   14 MINF=KX(K)-KU(K)
      GO TO 60
   20 IF(KX(K).LT.0) GO TO 13                                    OKF04750
      IF(KX(K)-KU(K)) 40,40,35                                   OKF04760
   30 IF(KX(K)) 32,40,35                                         OKF04770
   32 MINF=-KX(K)                                                OKF04780
      GO TO 50                                                   OKF04790
   35 MINF=KX(K)                                                 OKF04800
      GO TO 60                                                   OKF04810
   50 KOR=LADDR(IJ(K))                                           OKF04820
      KAT=0                                                      OKF04830
      KTER=I                                                     OKF04840
      GO TO 65                                                   OKF04850
   60 KOR=I                                                      OKF04860
```

```
      KAT=10                                                            OKF04870
      KTER=LADDR(IJ(K))                                                 OKF04880
   65 CALL LABELN(KRR)                                                  OKF04890
      IF(KRR.EQ.0) GO TO 68                                             OKF04900
      CALL BREAKT                                                       OKF04910
      GO TO 5                                                           OKF04920
   68 CALL UPNOPR                                                       OKF04930
      IF(LER.EQ.0) GO TO 5                                              OKF04940
   40 RETURN                                                            OKF04950
      END                                                               OKF04960
```

```
ESD                          %LABELN          BREAKT     I              0001
ESD             UPNOPR
ESD
ESD             NL           NN               NP                        0004
ESD             IJ           IL               JL                        0005
ESD             JI           M                N                         0006
ESD             LER          KAT              KOR                       0007
ESD             KTER         MINE             LC                        0008
ESD             KA           IFIN             KI                        0009
ESD             KQ           KQ               K                         0010
TXT             G00  LABELN &%&   A F &&  & & Q FB$ F Q-F6$-F Q&F%$&F Q-  0011
TXT             F2$-F Q0F>$0F Q     &   & F Q F+$ F Q F5Q   & F I   Q F   0012
TXT   0         Q F  7I0  G0 0Q0F Q00  2- Q0F  0  Q F Q&FF$&F HK&  &I&    0013
TXT   Y         H5& I-
TXT   -          7R6- G&A Q0F &R- H-   Y- I-  H    D  ! F Q0FWQ00
TXT             I-F G0 !Q&Fe%&F HK&  &I&  H5& I-
TXT   &         AQHQ  I&  R9& G&A-09- R90 G A%G0AO 7R9- G AQ00F  7&8- H&    0017
TXT   H           Q- H    D  ! F Q0FWQ00
TXT             F I   G0 DQ F Q0F &0  H      Q0F>! 0 & 0 & F Q&F Q%& K"&    0019
TXT   9          =G00  HREAKT &%&   A D &&  & & Q-D#$-D Q DH$ D Q&DQ%&D    0020
TXT             Q-DA%-D Q0D4$0D Q D Q   Q0D Q&D !&D &&0 Q D- 7&0  Q Q%Q    0021
TXT   Y          I   Q Q   Q DQ0   I   HM- HU- I-  Q0D H00
TXT   -          7R6& G&  06- $60
TXT   Q          7R6& G  $060 $A-
TXT
TXT
TXT
TXT
TXT
TXT             R+  0 R!Q   I
TXT   -         - G0 FQ1- R10 G   G0 FR6- G  FR1- G  FQ1&  7
TXT   Q         Q0B:Q00
TXT   &         R+  0 R0Q   I   HA&   I   HJ& I&   7R1- G0A%Q10  R&10 R0    0033
TXT             C G A*
TXT               I-   7R6- G0A<Q5&  R&5& !&B+G0A%G A G0AYQ0B!Q C & 0 Q&    0035
TXT   B         R%Q%& K"&  =Q0B!Q C & 0 G0AYG00  NODENO &%&   A  e&&  &     0036
TXT             & 0- w$- 6Q0  Q00 Q   Q   Q  :  Q  =Q   I
TXT   Y         RA- G  S!  !G   H
TXT             0%& K"&  =                                                0039
TXT   9                                                                   0040
TXT                                                                       0041
TXT   -          5/    M   .                                            0042
PLD                    A
PLD U       Y       % M  0044 Q       *       -
PLD                                                      D       H        0045
PLD                    <       &
END                                  F15OCT70 21.44  3/20/72              0047
ESD                             -PLACE           LADDR                    0001
ESD             LDECR     W                                               0002
TXT             G00  PLACE&%&   A  H&%  &   Q/  Q1  K 0 - Q%& K"&  =G00   0003
```

```
TXT   0      LADDR&& A M&S K  Q/ HW &_ 0&& K"& =600 LOECR      0004
TXT   0      &&& A &$ &  Q/ HW &_ 0&& K"& =                    0005
END                        F150C770 21.45  3/20/72            0006
//G.FT02F001 DD DSNAME=U.M10009.10002.DATA.DISP=(OLD.KEEP)
READY
TAPE
OUTPUT POINTER
COMPUTE
PAUSE
/*
```


Information Processing Center

BLANK

APPENDIX E

Seaport Simulation

(Tanker Terminal)

Seaport Simulation

The seaport simulation is written in PL/1, and is currently implemented on the MIT timesharing computer MULTICS. However, the program's memory requirements are such that a much smaller machine would be capable of executing the program. A listing and flowchart appear in Appendix VI. Numbers appearing in the following text, preceded by a #, refer to the numbered elements of the flowchart.

Preceding the simulation is a rather lengthy section of program labeled 'INPUT'. This set of instructions allows the program operator considerable flexibility in changing the parameters of the simulation. These parameters are as follows:

berth_capacity	=	total number of berths in the port. All berths are identical and capable of servicing one ship at a time.
tank_capacity	=	capacity of tank form, as expressed in shiploads of oil.
tank_outflow	=	rate of flow of oil from tank farm to inland users, as expressed in shiploads/day.
pumping_rate	=	rate at which a berthed ship can discharge its oil into the tank farm, as expressed in shiploads/day.
av_intarrival	=	average time between ship arrivals, as expressed in days.
length_of_run	=	number of days the simulation will run.
empty_tk_fine	=	fine levied against the port each day it can't supply the inland oil demand, as expressed in thousands of dollars.

berth_charge (n) = charge for keeping a ship in berth on its n^{th} day in berth, as expressed in thousands of dollars.

anchoring_chg (n) = charge for keeping a ship in anchorage on its n^{th} day in anchorage, as expressed in thousands of dollars.

Once the operator is satisfied with the input parameters, he begins the simulation. Initialization of all variables is done first (#1). The port is cleared of all ships, all charges are zeroed, and the tank farm is set at an initial state of being half full.

A one-dimensional array, larrival (i) is set up. The elements of larrival are the number of new tankers that arrive at the port on day i. For instance, if larrival (3) = 2, then two fully loaded tankers will arrive at the port on the third day of the simulation. The arrival times are such that interarrival times are randomly distributed according to an exponential distribution with a mean equal to av_intarrival (#2).

All port activities are handled on a day-to-day basis. The remainder of the program simulates one day's activities, and will be repeated length_of_run times. At the end of each iteration, the variable lday will be incremented by one (#3). Lday can be considered as the date of the iteration. It will begin as being equal to one.

The first question asked is whether or not a new ship has arrived at the port, i.e., is larrival (lday) greater than zero (#4). If not, the program immediately moves

to the processing of the ships already in port (#19). If a ship has arrived, the problem arises of determining whether or not it should be serviced. It is possible that the port may be so congested that more profit can be realized by refusing to service the ship and sending it on its way. To make this decision, the program enters the 'optimization algorithm'.

The algorithm is intended to simulate the decision-making process of the port manager. It considers the scheduling of the next seven ships that will arrive at the port. All possible combinations of servicing and rejection of these seven ships are formed. This results in 128 possible schedules ($2^7 = 128$). Each schedule is evaluated by running a simulation of the port with the schedule in force. The solution which minimizes charges against the port is chosen, and is used to determine the fate of the newly arrived ship. Note that this strategy does not necessarily lead to the optimum schedule, but the solution it obtains is near-optimum, i.e., a "good" solution. The details of this strategy follow.

The program fills an array, lable (128,7), with binary representations of all possible schedules (#5). A zero represents not servicing a ship, while a one implies servicing it. Thus, lable can be considered as a list of all seven-digit binary numbers from 0000000, representing rejection of all of the next seven ships, to 1111111, representing the servicing of all seven tankers.

A particular schedule is selected (#6) and a duplicate of larrival, la, is made. The selected schedule is used to delete those ships that won't be serviced from la. This modified la is scanned to find the arrival time of the eighth ship (#7&8).

A fake port is set up, identical in every way to the current port configuration and state (#9). This fake port is then led through a simulation of day-to-day activities, from the present day to the date of the arrival of the eight ship (#10-14). Except for the fact that the ship schedule is fixed, this simulation is identical to that which the "real" port would experience. The charges accrued by the fake port are calculated and stored for future reference (#15).

This process of selecting a schedule and implementing it on a fake port is repeated until all the schedules in lable have been evaluated. The charges accrued in the imulations are compared, and the schedule that results in the minimum cost is selected. This is used to determine the fate of the ship that has just arrived in port. It is scanned (#17), and if it tells us not to service any ship arriving in port "today", that ship is deleted from larrival (1day).

If, after this process, there still remain any ships arriving "today", they are moved into anchorage (#18). If any free berths are available, the ship(s) that has remained in anchorage the longest is berthed (#19). Note that this

is the set of instructions the program would have skipped to if larrival (lday) were equal to zero (no ships arriving today).

The program then simulates the day's cargo discharging by decreasing the amount of oil in each berthed ship by pumping_rate, and correspondingly increasing the level of the tank farm. The level of the tank farm is decreased by an amount equal to tank_outflow to simulate the transmission of oil to inland users. The oil level of the tank farm is not permitted to exceed tank capacity or to drop below zero. All the ships in the port have associated "lengths of time in anchorage" or "lengths of time in berth". These values are incremented by one, representing the spending of one more day in their positions. These values are used to calculate the charges made on the port, and these charges are added to the total charges already accrued (#20). A fine is imposed on the port if the tank farm was unable to supply the inland demand. If any berthed ships have drained their tanks during the day, they are removed from the port (#21). This concludes one day of port operations. The value of lday is incremented by one, and the program returns to flowchart box #3 to begin another day.

After length_of_run iterations, the following cumulative values are printed:

total ships served
total ship-days spent in anchorage
total ship-days spent in berth
total oil pumped
total anchorage charges
total berth charges
total empty tank charges
total cost

A sample run of the simulation has been prepared, and appears in Appendix II. It is heartily suggested that the reader study it in an effort to make the above discussion meaningful.

The program, as now written, can simulate a port with up to 20 berths, 20 ships in anchorage, and infinite tank farm capacity for runs up to 2000 days. Any level of demand is permissible. Execution time is approximately proportional to the value of av_intarrival and porportional to the length_of_run. MULTICS executes a five-berth, 50-day run in about thirty seconds of CPU time.

Discussion of the Seaport Simulation

The reader will probably notice that the simulation described above is a rather simple, brute-force model. Undoubtedly, it would be possible to devise a far more elegant model. However, for the purposes of this work, such sophistication is not needed, nor even desired.

As it is intended to incorporate this model with a larger dynamic programming routine, it is of great importance that the model be capable of rapid execution. Attempts to more accurately reflect real-world operations would certainly lead to greater execution times. As was previously mentioned, the "optimization algorithm" can find only "good" solutions to the problem of ship scheduling. It is possible to construct dynamic programming routines capable of determining the optimum solution, but again, these involve

an excessive amount of execution time.

Although the above considerations led to the development of a simple, straight-forward model, several unanticipated advantages resulted from the decision. In future work, it will be necessary to construct a function giving a "quality of service" parameter. The analytic structure of this function is still rather vaguely defined. It has not even been determined just which parameters the function will use as input. However, no matter what values the quality of service will depend on, the present simulation will probably be able to generate them. A more ingenious model, utilizing clever relationships and dimensionless parameters, might very well not have this flexibility.

Through the use of negative pumping rates and the modification of only two statements, the seaport simulation can model a port that exports, rather than imports, oil. A few simple modifications could result in a model for light cargo handling ports with the tank farm representing a set of transit sheds. In short, the present model can be easily modified to represent a wide variety of ports. These changes might not be feasible for a more sophisticated program.

The present model attempts to operate the seaport in such a way as to minimize the charges incurred. It is not at all clear that this is the criterion used in actual ship scheduling. In future work, existing ports should be studied to determine the decision function used. It is highly probable that the existing model would need only small changes to

conform more closely with reality.

The Feedback Model

The following is a discussion of planned future work.

In the visualization outlined in Figure 5, the seaport simulation is viewed as a "black box", into which the parameters of berth capacity, tank farm capacity, and demand (tank_outflow and av_intarrival) are fed. The simulation uses these parameters, and produces a "quality of service" parameter. Thus, the quality of service is a function of the three input parameters.

Future demand made on the port is a function of the quality of service rendered. Specifically, it is assumed that the quality of service is the input to a simple, first-order economic model. The rate of change of demand with time is directly proportional to the quality of service:

$$\frac{d\ (\text{demand})}{d\ (\text{time})} = C_1 \times (\text{quality of service}) - C_2$$

where C_1 = constant, C_2 = constant

Implementation of this model will be as follows. The seaport model will be run for a period of a year with a constant port configuration and demand. The resulting quality of service parameter will then be used to modify the demand. Another simulation will be executed with this new demand level , resulting in a new quality of service parameter. This will determine a new demand level, etc. The trajectory followed by the demand will be compared to those

followed in existing ports. Modifications to C_1, C_2, and the quality of service function will be made until the model realistically represents the actual situation.

It is felt that even this simple model will be capable of an accurate simulation of reality. It is hoped that a more sophisticated approach will not prove to be necessary. However, as long as the feedback model does not incorporate a "memory", it can consist of arbitrarily complex functions without seriously affecting future work.

The Dynamic Programming Algorithm

After the implementation of a suitable economic feedback model, the seaport system has two remaining input parameters, berth capacity and tank farm capacity. These parameters are within the control of some sort of port authority, which has the power to augment the capacities in any way desired. The port authority will attempt to schedule port expansion in such a way as to maximize the profit realized from the port's operation, where "profit" is a weighted function of monetary gain and benefit to the country served by the harbor. It is apparent that the port should maintain a proper mix of berth capacity and tank farm capacity so as to prevent the under-utilization of facilities. The optimum schedule for such expansion can be found by a straight-forward, three-dimensional dynamic programming algorithm staged forward over time. The proposed algorithm is outlined below.

The simulations described in previous sections of this report result in a model whose state is completely described by three state variables: demand, berth capacity, and tank farm capacity. For a given set of these variables, the quality of service rendered by the port and the profit realized through the port's operation can be found simply by plugging in the variables and simulating port activity for a year's time. It is assumed that the state variables can assume only a finite number of values. To keep the size of the search space within practicable limits, it is assumed that the total number of possible states is less than one thousand. For instance, each state variable could be allowed to assume ten different values (one to ten berths, tank capacity of from one to ten shiploads of oil, ten levels of demand).

The long-range planning in question has a time scale of perhaps twenty years. In order to simplify the problem, it is assumed that the port configuration and demand level will not change during any one year of operation. Changes will take place only between years. That is, the system's state will not vary continuously, but rather, instantaneous changes will occur at the end of each year. Continuous variation leads to an algorithm involving an infinite number of stages, while the above assumption leads to only twenty stages.

With the above restrictions in force, a dynamic programming routine can be executed. The first and longest act is to run the seaport simulation for all possible combinations

of the state variables. Each run produces two values, the quality of service parameter and the profit realized. The results of all runs are stored in a large matrix for future reference.

The second act is the staging of the algorithm through time, beginning at year zero with the port in an initial state. Reference to the matrix yields the profit and quality of service parameter for this state. The latter value, through the economic feedback model, determines the demand during year one. The one hundred possible combinations of berth and tank capacities are formed and through reference to the matrix, the profits from each combination are found. By adding the profits from year zero to each of the profits from year one and subtracting the construction costs resulting from capacity addition, the total profits are computed. Also, the quality of service parameter for each combination is found through reference to the matrix. These two values are stored in a large table, along with their associated descriptions of the year one states.

The next stage sets the routine for all following stages. From each of the states the system can be in during year one, the system can move to a large number of states for year two. However, these transitions are limited by the fact that the state at year one determines the demand for year two. That is, if one of the possible states for year one is described as (bc_1, tk_1, d_1), then the system can go from this state to the state $(bc_2, tk_2, F(bc_1, tk_1, d_1))$ for year two, where F

represents the economic feedback relation. Thus, the system can move from any possible state at year one to only a finite subset of all states at year two, namely, that subset in which the demand variable equals that determined by $F(bc_1,tc_1,d_1)$.

All allowable transitions from the states at year one are attempted. If two or more transitions result in the system being in the same state at year two, a simple search is executed to determine which transition produces the maximum profit. This transition is recorded, total profit calculated, quality of service found, and all three quantities stored in a table of all possible states which the system can be in at year two.

The process is repeated for the transitions to year three, etc. Recapitulating, at year n, the system can be in any of m states, where m is less than 1000. Each of the m states is described by a set of state variables, (bc_n,tk_n,d_n), and an associated total profit. In the transition to year n + 1, the system can go from each of the m states to l new states, where l is less than 1000. Each of these l states can be described as $(bc_{n+1},tc_{n+1},d_{n+1})$, where $d_{n+1} = F(bc_n,tc_n,d_n)$, and $bc_n \leq bc_{n+1} \leq 10$, $tc_n \leq tc_{n+1} \leq 10$ (assuming capacity will not be removed from the port). Each possible transition has associated with it a total profit. The transitions from all m states at year n to all their respective allowable states at year n+1 should be attempted. If two or more transitions result in the same state at year

n+1, the profits from the transitions are compared, and the transition showing the highest profit is recorded. After all possible transitions have been evaluated, the process is repeated for all possible transitions from year n+1 to year n+2.

At the end of the twentieth stage, the algorithm has generated twenty tables of possible states, one table for each year. Each table lists all possible states the system could be in during that year, the transitions for these states from the preceding year, and the associated profits. To find the optimum schedule, begin at the twentieth table. Find the state showing the highest total profit, and note the transition that brought the system to this state from year nineteen. This tells which state the optimum schedule specifies for year nineteen. Look at the transition listed for this state from the year eighteen. This tells which state the optimum schedule specifies for year eighteen. Continue backtracking through the tables to year zero. The state trajectory followed is the optimum schedule.

Proposals for Future Work

Continuation of this study should be composed of the following sequential steps:

1) Extensive testing of the seaport simulation. Investigation of a suitable tanker port to compare the simulation's behavior to actual practice. If necessary, modification of the simulation.

2) Determination of appropriate quality of service function. Development and testing of an economic feedback model. Development of a suitable profit function.

3) Rewrite program in a language with faster execution time than PL/1, such as FORTRAN. Attempt to streamline the program by reducing the number of redundant operations.

4) Production of the matrix needed by the dynamic programming routine.

5) Write and execute the dynamic programming algorithm.

6) Compilation of final report.

APPENDIX F

Sample Run of Seaport Simulation

This appendix includes a short sample run of the seaport simulation. During execution, the command "data dump" was activated, which results in a day-by-day description of port activities. During normal operation, this command is usually deactivated, as the information it results in is printed on a console typewriter, a rather slow device.

The first page of input lists all the input data. Note that the port is a rather small one, with only one berth. However, the demand on its services is rather high; 0.5 shiploads of oil will be drained from its tank farm every day. The port will need at least one ship to arrive every two days to maintain this high outflow. Note that av_intarrival = 1.0, so that on the average one ship will arrive at the port every day. Thus, we would expect the port to have plenty of oil, i.e. the tank farm should never run dry.

Note that pumping_rate = tank_outflow = 0.5 shiploads per day. The port must keep its one berth working continuously to maintain its initial tank farm level. The charges are as listed, and the length of the run is ten days.

The second page of output begins with a listing of the ship arrival times, as generated by a Poisson exponential random number generator. These times are used to produce the next listing, that of larrival. Larrival is printed horizontally, so we see that on the first day of the simulation, (lday = 1) three ships will arrive at the port. The

second day (lday = 2) one ship will arrive; the third day (lday = 3) two ships will arrive, etc.

The rest of the second page is a listing of the port's activities for the first day of operation (lday = 1). Since a ship did arrive at the port this day, the optimization algorithm was utilized. The algorithm determined that the best schedule was number 17, which can be represented as 1000100. The three ships that arrived on lday are represented by the first three digits of the binary number, 100. We see that one of the three ships is to be serviced, while the other two are to be sent away unserviced.

Beneath the word "Anchorage" is a blank line, indicating that the anchorage is empty. Under "Berths" is the line "1.0000000 .50000000." This indicates that there is one ship at berth, it has been there one day (1.0000000), and is half full of oil (at the end of lday = 1). This ship is the one the optimization algorithm just admitted to the port. The tank level, after one day's draining and ship discharging, is 1.0000000 shiploads of oil (it started out at lday = 0 with a tank level of 1.0000000, and equal amounts of oil have been pumped into and drained out of it).

The top of the third page lists the activities for the second day (lday = 2). A ship arrived at the port, and we see from the first digit of the optimization algorithm's output (0100010) that it should not be serviced. There are

no ships in anchorage or berth (the ship in berth on lday = 1 has completed its discharging during the second day and left). The tank level is still 1.0000000.

On lday = 3, another ship has arrived, and the optimization algorithm has decided to service it (1000100). The ship was immediately moved into berthage, and has discharged half its load. On lday = 4, another ship has arrived at the port, but it was decided not to service it. The ship present on the third day has finished unloading and has left. The next day, lday = 5, no ships arrive at the port, and none are in berth. The tank level correspondingly drops by one tank outflow. On lday = 6, a ship arrives and is moved into berthage. On lday = 7, a new ship is not serviced and the ship berthed on day six finishes unloading and departs. During the eighth day, three ships are sent away unserviced, and one is berthed. The port realizes that on lday = 10, no ships will arrive, so on lday = 9 it takes the arriving ship and temporarily stores it in anchorage while the ship berthed on the eighth day finishes discharging. Thus, on the tenth day, after the day-eight ship has left, the anchored ship can be berthed and unloaded to keep the tank level from dropping to zero.

The last few lines of output give the various costs and performance factors.

Numerical Example of Simplified Dynamic Seaport Algorithm

The following example illustrates the operation of the flow chart of Figure 4.5.Allocation to berth space has been neglected for simplicity. The recursion relation is then:

$$C_N = \underset{\{t_N\}}{\text{Min}}\{C_{N+1} + q_N + f_N - h_N + \lambda d_N\}$$

We shall assume:

$$q_N = \frac{dN}{tn}$$

$$\delta_N = t_N - t_{N-1}$$

$$h_N = d_N$$

$$d_N = 2d_{N+1} - t_N$$

$$\lambda = 0$$

$$kmax = 3$$

$$M = 3$$

$$d_3 = 5$$

$$t_0 = 2$$

stage 3

$$C_3 = \underset{\{t_3 \geq t_2\}}{\text{Min}}\{0 + \frac{5}{t_3} + t_3 - t_2 - 2.5 + t_3\}$$

suppose $t_2 = 1$

then $t_3 = 1 \Rightarrow C_3 = -4$

$t_3 = 2 \Rightarrow C_3 = -4.5$

$t_3 = 3 \Rightarrow C_3 = -3.33$

suppose $t_2 = 2$:

then $t_3 = 2 \Rightarrow C_3 = -5.5$

$t_3 = 3 \Rightarrow C_3 = -6.33$

suppose $t_3 = 3$:

then $t_3 = 3 \Rightarrow C_3 = -6.33$

The optimal policy table stored is thus:

t_2	C_3	$t_{3opt.}$	d_2
1	-4.5	2	8
2	-5.5	2	8
3	-6.33	3	7

stage 2

$$C_2 = \underset{\{t_2 \geq t_1\}}{\text{Min}} \{C_3(t_3(t_2)) + \frac{d_2(t_2)}{t_2} + t_2 - t_1 - 2d_2(t_2) + t_2\}$$

suppose $t_1 = 1$:

then $t_2 = 1 \Rightarrow C_2 = -11.5$

$t_2 = 2 \Rightarrow C_2 = -14.5$

$t_2 = 3 \Rightarrow C_2 = -5$

suppose $t_1 = 2$:

then $t_2 = 2 \Rightarrow C_2 = -15.5$

$t_2 = 3 \Rightarrow C_2 = -6$

suppose $t_1 = 3$:

then $t_2 = 3 \Rightarrow C_2 = -7$

The optimal policy table stored is:

t_1	C_2	$t_{2opt.}$	d_1
1	-14.5	2	15
2	-15.5	2	15
3	-7	3	11

stage 1:

$$C_1 = \underset{\{t_1 \geq t_2\}}{\text{Min}} \{C_2(t_2(t_1)) + \frac{d_1(t_1)}{t_1} + t_1 - t_0 - 2d_1(t_1) - t_1\}$$

$t_0 = 2$, so:

$t_1 = 2 \Rightarrow C_1 = -36.5$

$t_1 = 3 \Rightarrow C_1 = -21.33$

The optimal allocation is:

t_0	C_1	$t_{1opt.}$	d_0
2	-36.5	2	28

If the initial demand happened to be $d_0 = 28$, we would now have the answer. If not, we would attempt to arrive at $d_0 = d_0$(required) by inserting a lagrange multiplier.

$$C_N = \text{Min}\{C_{N+1} + q_N + f_N - h_N + \lambda d_N\}$$

The optimal value of t_N at each step would correspond to a small demand d_N if λ were large. The correct value of λ would produce the correct value of d_0.

If $d_0 = 28$ were the correct value, then the optimal schedule is clearly given by:

$t_0 = 2$

$t_1 = 2$

$t_2 = 2$

$t_3 = 2$

This says that because $t_{max} = 3$, we cannot add enough capacity to maintain the initial demand, and adding one unit (t=3) will cost more to install than it will raise profits.

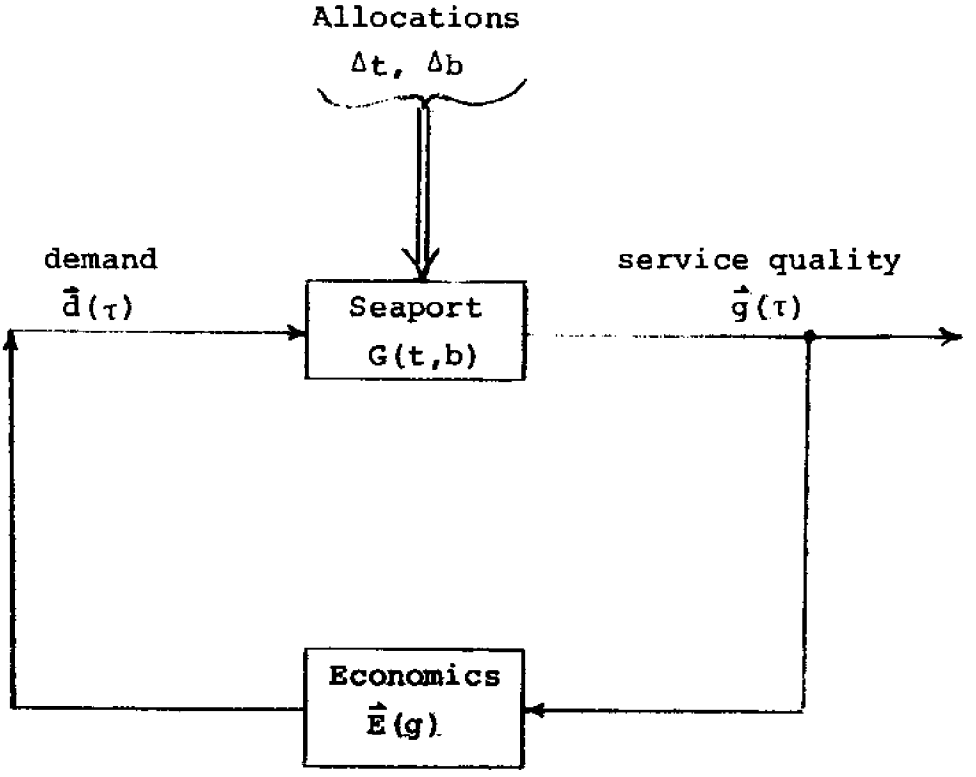


Figure F.1

System Configuration

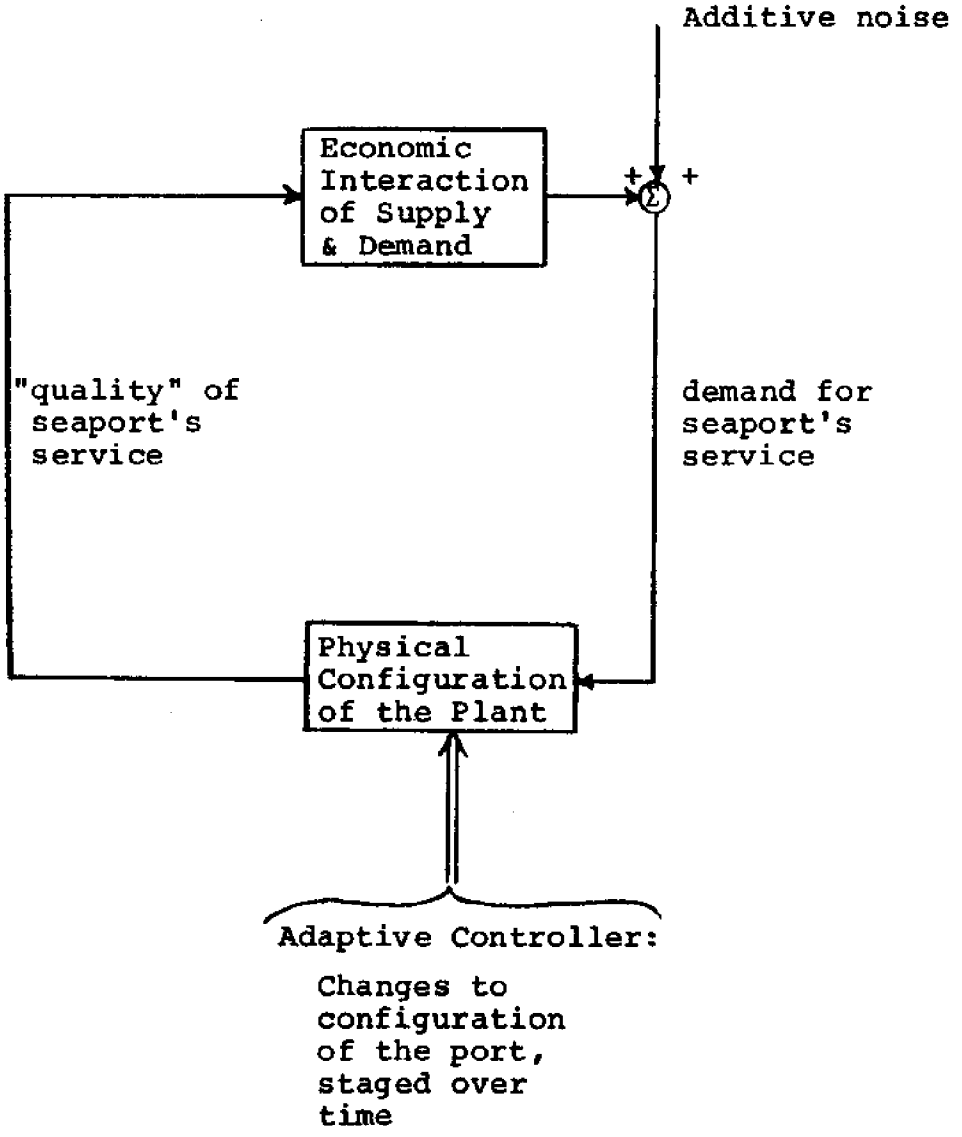


Figure F.2

Signal Flow Block Diagram

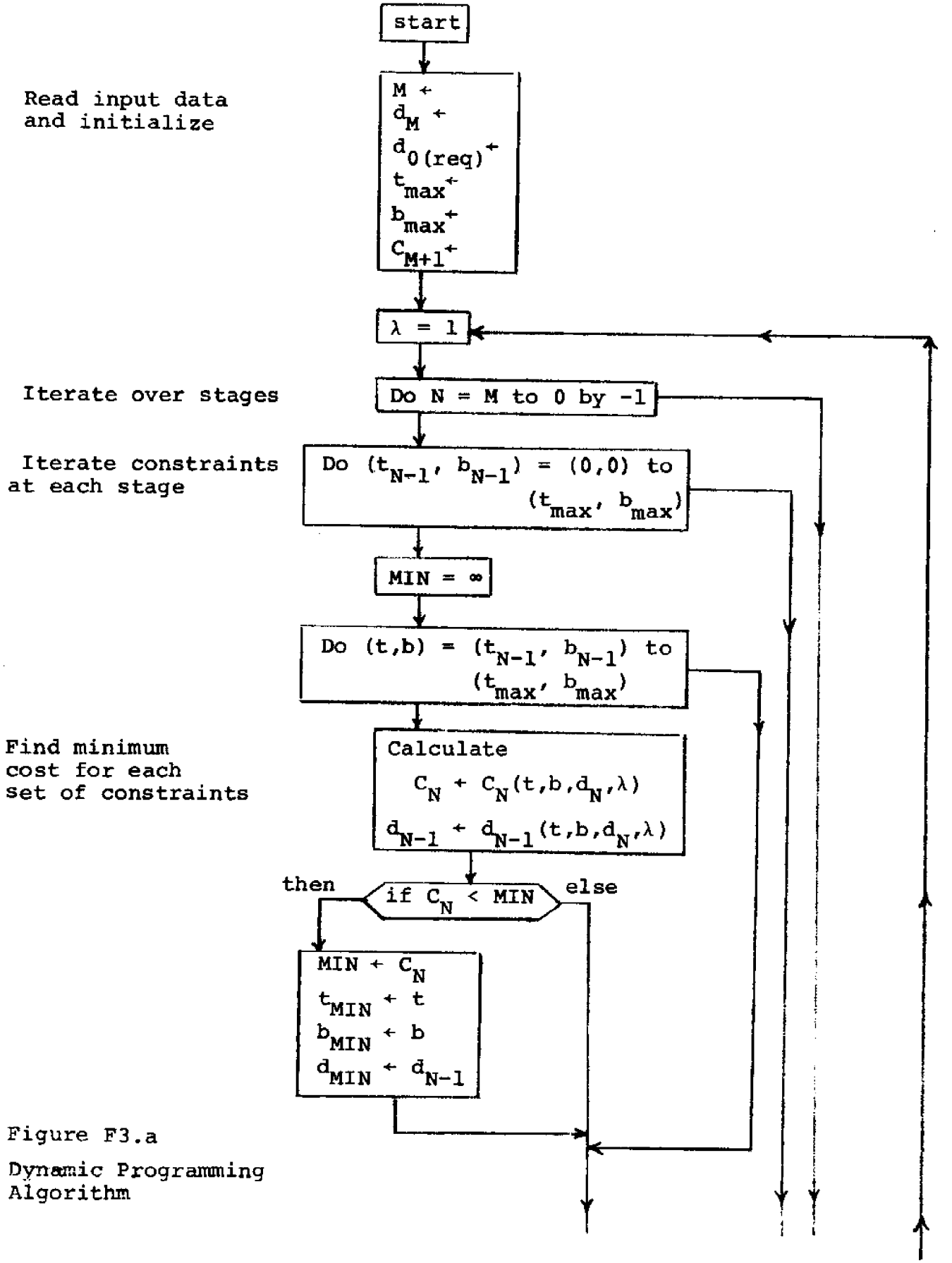


Figure F3.a

Dynamic Programming Algorithm

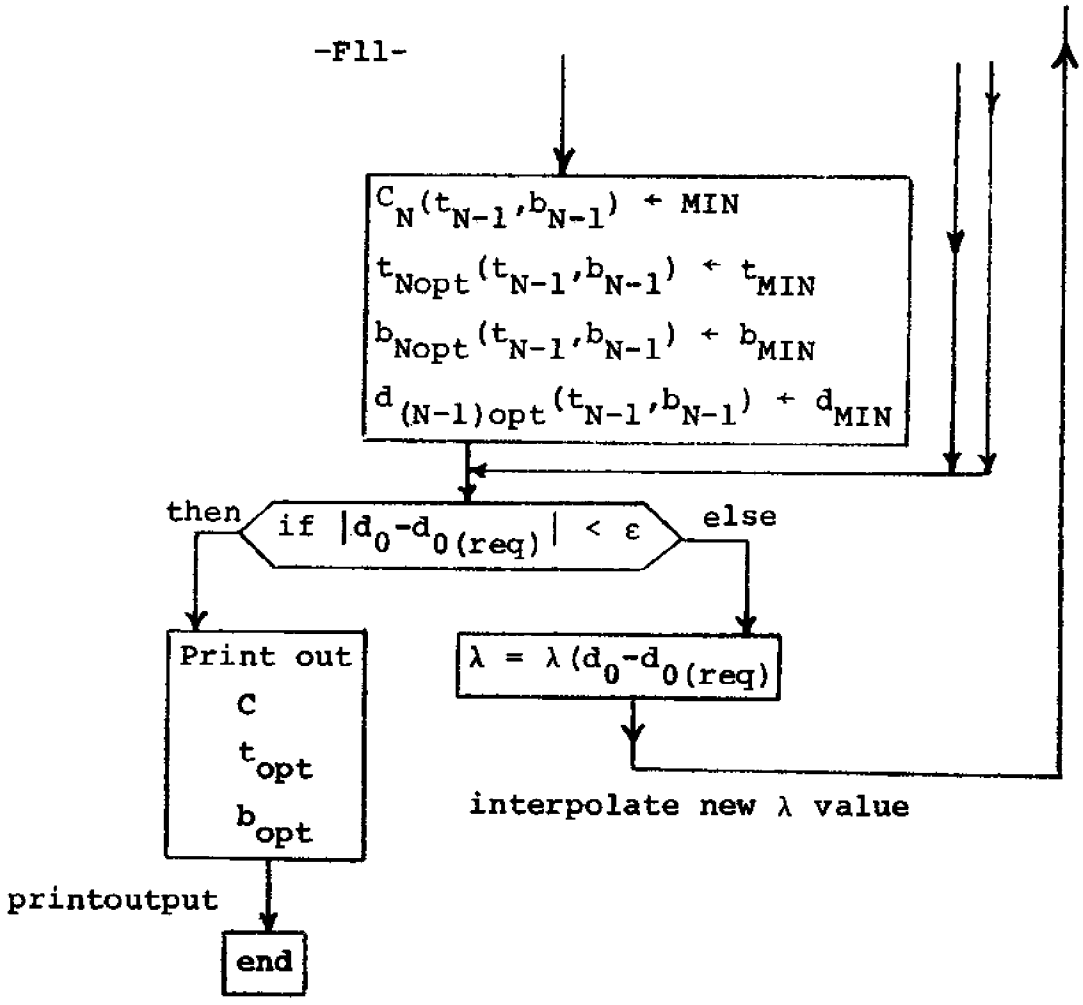


Figure F3.a (continued)

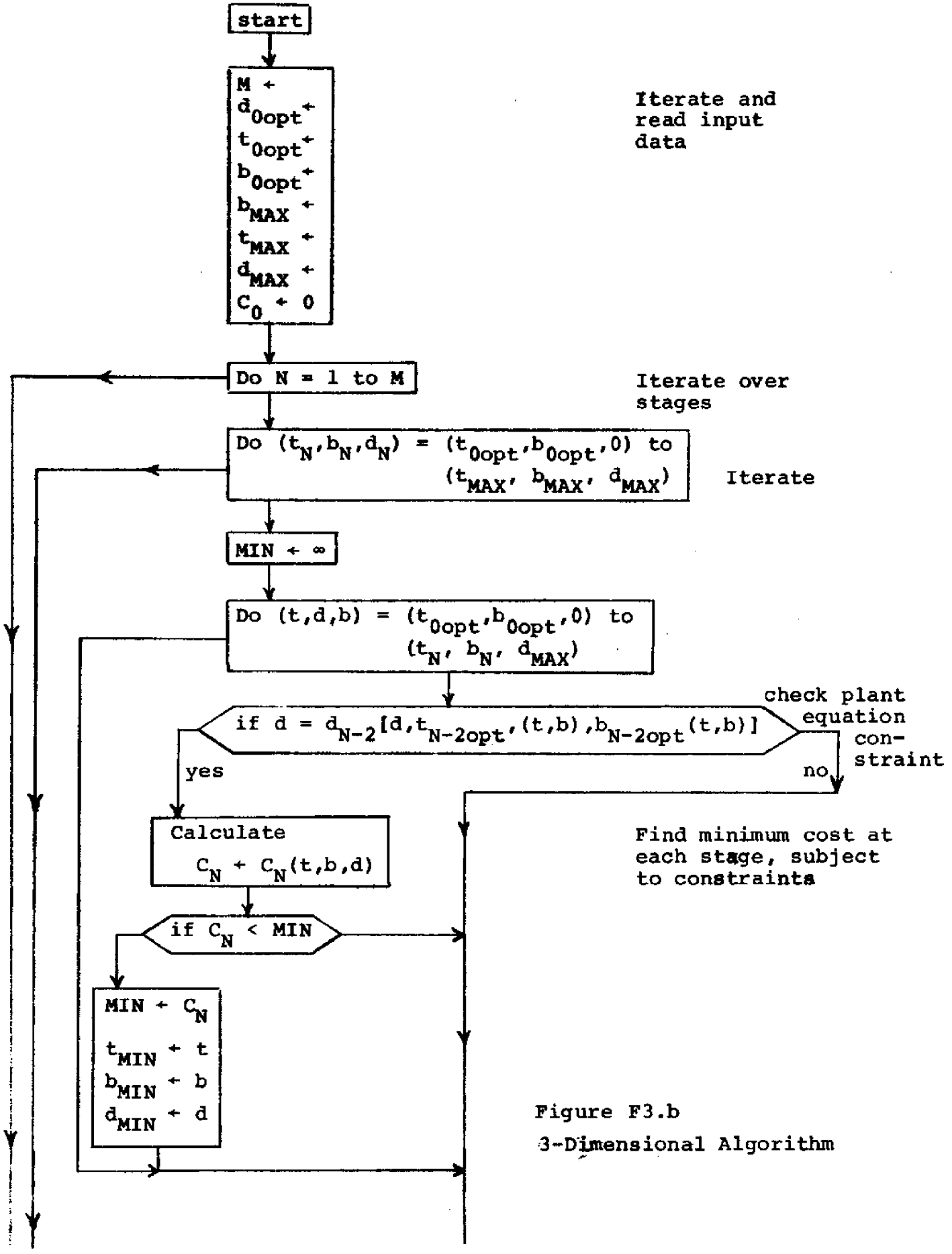
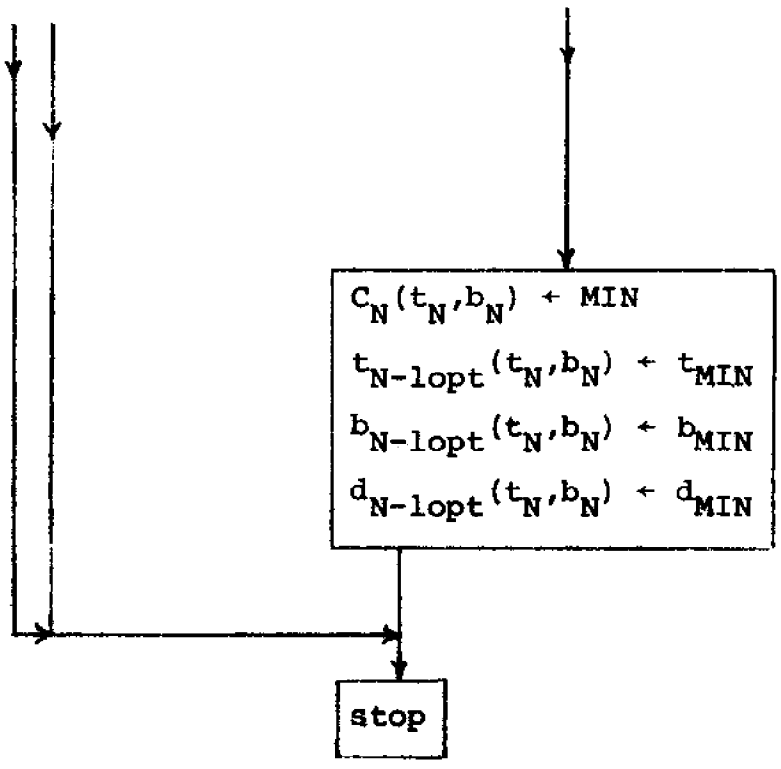


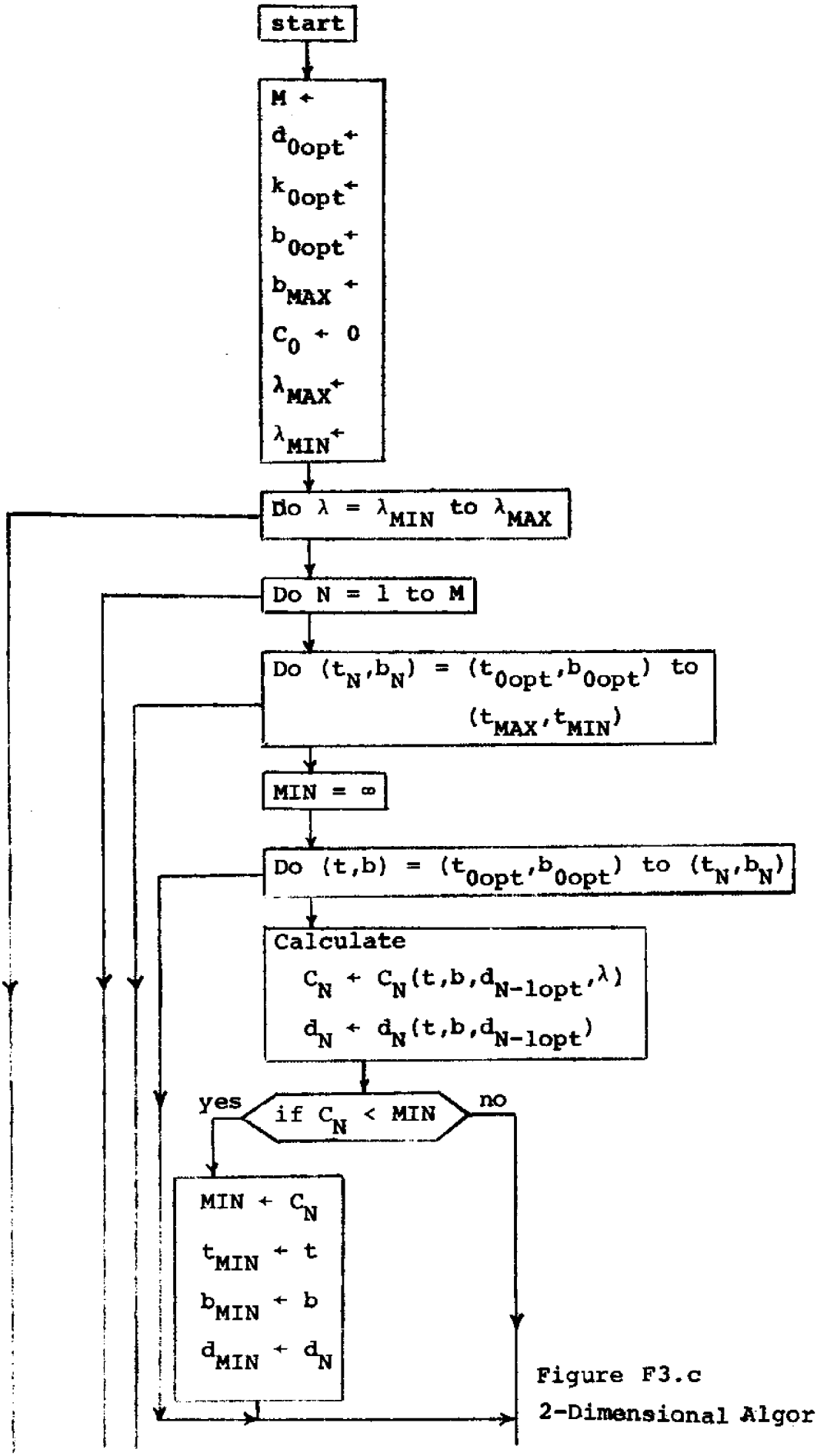
Figure F3.b

3-Dimensional Algorithm



store results in optimal policy table

Figure F3.b (continued)



Initialize and read input data

Iterate over stages

Iterate constraints at each stage

Find minimum cost for each set of constraints

Figure F3.c
2-Dimensional Algorithm

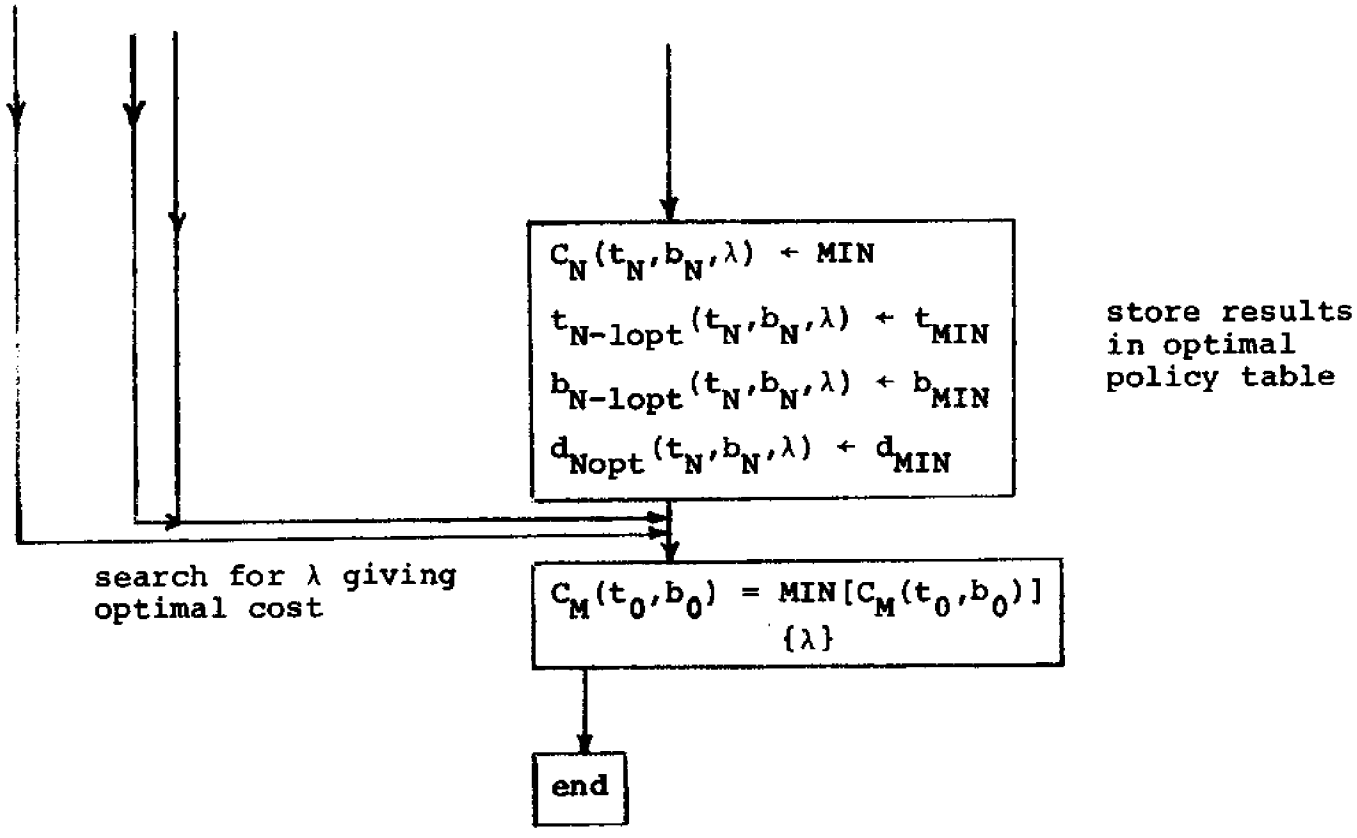
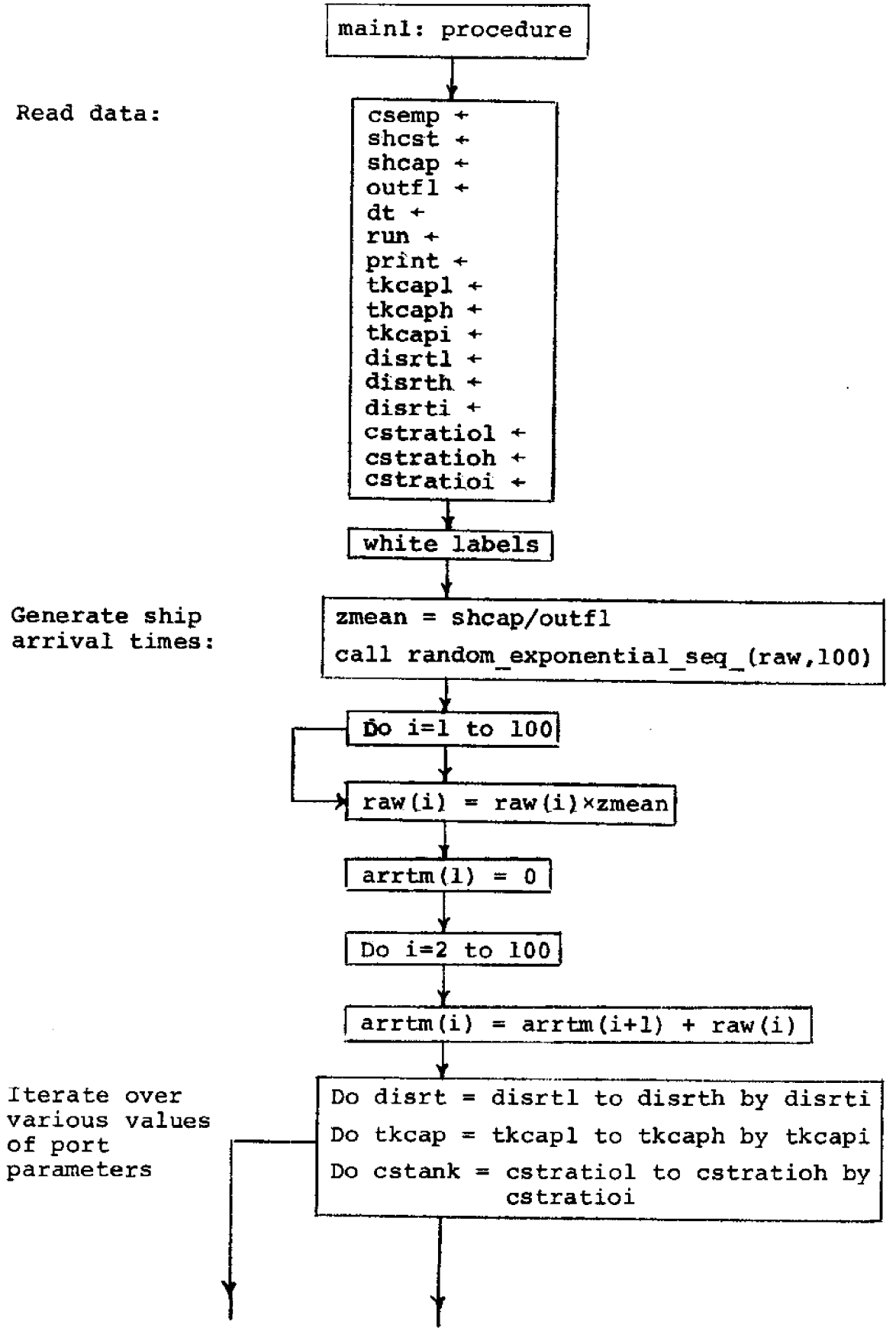


Figure F3.c (continued)

Figure F4

MAIN1: FLOWCHART



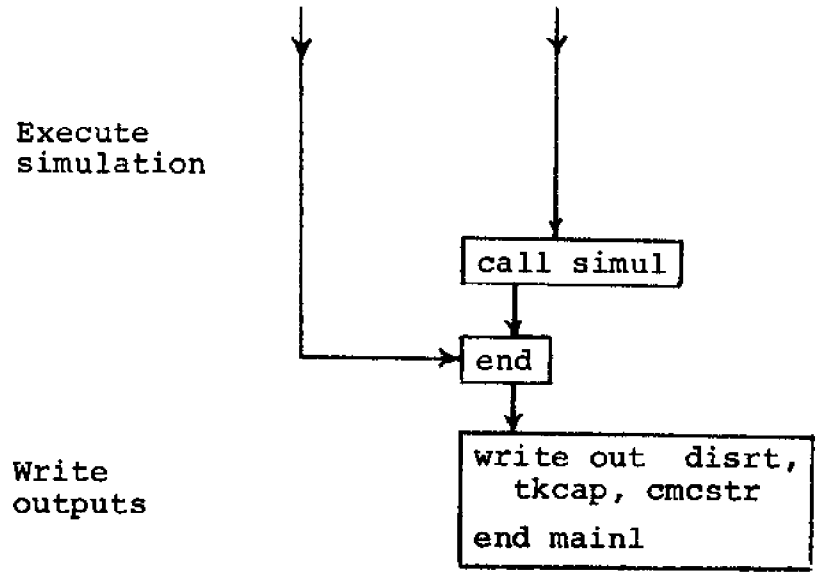


Figure F4 (continued)

Figure F5

SIMUL: FLOWCHART

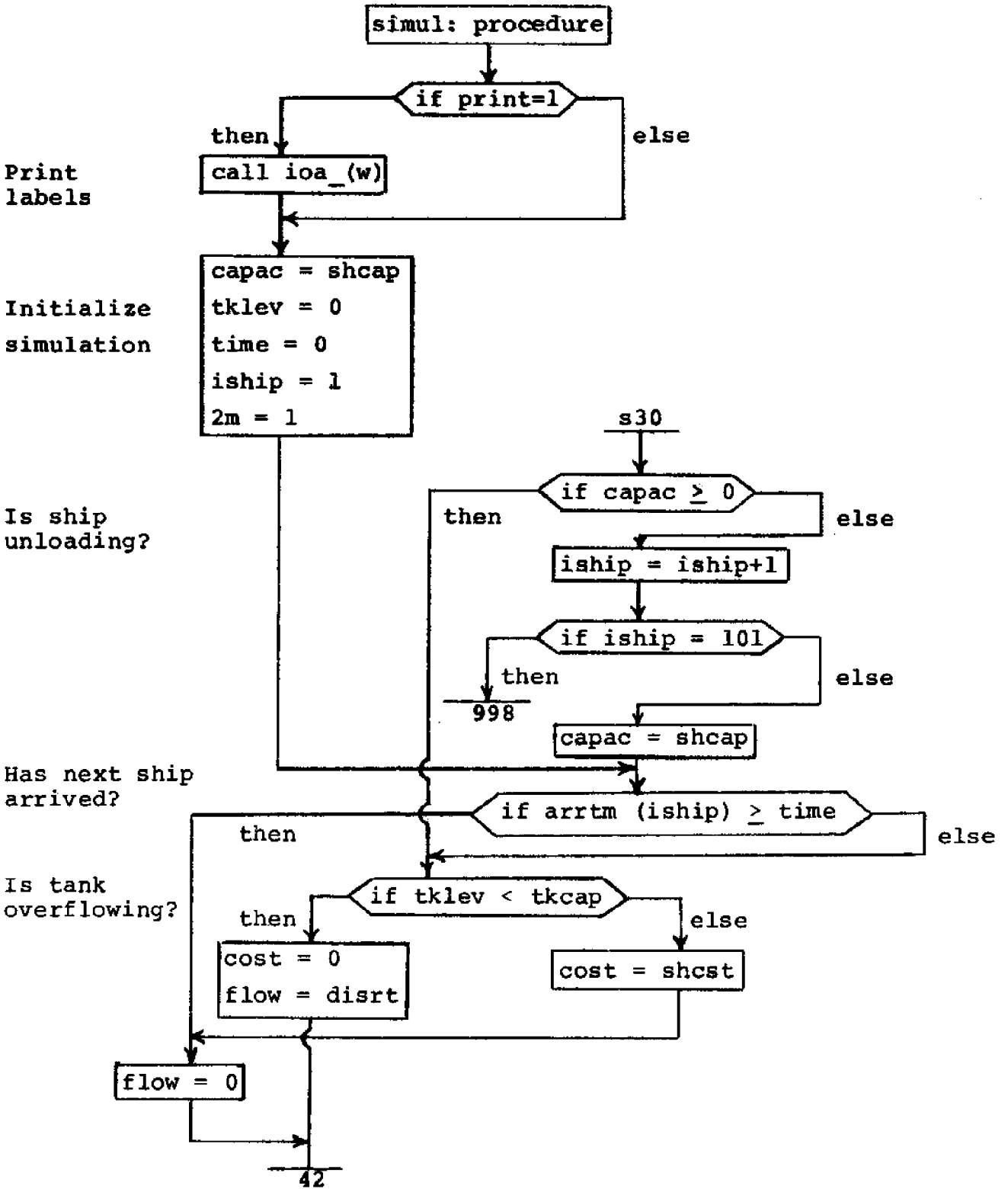


Figure F5 (continued)

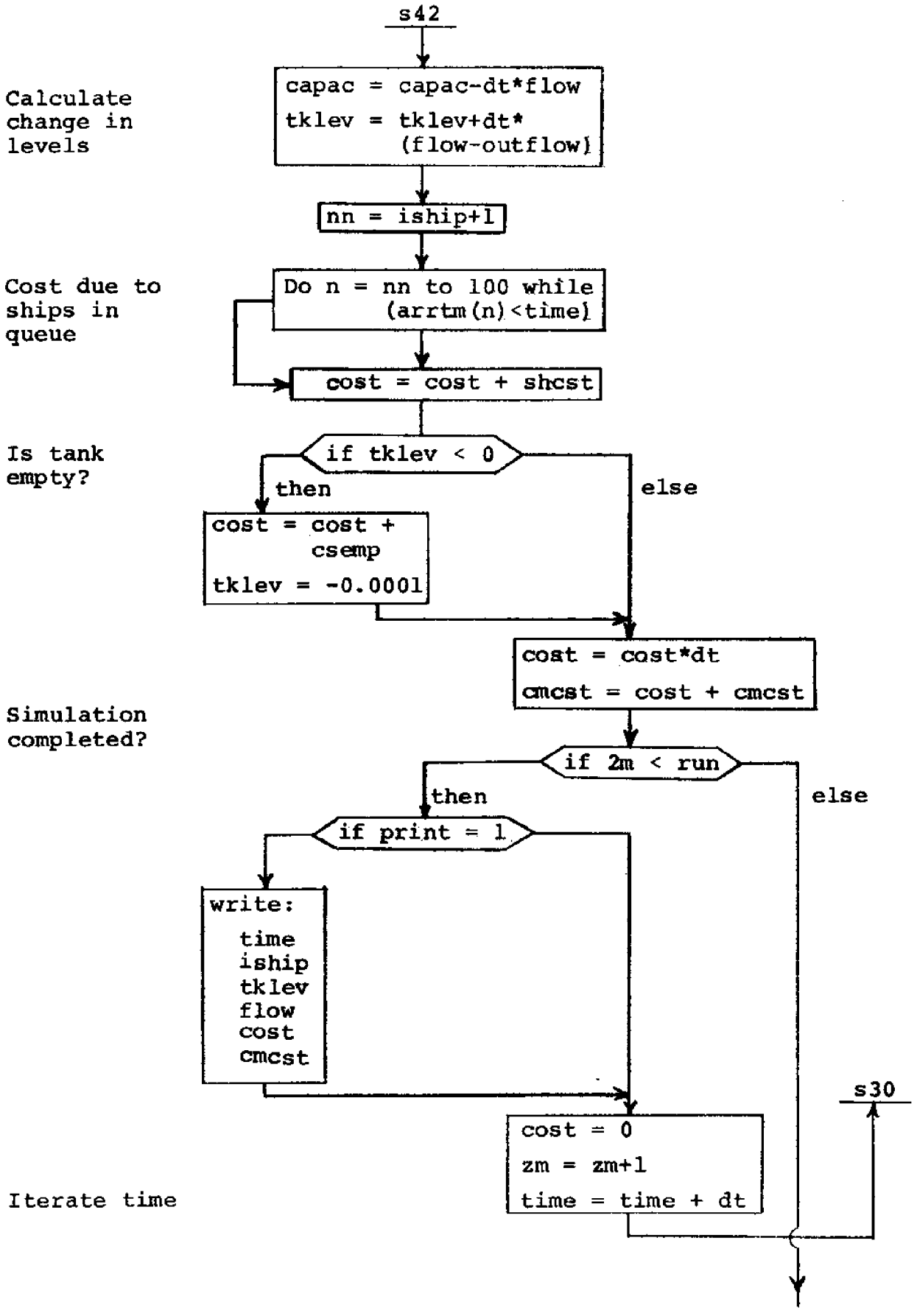


Figure F5 (continued)

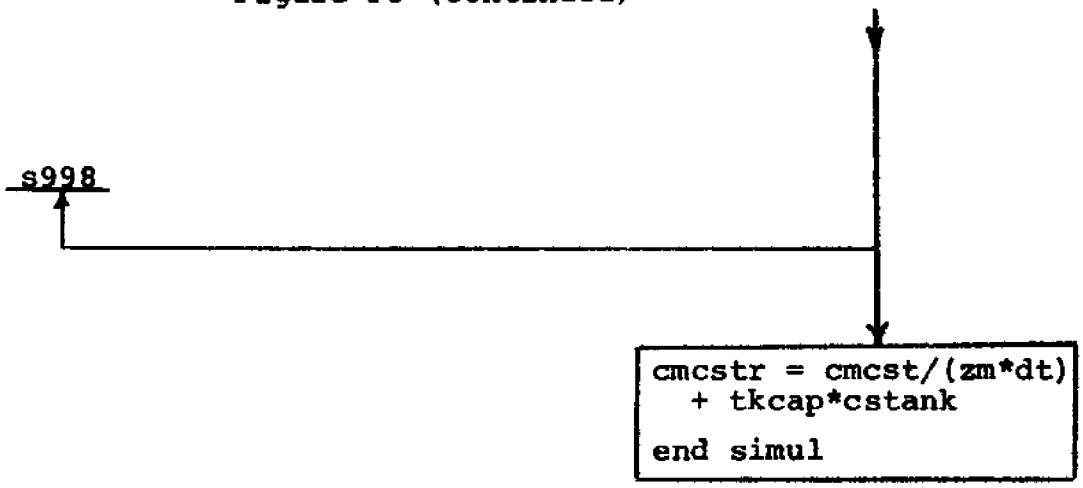


Figure F6

Program Logic

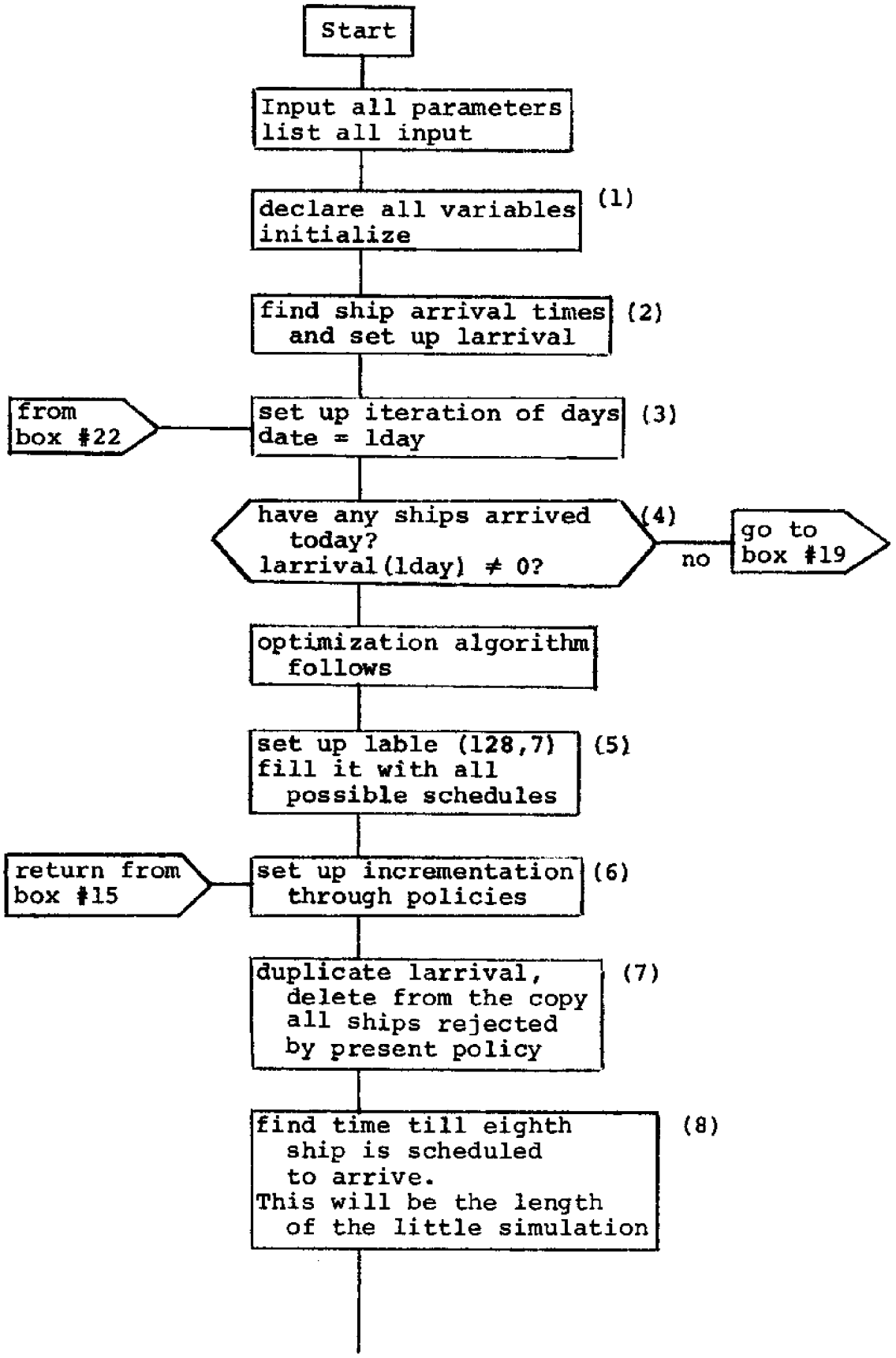


Figure F6 (continued)

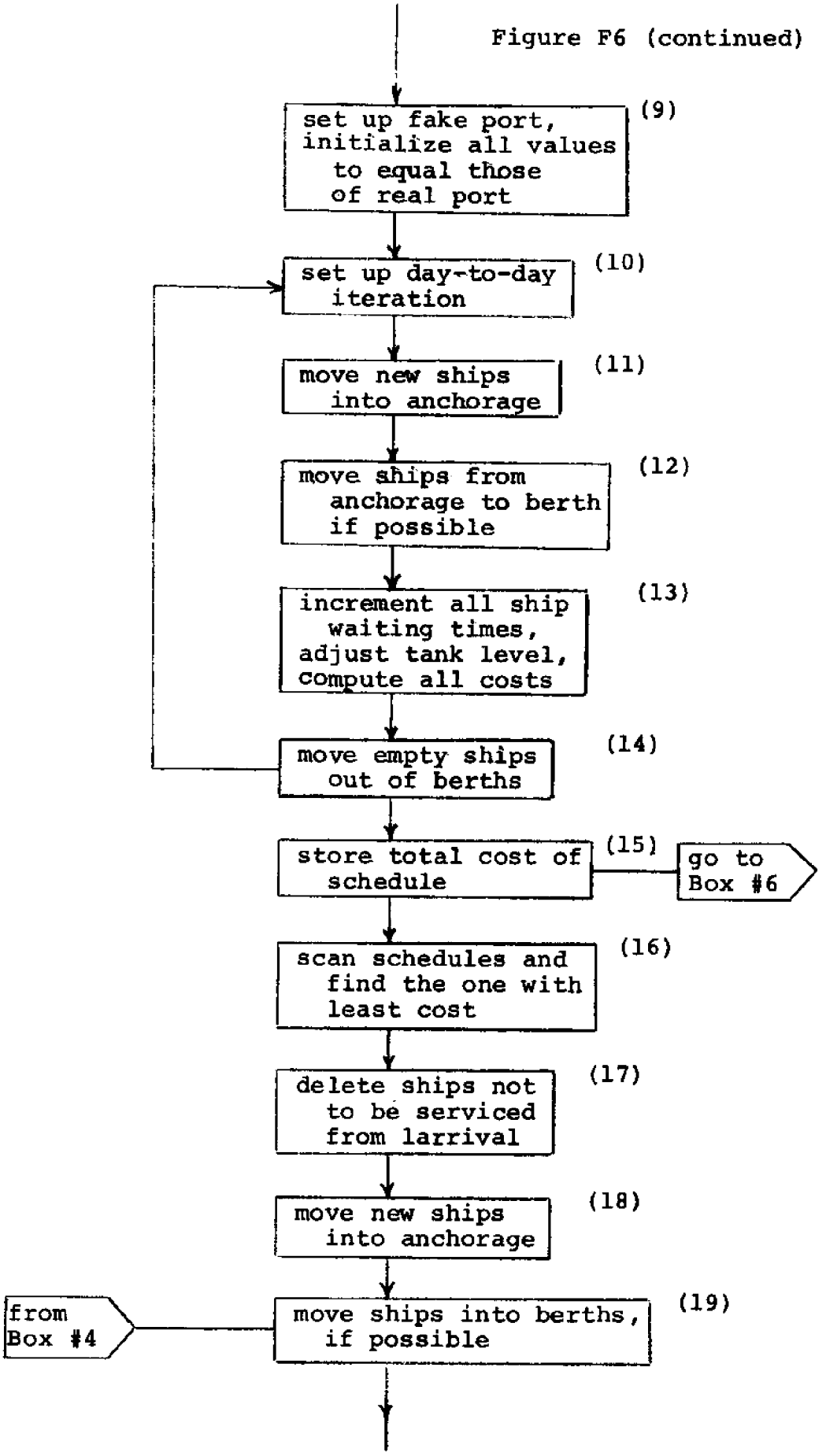
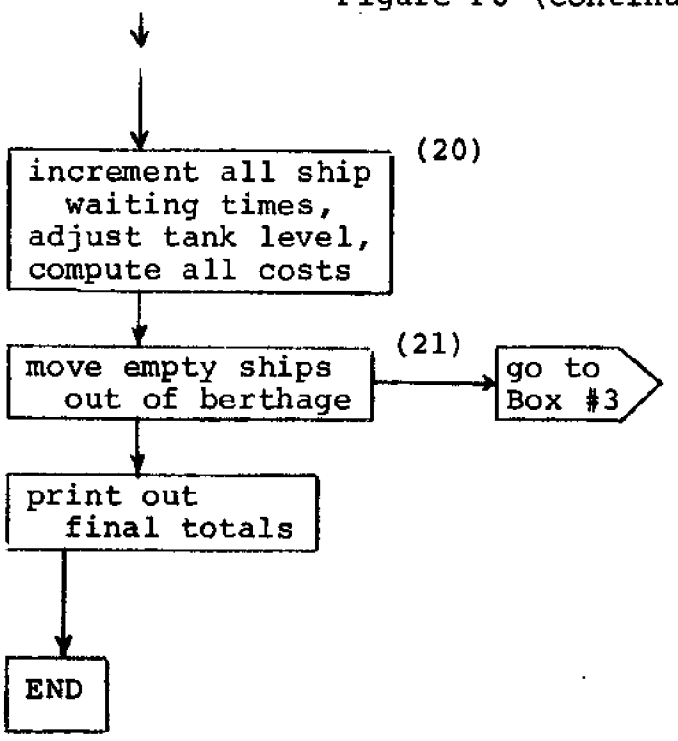


Figure F6 (continued)



BLANK

APPENDIX G

PORT SIMULATION PROGRAM DETAILS

The computer simulation program for determining port costs was written in Fortran. The program consists of one main routine in association with 21 subprograms. The major subprograms are as follows:

1)	GENERA	Generation of ship arrivals.
2)	FEASI	Set matrix of feasible berths for each ship arrival.
3)	ASSIGN	Preschedules ships to berths.
4)	DEV	Sets durations in ship arrival times and sets cargo volumes for each arrival.
5)	SIML	Monitors simulation events.
6)	WHEN	Determines next event for simulation and in association with the simulation events.
7)	RELEAS	Calculates berth release times.
8)	TBIG	Calculates terminal capacity at any simulation time.
9)	LITT	Calculate time for terminal to reach a given capacity.
10)	TERM	Calculates load/unload rate for each service unit when terminal limit is reached.
11)	CARGO	Calculates import and export cargo volumes remaining in each berthed ship.
12)	NEXT	Determines next ship scheduled for a given berth.
13)	IDPT	Calculates berth preference rating for a ship and berth combination.
14)	END	Accumulates and prints simulation statistics.

In the following paragraphs the main program and the various subroutines will be described in further detail.

Main Program

The main program serves to link together the operational subroutines and provide input statements for most of the simulation initiation data. It calls, in turn, REDIN, to read further data, GENERA, ASSIGN, DEV and SIML and it provides the looping function to pass to subsequent time periods and to loop over the subsystem numbers.

Within the main program the array PARMAX receives a sequential listing of ship arrivals from GENERA. This listing is ordered by ship priority with arrival sequence order maintained within each priority level.

At the close of each time period, the main program deletes, from PARMAX, those ships that have passed through the system. GENERA then adds new arrivals for the next period into PARMAX.

REDIN

The subroutine REDIN is used to enter some terminal descriptive data and to initiate certain terminal parameters.

GENERA

The GENERA subprogram operates the list of ship arrivals for a single time period and assigns ship characteristics to each arrival. The ship arrival times and the parameters representing the ship characteristics are stored into the array PARMAX.

Ship arrivals are generated by accumulating interarrival times, each of which is generated through random selection from an exponential distribution. From the function,RANDU,a random

number is first obtained from the uniform distribution over the interval 0 1. The interarrival time, then, is the value of X since that:

$$R = \int_0^X \frac{1}{m} e^{-\frac{x}{m}} dx$$

where R is the random number on the interval 0→1 and m is the mean interarrival time, then:

$$X(\text{interarrival time}) = -m \log(1-R)$$

but since 1-R is also a random number:

$$X = -m \log(R)$$

The ship characteristics table is stored in the array FPAR. Each row of this array contains parameters representing values of the various ship characteristics and describing a single ship type. The initial entry to each row is a relative frequency of occurance of this ship type and the variable NFEQ is the total of the relative frequency counts. Again the function RANDU is used to generate a random number, R, from a uniform distribution on the interval 0→1 so that R*NFEQ is some number in the interval 0→NFEQ. The value of R*NFEQ is then compared to the accumulated relative frequency count to determine in which interval it falls and thus identify one of the listed ship types. The parameters of the corresponding row of FPAR are then entered into the PARMAX matrix.

FEASI

The subprogram FEASI determines, by matching ship type to berth type and ship length and draft to berth length and depth, which berths are feasible for each arriving ship. This routine generated the matrix A(S,B) where S is the count of ships and B is the berth number. The value of each element of A(S,B) is zero if the ship and berth do not match, 1 otherwise.

MAXO

The subprogram MAXO is used by MAIN in sorting ship arrivals by priority and time. The routine determines the earliest arriving ship from among the lightest priority arrivals.

ASSIGN

The subroutine ASSIGN is responsible for prescheduling ships to berths - based on the expected arrival times. Before entry to this subprogram, all ship arrivals in the PARMAX array will have been sorted, in MAIN, into priority order and, within each priority into arrival order. The array KAVAL is used in ASSIGN to indicate availability status of all berths. If a given berth is expected to be or become free within the current time period, the corresponding element of KAVAL is set to 0. Otherwise its value is 1. The array TREL also contains one element for each berth in the port system. The values of the elements of TREL indicate the expected release times for the corresponding berths.

Each ship arrival entered into the PARMAX matrix is examined in sequence and from among the feasible berths for that ship, the one with release time closest to, but still earlier than, the ships expected arrival time, is selected. If no feasible berths become free before the ship arrival, the first feasible berth to be freed will be chosen. This berth becomes the prescheduled berth for that ship. If no feasible berths are available during the current time period, the ship is placed in a queue status - no berth prescheduled. Once a berth has been prescheduled to a given arrival, that berth's release time is advanced by the expected service time for the arriving ship, where expected service time is based on cargo volumes expected, no. of service units available, and load/unload rate of service units. If the new release time extends beyond the end of the current time period the berth is immediately placed in the "not available" status (KAVAL=1).

DEV

The subroutine DEV generates and applies random deviations from schedule to ship arrival time and to cargo import and export volumes. Normally distributed, random variates are determined from the routine GAUSS.

SIML

The subroutine SIML handles the actual simulation of a single time period. This routine is entered at the start of each time period. In an itterative procedure it determines a next event time, advances the simulation clock to that time, identifies the type of event and then enters the activities sequency appropriate to that event. This process is repeated until the event "end of time period" is encountered when the SIML routine is exited.

WHEN(IERLY,IF,JF,KF)

The subroutine WHEN is the routine that actually determines the next event from the tables of event times. It searches over berth release times, ship arrival times, terminal limit times and end of time period to determine the earliest impending event. On exit from WHEN the variable IERLY contains the next event time, the variable IF contains a code identifying the event type, the variable JF contains the port subsystem number in which the event occurs, and the variable KF contains any berth number or ship number that may be required to identify the event location. For example, if the next event is a berth release, the KF variable will hold the berth number.

BERTRL(J,K,IF)

The BERTRL subroutine carries out the activities sequence called for when the next event is either "ship overdue" or "berth release." Details of these activities, along with flow charts,are presented in Sections 3.3.4b) and c). The input variables (J,K,IF)

to BERTRL are, respectively - subsystem number, berth number, and event code.

NEXT(J,L)

The subroutines NEXT, IDPT, FLOW, LITT, TBIS, CARGO, and RELEAS are used primarily by the BERTRL routine. NEXT is used to determine, for any berth at any time, the next ship scheduled for occupancy. Again J and L are input variables containing subsystem number and berth number.

IDPT(J,L,M,IR)

The routine IDPT determines, for a given ship, the expected service time at a specified berth. This is calculated from the cargo volume to be loaded/unloaded, the number of service unit expected to be available and the expected rate of load/unload per service unit. The calculated service time is added to the berth availability time to give an expected time of release (IR) for that ship at that berth. The input variables J,L,M are, respectively,subsystem number, berth number and ship number.

FLOW(J,K)

The routine FLOW calculates, for terminal K of subsystem J the value

$$\frac{V_{m_i} - V_{x_i}}{V_{m_i} + V_{x_i}} u_i$$

where summation is over all ships at the terminal, and V_m, V_x and u are import, export cargo volumes and number of service units at each ship.

This value is used in calculating ship service rates when the terminal is full or empty.

LITT(J,K,T,L1,TO)

The routine LITT calculates, for terminal K of system J the time at which terminal storage will reach capacity T. The equations involved in this calculation are presented in Section 3.3.2.3. The variables TO and LI are, respectively, the initial terminal capacity and initial time.

TBIG(J,K,L1,LO,TO)

The subroutine TBIG is essentially the inverse of LITT. It calculates the terminal capacity, for terminal K of subsystem J, at time L1, given LO and to the initial time, and initial capacity of the terminal. Again the relevant equations are presented in Section 3.3.2.3.

CARGO(J,K,L,M)

This routine determines the volume of cargo remaining to be loaded/unloaded for ship M, at berth L, terminal K and subsystem J. This is calculated from given initial cargo volumes and times and from the number of service units and their rates of loading/unloading.

RELEAS(J,K,L,M)

The routine RELEAS determines, from load/unload rates of service units, number of service units and cargo volume remaining, the expected release time of ship M at berth L of terminal K in subsystem J.

TERM(J,K)

The subroutine TERM is called when the event "terminal limit" is encountered. Within this routine the new load/unload rates for service units are calculated to match the terminal throughput balance requirements. Again J and K refer to subsystem and terminal number, respectively. TERM calls on the routine FLOW in calculating service rates.

END

The subroutine END is entered after SIML encounters the event "end of time period." Within this subroutine, queue statistics and throughput statistics are saved and printed out. The printout statistics are described in Section 3.3.5.2

IROUND(A)

This routine rounds the real variable A to an integer variable.

RANDU and GAUSS

RANDU and GAUSS are both IBM supplied random number generators. RANDU generates numbers from a uniform distribution on the interval $0 \rightarrow 1$. GAUSS generates numbers from a standard normal distribution - mean zero and standard deviation of one.

USER INSTRUCTIONS

Input data to the port simulation model has been described in Section 3.3.5.1. In the following section, formatting details for data input will be presented:

1 Card 1 - variables NDAYS, N9, N8

- format (3I3)

NDAYS is number of days of simulation

N9 is the output unit for detailed simulation printout

N8 is the output unit for statistical summaries

2 Card 2 - variables IX1, XTIME, ITIME, NSYS, NN

- format (I6,4I3)

IX1 - seed for interarrival time distribution

XTIME - numbers of time units per time period

ITIME - initial simulation time

NSYS - number of subsystems in the port

NN - initial ship count number

3 Card 3 - variables IX, NSAM, NFEQ, NSPO, NP

- format (I6,4I3)

IX - seed for ship type distribution

NSAM - number of ship types

NFEQ - total relative frequency count for ship type (see Section 6 above)

NSPO - fixed value of 10

NP - fixed value of 14

4 Card 4 - variables FPAR array

- format (14I5)

One Card 4 for each of NSAM ship types.

FPAR-1 - relative frequency count

2 - ship type

3 - import cargo volume

4 - export cargo volume

5 - maximum ship cargo capacity

6 - not used

7 - maximum number of service units the ship can use

8 - not used

9 - not used

10 - subsystem number

11 - ship length

12 - ship draft

13 - not used

14 - not used

5 Card 5 - variables NPR array

- format(3I3)

NPR(J) is 1 or 0 index to indicate whether priority is used or not for system J.

6 Card 6 - variables JCAT, JPAM

- format(2I3)

JPAM - twice the number of systems for which priority is used

JCAT - number of ship types - up to 5

7 Card 7 - variables PRMTX array

- format(5I3)

As many card 7's as the number JPAM. Every other card (even numbered cards) is not used and may be left blank. The remaining cards indicate priorities to be assigned to each ship type within each subsystem. For example, the 2nd card lists priorities for the nth subystem for which priorities are used. PRMTX(2n,k) is priority for kth ship type in system n.

8 Card 8 - variables LTYPE, LIMPO, LEXPO, LSIZE, LBUIM, LBUEX, LTYPI, LTYPX

- format(8I2)

LTYPE - the value 01

LIMPO - the value 02

LEXPO - the value 03

LSIZE - the value 04

Remaining variables may be zero.

9 Card 9 - variable NPB

- format(I3)

NPB - the value 11

10 Card 10- variables NNB

- format(3I3)

NNB(J) - number of berths in system J

11 Card 11 - variable NIDI

- format(10I3)

NIDI(J) - raise in priority for ships remaining in queue at end of a time period - system J

12 Card 12 - variable ILAM

- format(I6)

ILAM - mean interarrival time

13 Card 13 - variable NCOMP

- format(I5)

NCOMP - total number of terminals within the port

14 Card 14 - variable SRATE1, RATE, CAP arrays

- format(3F6.0)

One Card 14 for each terminal (NCOMP) in the port.

SRATE1(k) - load/unload rate maximum for service units.

RATE(k) - maximum flow rate through land side of terminal

CAP(k) - terminal maximum storage capacity each for terminal k

15 Card 15 - variable MPLX array

- format(6I6)

One Card 15 for each terminal.

16 Card 16 - variables IRRD, IRRL, Ix3, Ix4

- format(6I6)

IRRD - standard error in ship arrival time deviations

IRRL - standard errors in ship cargo volume

Ix3 - random generator seed for ship arrival time deviation

Ix4 - random generator seed for ship cargo volume

17 Card 17 - variables PARBER array

- format(11I5)

One card for each berth in the port. These cards must be sorted by subsystem number.

PARBER(J,I,1) - not used on input

(J,I,2) - berth identification number

(J,I,3) - berth type identifier

(J,I,4) - berth length

(J,I,5) - berth depth

(J,I,6) - not used on input

(J,I,7) - not used on input

(J,I,8) - not used on input

(J,I,9) - not used on input

(J,I,10) - terminal to which berth belongs

(J,I,11) - not used on input

Each for berth I of system J.

APPENDIX H

Approximate Fixed Charge Transportation Algorithm (Out of Kilter Algorithm)

```
/*EOJ ********
// 'CHELST',REGION=250K,CLASS=A
//C.SYSIN DD *
      SUBROUTINE MAINE                                                  OKF00030
      DIMENSION KL(3500),KC(3500),KU(3500),KX(3500),JI(3500),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(3500),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9),KE(101)
       DIMENSION NOPORT(9),TRAN(18,30),F(9,3),P(9),NFIX(9,9),NF(9),NCOST
     CR(9,9),PY(9,3),NCOST1(7,3,9),NCAP1(7,3,9),KCLOSE(9,3),NTRACK(9)
      DIMENSION  MTRACK(9),NOPT9(10)
      DIMENSION MAXOPT(9)
      COMMON/KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      OKF00080
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER     OKF00090
      COMMON /MINF/MINF/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K     OKF00100
      COMMON/KPORT/KPORT/TRAN/TRAN/KCUM/KCUM/RUN/RUN/CAPARC/CAPARC
      COMMON /MOCOST/MOCOST/NCOSTR/NCOSTR/NOCOST/NOCOST/LOCOST/LOCOST/M
     CAX/MAX/NOPORT/NOPORT/NCHECK/NCHECK/NOARCS/NOARCS
      COMMON/NOHELP/NOHELP
      INTEGER TRAN,TALTER,R,SOURC1,SOURC2,P,ZERO,TCOMPU,UTE,CAPARC,TR,F,
     -PY,OUT1,OUT2,OPEN
      DATA NOPT9/10*3/
      DATA MAXOPT/9*1/
      DATA F/1500000,4700000,3000000,1500000,5000000,7800000,17000000,15
     C000000,4200000,750000,2350000,1500000,750000,2500000,3900000,85000
     C00,7500000,2100000,9*0/
      DATA NTRACK,MTRACK/18*0/
      DATA NCOST1/10,15,15,20,45,125,380,10,15,15,20,45,125,380,10,15,15
     C,20,45,125,380,5*3,14,60,5*3,14,60,5*3,14,60,10,3*13,18,70,175,10,
     C3*13,18,70,175,10,3*13,18,70,175,10,15,15,20,45,125,380,10,15,15,2
     C0,45,125,380,10,15,15,20,45,125,380,2,3,3*4,10,69,2,3,3*4,10,69,
     C 2,3,3*4,10,69,2,5,8,8,9,15,50,2,5,8,8,9,15,50,2,5,8,8,9,15,50,4*3
     C,7,29,40,4*3,7,29,40,4*3,7,29,40,2*3,4,4,8,32,85,2*3,4,4,8,32,85,
     C2*3,4,4,8,32,85,5,10,10,20,40,150,167,5,10,10,20,40,150,167,5,10,
     C10,20,40,150,167/
      DATA NCAP1/5*100000,40000,50000,5*50000,20000,25000,7*0,7*200000,
     C7*100000,7*0,5*200000,100000,40000,5*100000,50000,20000,7*0,100000
     C,4*100000,40000,50000,5*50000,20000,25000,7*0,2*4000000,2*2000000,
     C2*4000000,800000,2*2000000,2*1000000,2*2000000,400000,7*0,400000,3
     C00000,100000,2*200000,400000,100000,200000,150000,50000,2*100000,2
     C00000,50000,7*0,6*500000,300000,6*250000,150000,7*0,4*500000,2*250
     C000,200000,4*250000,2*125000,100000,7*0,400000,100000,300000,10000
     C0,300000,40000,60000,200000,50000,150000,50000,150000,20000,30000,
     C7*0/
      DATA  KCLOSE/27*11/
      DATA P/4HP 1 ,4HP 2 ,4HP 3 ,4HP 4 ,4HP 5 ,4HP 6 ,4HP 7 ,4HP 8 ,4HP
     C 9 /
      DATA TALTER,R,SOURC1,SOURC2,ZERO/4HALTE,2HR ,4HS    ,2*4H    /
      DATA TCOMPU,UTE/4HCOMP,4HUTE /
      DATA L,NALTER,NF,PY,NFIX/11*0,9*1,18*0,81*200000/
      NOHELP=0
      LCOUNT=1
      LOCOST=400000000
      DO 731 IJK=1,9
  731 NOPORT(IJK)=11
      NOCOST=LOCOST
      RUN=1
      NOARCS=7
      KPORT=1
      NOPT1=1
      MAX=9
```

```
C    SYSTEM INPUT TAPE                                                  OKF00110
       KI=5
C    SYSTEM OUTPUT TAPE                                                 OKF00130
       KO=6
C    CARD PUNCH                                                         OKF00150
       KO(1)=7                                                          OKF00160
C    RESERVED OUTPUT TAPE                                               OKF00170
       KO(2)=3                                                          OKF00180
C    RESERVED INPUT TAPE                                                OKF00190
       KO(3)=2                                                          OKF00200
C    MAXIMUM ARCS                                                       OKF00210
       KO(4)=3500
C    MAXIMUM NODES                                                      OKF00230
       KO(5)=500
       KO(6)=0                                                          OKF00250
       KO(9)=0                                                          OKF00260
       IFIN=32767                                                       OKF00270
   100 CALL PREDAT(KS)                                                  OKF00280
   101 IF(KS.NE.0) GO TO 1                                              OKF00290
       CALL ARCASY(LO)
       CALL MAKEJL                                                      OKF00310
       IF(LER.GE.KO(4)) GO TO 98
       LER=LER*KO(9)                                                    OK60  0
       IF(LO.EQ.0) GO TO 1
       CALL NODASY
     1 CALL READER
       CALL TRANSL
       IF(LER.NE.0) GO TO 88
       I=0
       KUP=1
       KE(101)=0
       DO 26 K=1,N
       IF(K.LT.KUP) GO TO 38
       I=I+1
       KUP=LOCR(IL(I+1))
    38 CALL KILTER(I)
       IF(LER.EQ.0) GO TO 26
       IF(LER.NE.107) GO TO 24
       KE(101)=KE(101)+1
       KF=KE(101)
       IF(KE(101).GT.100) GO TO 26
       KE(KF)=K
    24 CONTINUE
C    COMPLETED CHECKING ALL ARCS
       LER=0
    99 CALL OUTPUT(KE)
       WRITE(6,999) KCUM,((KCLOSF(IJKL,IJKLM),IJKL=1,9),IJKLM=1,3)
   999 FORMAT(I10,27I3)
       MOCOST=KCUM
       DO 997 IJK=1,9
       NO1OPT=NOPT9(IJK)
       DO 998 IJKL=1,NO1OPT
   998 MOCOST=MOCOST+PY(IJK,IJKL)*(F(IJK,IJKL)+NF(IJK))
   997 CONTINUE
       WRITE(6,57) MOCOST,(NOPORT(IJKL),IJKL=1,9),NOCOST
    57 FORMAT(20H0 THE TOTAL COST IS ,I10,11H AND PORTS ,8(I3,1H,),4HAND
      C,I4,12H ARE CLOSED.,I10)
       CALL TIMING(ICPU,IEXCP)
       WRITE(6,8888) ICPU,IEXCP
  8888 FORMAT(1H0,2I10)
```

```
      IF (LCOUNT.EQ.3) GO TO  792
      IF(KPORT.EQ.10) GO TO 700
      IF(NOHELP.EQ.1)  GO TO 10
      IF(NOHELP.EQ.1)  KCLOSE(KPORT,NOPT1)=1
      IF(MOCOST.GE.NOCOST) KCLOSE(KPORT,NOPT1)=1
      IF(MOCOST.GE.LOCOST) GO TO 10
      MPORT=KPORT
      MOPT1=NOPT1
      LOCOST=MOCOST
   10 NALTER=0
 6000 IF(KPORT.EQ.1.AND.NOPT1.EQ.1) NOCOST=LOCOST
      IF(L.EQ.10.AND.LCOUNT.EQ.1) GO TO 78
      IF(L.EQ.10.AND.LCOUNT.EQ.2) GO TO 79
   61 IF(NOPT1.NE.NOPT9(KPORT)) GO TO 11
  610 KPORT=KPORT+1
      NOPT1=1
   11 NOPT1=NOPT1+1
      IF (KPORT.EQ.10) GO TO 65
      IF(NOPORT(KPORT).NE.11.AND.NCOUNT.EQ.2) GO TO 610
      IF(NOPT1.GT.NOPT9(KPORT).AND.NCOUNT.EQ.2) GO TO 610
      IF(NOPT1.GT.NOPT9(KPORT)) GO TO 62
      IF(NOPORT(KPORT).NE.11) GO TO 62
      IF(KCLOSE(KPORT,NOPT1).NE.11) GO TO 62
      NALTER= NALTER+1
      PY(KPORT,MAXOPT(KPORT))=0
      PY(KPORT,NOPT1)=1
      IN1= NOARCS*(NALTER-1)+2
      IN2=NOARCS*NALTER+1
      DO 250 NCLOSE=IN1,IN2
      TRAN(1,NCLOSE)=TALTER
      TRAN(2,NCLOSE)=R
      TRAN(3,NCLOSE)=SOURC1
      TRAN(4,NCLOSE)=SOURC2
      TRAN(5,NCLOSE)=P(KPORT)
      TRAN(6,NCLOSE)= ZERO
      TRAN(7,NCLOSE)=NCLOSE-IN1+1
      TRAN(8,NCLOSE)=NCOST1(NCLOSE-IN1+1,NOPT1,KPORT)
  250 TRAN(9,NCLOSE)=NCAP1(NCLOSE-IN1+1,NOPT1,KPORT)
      JUMP=0
      GO TO 63
   62 JUMP=1
   63 IF(MCOUNT.EQ.2) GO TO 65
      IF(NOPT1.NE.1) PY(KPORT,NOPT1-1)=0
      MCOUNT=2
   65 IF(NOPT1.NE.2) GO TO 66
      IF(KPORT.EQ.1) NCOUNT=2
      IF(NCOUNT.EQ.2) GO TO 660
      NCOUNT=2
C     OPEN PORT KPORT-1
      NALTER=NALTER +1
      PY(KPORT-1,NOPT9(KPORT-1))=0
      PY(KPORT-1,MAXOPT(KPORT-1))=1
      OUT1=NOARCS*(NALTER-1)+2
      OUT2=NOARCS*NALTER+1
      DO 300 OPEN=OUT1,OUT2
      TRAN(1,OPEN)=TALTER
      TRAN(2,OPEN)=R
      TRAN(3,OPEN)=SOURC1
      TRAN(4,OPEN)=SOURC2
      TRAN(5,OPEN)=P(KPORT-1)
```

```
      TRAN(6,OPEN)=ZERO
      TRAN(7,OPEN)=OPEN-OUT1+1
      TRAN(8,OPEN)=NCOST1(OPEN-OUT1+1,MAXOPT(KPORT-1),KPORT-1)
  300 TRAN(9,OPEN)= NCAP1(OPEN-OUT1+1,MAXOPT(KPORT-1),KPORT-1)
   66 IF(KPORT.EQ.10) GO TO 661
  660 IF (JUMP.EQ.1) GO TO 61
  661 CONTINUE
      TRAN(1,NOARCS*NALTER+2)=TCOMPU
      TRAN(2,NOARCS*NALTER+2)=UTE
      KS=1
      MCOUNT=0
      NCOUNT=0
      GO TO 101
  700 KPORT=1
      NOPT1=1
      KS=1
      L=L+1
      IF(NOCOST.EQ.LOCOST.AND.LCOUNT.EQ.1) GO TO 78
      IF(NOCOST.EQ.LOCOST.AND.LCOUNT.EQ.2) GO TO 79
C     CLOSE PORT MPORT PERMANENTLY
      NOPORT(MPORT)=MPORT
      IF(MOPT1.NE.NOPT9(MPORT)) PY(MPORT,MOPT1)=1
      PY(MPORT,MAXOPT(MPORT))=0
      KCLOSE(MPORT,MOPT1)=2
      MAXOPT(MPORT)=MOPT1
      LIN1=2
      LIN2=NOARCS+1
      DO 400 NCLOSE=2,LIN2
      TRAN(1,NCLOSE)=TALTER
      TRAN(2,NCLOSE)=R
      TRAN(3,NCLOSE)=SOURC1
      TRAN(4,NCLOSE)=SOURC2
      TRAN(5,NCLOSE)=P(MPORT)
      TRAN(6,NCLOSE)=ZERO
      TRAN(7,NCLOSE)=NCLOSE-1
      TRAN(8,NCLOSE)=NCOST1(NCLOSE-1,MOPT1,MPORT)
  400 TRAN(9,NCLOSE)=NCAP1(NCLOSE-1,MOPT1,MPORT)
      JUMP=0
      NALTER=1
      NCOUNT=2
      MCOUNT=2
      GO TO 6000
   24 WRITE(KO,54)
      LL=LADDR(IJ(K))
      WRITE(KO,55) NN(2*I-1),NN(2*I),NN(2*LL-1),NN(2*LL)                OKF00610
      GO TO 99
   88 WRITE(KO,56)
      STOP
   51 FORMAT(A4)
   54 FORMAT(24H0OVERFLOW IN NODE PRICES)                               OKF00660
   55 FORMAT(23H0RUN TERMINATED AT ARC ,4A4)
   56 FORMAT(37H0RUN TERMINATED DUE TO ERRORS IN DATA)                  OKF00680
   78 L=0
      WRITE(6,800)
  800 FORMAT(12H1 STAGE TWO )
      LCOUNT=2
      KPORT=1
      NOPT1=1
      MCOUNT=0
      NCOUNT=0
```

```
      DO 783 IJKL=1,9
      DO 782 IJK=1,MAX
  782 NF(IJKL)=NF(IJKL)+NFIX(IJKL,IJK)
      NO1OPT=NOPT9(IJKL)
      DO 784 IJKLM=1,NO1OPT
      IF(KCLOSE(IJKL,IJKLM).NE.2) KCLOSE(IJKL,IJKLM)=11
      MTRACK(IJKL)=MTRACK(IJKL)+PY(IJKL,IJKLM)
  784 NOCOST=NOCOST+PY(IJKL,IJKLM)*NF(IJKL)
  783 IF(MTRACK(IJKL).NE.0) NOPORT(IJKL)=11
      LOCOST=NOCOST
      GO TO 61
   79 DO 791 IJKL=1,9
      NO1OPT=NOPT9(IJKL)-1
      DO 790 IJKLM=1,NO1OPT
  790 MOCOST=MOCOST-PY(IJKL,IJKLM)*NF(IJKL)
      NF(IJKL)=0
  791 CONTINUE
      DO 796 IJKL=1,9
      NO1OPT=NOPT9(IJKL)
      DO 794 IJK=1,NO1OPT
      WRITE(6,803) MTRACK(IJKL)
  803 FORMAT(1H0,I10)
  794 MTRACK(IJKL)=MTRACK(IJKL)+PY(IJKL,IJK)
      IF(MTRACK(IJKL).NE.0) NOPORT(IJKL)=11
      DO 795 IJK=1,MAX
  795 NCOSTR(IJKL,IJK)=MTRACK(IJKL)
  796 CONTINUE
      WRITE(6,801)
  801 FORMAT(13H1 STAGE THREE)
      WRITE(6,802) MOCOST,(MTRACK(IJKL),IJKL=1,9)
  802 FORMAT(1H0,I11,9I8)
      LCOUNT=3
  792 CALL TRADER
      IF (NCHECK.EQ.1) GO TO 80
      GO TO 101
   80 RETURN
      END
      SUBROUTINE TRADER
      DIMENSION NCOSTR(9,9),NFIX(9,9),NOPORT(9),NOPOR2(9,9),KOPEN3(9,9),
     CTRAN(18,30),KOPEN1(9),NCOST2(9,9,2)
      DIMENSION NCAPAR(9,9,2),ND(9),ND0(9),NR(9),NR0(9)
      COMMON /NCAPAR/NCAPAR/KCUM/KCUM/NCOST2/NCOST2
      COMMON /MOCOST/MOCOST/NCOSTR/NCOSTR/NOCOST/NOCOST/LOCOST/LOCOST/M
     CAX/MAX/NOPORT/NOPORT/NCHECK/NCHECK/TRAN/TRAN/NOARCS/NOARCS
      COMMON/NOHELP/NOHELP
      INTEGER TALTER,R,TCOMPU,UTE,TRAN,OPEN
      DATA KLPORT,L,KOUNT,KROOT1,KROOT2,NALTER/2*1,4*0/
      DATA TALTER,R,TCOMPU,UTE/4HALTE,2HR ,4HCOMP,4HUTE /
      DATA NR/4HR 1 ,4HR 2 ,4HR 3 ,4HR 4 ,4HR 5 ,4HR 6 ,4HR 7 ,4HR 8 ,4H
     CR 9 /
      DATA ND/4HD 1 ,4HD 2 ,4HD 3 ,4HD 4 ,4HD 5 ,4HD 6 ,4HD 7 ,4HD 8 ,4H
     CD 9 /
      DATA KOPEN3,NOPOR2,KOPEN1/81*11,81*11,9*11/
      DATA NP0/2H1 ,2H2 ,2H3 ,2H4 ,2H5 ,2H6 ,2H7 ,2H8 ,2H9 /
      DATA ND0/2H1 ,2H2 ,2H3 ,2H4 ,2H5 ,2H6 ,2H7 ,2H8 ,2H9 /
      DATA NFIX/81*200000/
      NCOUNT=0
      DO 2 ILN=1,MAX
      DO 2 ILM=1,9
    1 MOCOST=MOCOST+NCOSTR(ILM,ILN)*NFIX(ILM,ILN)
```

```
    2 CONTINUE
      WRITE(6,900) KCUM,MOCOST,KLPORT,KROOT1,L
  900 FORMAT(9HOKCUM IS ,I10,11H MOCOST IS ,I10,7H KLPORT,I3,7H KROOT1,
     CI3,2H L,I3)
      KOUNT=KOUNT+1
      IF (KOUNT.EQ.1) GO TO 11
      IF (KLPORT.EQ.10) GO TO 700
      IF(NOHELP.EQ.1)  KOPEN3(KLPORT,KROOT1)=1
      IF(NOHELP.EQ.1)  GO TO 10
      IF(MOCOST.GE.NOCOST) KOPEN3(KLPORT,KROOT1)=1
      IF (LOCOST .LE. MOCOST) GO TO 10
      MPORT=KLPORT
      MROOT1=KROOT1
      LOCOST=MOCOST
   10 NALTER=0
 6000 IF(KLPORT.EQ.1.AND.KROOT1.EQ.0) NOCOST=LOCOST
      IF (L.EQ.81) GO TO  80
   61 IF (KROOT1.NE.MAX) GO TO 11
  610 KLPORT=KLPORT+1
      KROOT1=0
   11 KROOT1=KROOT1+1
      IF(KLPORT.EQ.10) GO TO 65
      IF(NOPORT(KLPORT).NE.11.AND.NCOUNT.EQ.2) GO TO 610
      IF(KOPEN1(KLPORT).NE.11.AND.NCOUNT.EQ.2) GO TO 610
      IF(NOPORT(KLPORT).NE.11.OR.KOPEN1(KLPORT).NE.11) GO TO 62
      IF(NOPOR2(KLPORT,KROOT1).NE.11.OR.KOPEN3(KLPORT,KROOT1).NE.11) GO
     C TO  62
C       CLOSE PORT TRADE ROUTE TEMPORARILY
      NALTER= NALTER+1
      NCOSTR(KLPORT,KROOT1)=0
      NCLOS1=NALTER*2
      NCLOS2=NALTER*2+1
      DO 250 NCLOSE=NCLOS1,NCLOS2
      TRAN(1,NCLOSE)=TALTER
      TRAN(2,NCLOSE)=R
      TRAN(3,NCLOSE)=ND(KLPORT)
      TRAN(4,NCLOSE)=NDO(KROOT1)
      TRAN(5,NCLOSE)=NR(KLPORT)
      TRAN(6,NCLOSE)=NRO(KROOT1)
      TRAN(7,NCLOSE)=NCLOSE-NCLOS1+1
      TRAN(8,NCLOSE)=0
  250 TRAN(9,NCLOSE)=0
      JUMP=0
      GO TO 65
   62 JUMP=1
   65 IF(NCOUNT.EQ.2) GO TO 66
      IF(KROOT1-1.EQ.0.AND.KLPORT-1.EQ.0) NCOUNT=2
      IF(KROOT1-1.EQ.0.AND.KLPORT-1.EQ.0)GO TO 66
      IF(KROOT1-1.NE.0) GO TO 720
      KLP=KLPORT-1
      KRO1=MAX
      GO  TO 725
  720 KLP=KLPORT
      KRO1=KROOT1-1
C       REOPEN PORT-TRADE ROUTE COMBINATION
  725 NCOUNT=2
      NALTER=NALTER+1
      NCOSTR(KLP,KRO1)=1
      NOPEN1=2*NALTER
      NOPEN2=2*NALTER+1
```

```
      DO 300 OPEN=NOPEN1,NOPEN2
      TRAN(1,OPEN)=TALTER
      TRAN(2,OPEN)=R
      TRAN(3,OPEN)=ND(KLP)
      TRAN(4,OPEN)=NDO(KRO1)
      TRAN(5,OPEN)=NR(KLP)
      TRAN(6,OPEN)=NRO(KRO1)
      TRAN(7,OPEN)=OPEN-NOPEN1+1
      TRAN(8,OPEN)=NCOST2(KLP,KRO1,OPEN-NOPEN1+1)
  300 TRAN(9,OPEN)=NCAPAR(KLP,KRO1,OPEN-NOPEN1+1)
   66 IF(KLPORT.EQ.10) GO TO 661
      IF (JUMP .EQ. 1) GO TO 61
  661 CONTINUE
      TRAN(1,2*NALTER+2)=TCOMPU
      TRAN(2,2*NALTER+2)=UTE
      KS=1
      RETURN
  700 KLPORT=1
      KS=1
      KROOT1=0
      KROOT2=0
      IF (NOCOST.EQ.LOCOST) GO TO 80
      NOPOR2(MPORT,MROOT1)=1
C      CLOSE PORT-TRADE ROUTE PERMANENTLY
      NOCOST=LOCOST
      NCOSTR(MPORT,MROOT1)=0
      DO 850 NCLOSE=2,3
      TRAN(1,NCLOSE)=TALTER
      TRAN(2,NCLOSE)=R
      TRAN(3,NCLOSE)=ND(MPORT)
      TRAN(4,NCLOSE)=NDO(MROOT1)
      TRAN(5,NCLOSE)=NR(MPORT)
      TRAN(6,NCLOSE)=NRO(MROOT1)
      TRAN(7,NCLOSE)=NCLOSE-1
      TRAN(8,NCLOSE)=0
  850 TRAN(9,NCLOSE)=0
      L=L+1
      JUMP=0
      NALTER=1
      NCOUNT=2
      GO TO 6000
   80 NCHECK=1
      RETURN
      END
      SUBROUTINE OUTPUT(KZ)                                          OKF05000
      DIMENSION KL(3500),KC(3500),KU(3500),KX(3500),JI(3500),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(3500),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)                                 OKF05030
      DIMENSION K7(101)                                              OKF05040
      DIMENSION L7(3500)
      INTEGER SUMX(100),SUMC(100)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL  OKF05050
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER  OKF05060
      COMMON /MINE/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K  OKF05070
      COMMON /KCUM/KCUM
      COMMON/NOHELP/NOHELP
      COMMON /KPORT/KPORT
      DATA KILT,BLANK,TEN/1HK,1H ,1HN/                               OKF05080
      NOHELP=0
      KQ1=KQ(1)                                                      OKF05090
```

```
      KCUM=0                                                            OKF05100
      IF(KZ(101).NE.0) GO TO 10                                         OKF05110
      IF(LER.NE.0) GO TO 30                                             OKF05120
      MZ=KILT                                                           OKF05130
      GO TO 100                                                         OKF05140
   10 IF(LER.NE.0) GO TO 18                                             OKF05150
      NOHELP=1
      WRITE(KO,99) KZ(101)
   18 KZ(101)=MINO(KZ(101),100)                                         OKF05170
   30 MZ=BLANK                                                          OKF05180
  100 K2=KQ(2)                                                          OKF05190
      IF(LC(2).EQ.0) GO TO 12                                           OKF05200
      WRITE(KO1,90) (KA(I,2),I=1,18)                                    OKF05210
   12 IF(LC(1).EQ.0) GO TO 41                                           OKF05230
      IF(KQ(6).NE.0) GO TO 24                                           OKF05240
      KQ(6)=1                                                           OKF05250
      REWIND K2                                                         OKF05260
   24 WRITE(K2,90) (KA(I,2),I=1,18)                                     OKF05270
   41 IF(KQ(7).EQ.0) GO TO 7                                            OKF05290
    7 L=1                                                               OKF05310
      LL=1                                                              OKF05320
      DO 3 I=1,M                                                        OKF05330
      LUP=LDECR(IL(I+1))                                                OKF05340
  302 IF(L.GE.LUP) GO TO 3
      LU=LADDR(IJ(L))                                                   OKF05360
      LLC=KC(L)                                                         OKF05370
      KC(L)=KC(L)-NP(I)+NP(LU)                                          OKF05380
      KX(L)=KX(L)+KL(L)                                                 OKF05390
      KU(L)=KU(L)+KL(L)                                                 OKF05400
      LZ(L)=KX(L)*KC(L)
      KCUM=KCUM+LZ(L)
      MX=MZ                                                             OKF05430
      IF(KZ(101).LE.0) GO TO 16                                         OKF05440
      IF(KZ(LL).NE.L) GO TO 16                                          OKF05450
      KZ(101)=KZ(101)-1                                                 OKF05460
      LL=LL+1                                                           OKF05470
      MX=IFN                                                            OKF05480
   16 IF(LC(1).EQ.0) GO TO 51                                           OKF05490
      WRITE(K2,93) NN(2*I-1),NN(2*I),NN(2*LU-1),NN(2*LU),KC(L),KU(L),   OKF05500
     1KL(L),KX(L),LZ(L),MX
   51 IF(KQ(7).EQ.0) GO TO 56
   56 IF(LC(2).EQ.0) GO TO 333                                          OKF05550
      WRITE(KQ1,93)  NN(2*I-1),NN(2*I),NN(2*LU-1),NN(2*LU),KC(L),       OKF05560
     1KU(L),KL(L),KX(L),LZ(L)
  333 L=L+1                                                             OKF05580
      GO TO 302                                                         OKF05590
    3 CONTINUE                                                          OKF05600
      IF(LC(3).EQ.0) GO TO 15                                           OKF05610
      IF(LC(1)+LC(2).EQ.0) GO TO 27                                     OKF05620
      IF(LC(1).EQ.0) GO TO 203                                          OKF05630
      WRITE(K2,96)                                                      OKF05640
  203 IF(LC(2).EQ.0) GO TO 115                                          OKF05650
      WRITE(KQ1,96)                                                     OKF05660
  115 DO 200 I=1,M                                                      OKF05670
      IF(LC(1).EQ.0) GO TO 85                                           OKF05680
      WRITE(K2,95) NN(2*I-1),NN(2*I),NP(I)                              OKF05690
   85 IF(LC(2).EQ.0) GO TO 200                                          OKF05700
      WRITE(KQ1,95) NN(2*I-1),NN(2*I),NP(I)                             OKF05710
  200 CONTINUE                                                          OKF05720
   15 IF(LC(1).EQ.0) GO TO 27                                           OKF05730
```

```
      WRITE(K2,97)                                                      OKF05740
   27 IF(KQ(7).EQ.0) GO TO 57                                           OKF05750
   57 IF(LC(2).EQ.0) GO TO 77                                           OKF05760
      WRITE(KQ1,97)                                                     OKF05790
   77 WRITE(KQ,92) (LC(I),I=5,8)
      RETURN
   88 FORMAT(24H0THIS RUN OUTPUT TO TAPE )                              OKF05820
   99 FORMAT(25H0THIS RUN OUTPUT TO PUNCH)                              OKF05830
   90 FORMAT(18A4/4HARCS22X,4HCOST5X,5HUPPER5X,5HLOWER6X,4HFLOW7X,
     154HCOST)
   91 FORMAT(1H1)8A4/5H ARCS16X,4HCOST6X,5HUPPER6X,5HLOWER7X,4HFLOW7X,
     15HTCOST9X,3HPI19X,3HPI28X,4HCBAR/1X)
   92 FORMAT(1H0NO OF BREAKTHRUS=I12,22H, NO OF NONBREAKTHRUS=I12,1H, OKF05870
     1NO OF X CHANGES=I12,/42H NO OF NODES FROM WHICH LABELING WAS DONE=OKF05880
     2I12)                                                              OKF05890
   93 FORMAT(6X,2(A4,A2),2X,4I10,I12,1X,A1)                             OKF05900
   94 FORMAT(2(1X,A4,A2),4(1X,I10),4I12,1X,A1)                          OKF05910
   95 FORMAT(6X,A4,A2,6X,I12)                                           OKF05920
   96 FORMAT(6HNODES )                                                  OKF05930
   97 FORMAT(3HEND)                                                     OKF05940
   98 FORMAT(4H0END)                                                    OKF05950
   99 FORMAT(1H0I5,23H ARCS ARE OUT OF KILTER)                          OKF05960
      END                                                               OKF05980
      SUBROUTINE READER
      DIMENSION KL(3500),KC(3500),KU(3500),KX(3500),JI(3500),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(3500),NL(500)
      DIMENSION TRAN(18,30)
      DIMENSION LC(9),KA(18,2),KQ(9)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER
      COMMON /MINE/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K
      COMMON /RUN/RUN
      COMMON /TRAN/TRAN
      INTEGER TRAN,TALTER,R,SOURC1,SOURC2,P,ZERO,TCOMPU,UTE,CAPARC,TR
       INTEGER TAPE1,PUNCH1,NODES1,PRINT1
       INTEGER ALTER,OUTPUT,COMPUT,TAPE,PUNCH,NODES,PRINT
      DATA TAPE1,PUNCH1,NODES1,PRINT1/4HAPE ,4HUNCH,4HODES,4HRINT/
      DATA ALTER,OUTPUT,COMPUT/4HALTE,4HOUTP,4HCOMP/
      DATA TAPE,PUNCH,NODES,PRINT/4H TAP,4H PUN,4H NOD,4H PRI/
      DATA NCOUNT/0/
      KCARD=1
      IF (RUN.GT.1) GO TO  51
    5 READ(KI,95) (KA(I,1),I=1,18)
      GO TO  52
   51 DO 1 I=1,18
    1 KA(I,1)=TRAN(I,KCARD)
      KCARD=KCARD+1
   52 IF(KA(1,1).EQ.ALTER) GO TO 18
      IF(KA(1,1).EQ.OUTPUT) GO TO 119
      IF(KA(1,1).EQ.COMPUT) GO TO 18
      WRITE(KO,96) (KA(I,1),I=1,18)
      DO 15 I=1,18
   15 KA(I,2)=KA(I,1)
      GO TO 5
   18 WRITE(KO,97)
      STOP
      WRITE(KO,96) (KA(I,1),I=1,18)
      LER=1
      IF(KA(1,1).EQ.COMPUT) GO TO 111
   20 IF (RUN.GT.1) GO TO  53
```

```
      READ(KI,90) (KA(I,1),I=1,11)
      GO TO 54
   53 DO 2 I=1,11
    2 KA(I,1)=TRAN(I,KCARD)
      KCARD=KCARD+1
   54 IF(KA(1,1).EQ.ALTER) GO TO 140
      IF(KA(1,1).EQ.OUTPUT) GO TO 121
      IF(KA(1,1).EQ.COMPUT) GO TO 111
      GO TO 200
C          COMPUTE
  111 CONTINUE
      RUN=RUN+1
  999 RETURN
C          SET OUTPUT CONTROL
  119 CONTINUE
      L=5
      IF(KA(3,1).EQ.TAPE1) L=1
      IF(KA(3,1).EQ.PUNCH1) L=2
      IF(KA(3,1).EQ.NODES1) L=3
      IF(KA(3,1).EQ.PRINT1) L=4
      GO TO 80
  121 WRITE(KO,88) (KA(I,1),I=1,6)
  120 L=5
      IF(KA(3,1).EQ.TAPE) L=1
      IF(KA(3,1).EQ.PUNCH) L=2
      IF(KA(3,1).EQ.NODES) L=3
      IF(KA(3,1).EQ.PRINT) L=4
   80 IF(L-4) 81,86,200
   81 LC(L)=1
      GO TO 20
   86 KO(7)=1
      GO TO 20
C          ALTER
  140 NCOUNT=NCOUNT+1
      IF(KA(7,1).GT.0) GO TO 142
      KA(7,1)=1
  142 IF(NCOUNT.LE.22) WRITE(KO,91) (KA(I,1),I=1,11)
      N1=NODENO(KA(3,1),KA(4,1))
      N2=NODENO(KA(5,1),KA(6,1))
      IF(N1.GT.M) GO TO 144
      IF(N2.LE.M) GO TO 145
  144 WRITE(KO,92)
      STOP
      LER=1
      GO TO 20
  145 L1=LOFCR(IL(N1))
      L2=LOFCR(IL(N1+1))-1
      IF(L2.LT.L1) GO TO 144
      DO 147 LL=L1,L2
      IF(LADDR(IJ(LL)).NE.N2) GO TO 147
      KA(7,1)=KA(7,1)-1
      IF(KA(7,1).EQ.0) GO TO 149
  147 CONTINUE
      GO TO 144
  149 KU(LL)=KA(9,1)
      KC(LL)=KA(8,1)
      GO TO 20
C          CARD PUNCHING ERROR
  200 LER=1
      WRITE(KO,87) (KA(I,1),I=1,6)
```

```
      STOP
      GO TO 20
   87 FORMAT(23H ILLEGAL CONTROL CARD (3(A4,A2),1H))
   88 FORMAT(1H03(A4,A2))
   90 FORMAT(3(A4,A2),I2,4I10)
   91 FORMAT(1H0,3(A4,A2),I2,4I10)
   92 FORMAT(47H0THE ARC ON THE ABOVE ALTER CARD IS NOT IN CORE)
   95 FORMAT(18A4)
   96 FORMAT(1H018A4)
   97 FORMAT(47H0OUTPUT CONTROL CARD MISSING OR OUT OF SEQUENCE)          OKF03970
      END
      SUBROUTINE ARCASY(LL)
      DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)
      DIMENSION NCAPAR(9,9,2),NCOST2(9,9,2)
      DIMENSION ND(9)
      DIMENSION KF(2),KE(2),KD(2)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      OKF01510
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER      OKF01520
      COMMON /MINE/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K      OKF01530
      COMMON /CAPARC/CAPARC/NOARCS/NOARCS/NCOST2/NCOST2
      COMMON /MAX/MAX/NCAPAR/NCAPAR
      DATA MPORT,MROOT,MULAR1,NCOUNT,ND/2*1,2*0,4HD 1 /
       INTEGER END,NODES,BLANK
      DATA END,NODES,BLANK/4HEND ,4HNODE,4H    /
      LER=0
      KF(1)=0
      KF(2)=0
      M=0
      N=1
    6 READ(KOR,90) KD(1),KD(2),KE(1),KE(2),IJ(2*N-1),IJ(2*N),(KA(I,1),I=OKF01610
     11,4)
      IF(KD(1).EQ.END) GO TO 1
      IF(KD(1).EQ.NODES) GO TO 2
      IF(KD(1).EQ.BLANK) GO TO 3
      GO TO 4
C         NO NODES TO DO
    1 LL=0
      GO TO 5
C         NODES TO DO
    2 LL=2
      WRITE(KO,94) KD(1),KD(2)
    5 N=N-1
   99 IF(LER.EQ.0) GO TO 101
      KQ(8)=2
  101 RETURN
C         ARC TO FILE
    3 IF(KE(1).EQ.BLANK) GO TO 6
      IF(KE(1).EQ.ND(1)) NCOUNT=2
      IF(NCOUNT.EQ.2) GO TO 800
  950 CONTINUE
      KC(N)=KA(1,1)
      KU(N)=KA(2,1)
      KL(N)=KA(3,1)
      KX(N)=KA(4,1)
      IF(KE(1).EQ.KF(1).AND.KE(2).EQ.KF(2)) GO TO 9
      IF(NODENO(KE(1),KE(2)).EQ.M+1) GO TO 11
      WRITE(KO,91) KE(1),KE(2),IJ(2*N-1),IJ(2*N)
      GO TO 12
```

```
 11 KF(1)=KE(1)
    KF(2)=KE(2)
    IF(M.GT.KQ(5)) GO TO 23
    M=M+1
    NN(2*M-1)=KE(1)
    NN(2*M)=KF(2)
    NL(M)=N
  9 IF(N.GT.KQ(4)) GO TO 20
    N=N+1
    GO TO 6
  4 WRITE(KO,92) N
 12 WRITE(KO,93) KD(1),KD(2),KE(1),KE(2),IJ(2*N-1),IJ(2*N),(KA(I,1),I=OKF01980
   11,4)
    LER=LER+1
    GO TO 6
 20 WRITE(KO,89)
 25 LER=10000
    GO TO 99
 23 WRITE(KO,88)
    GO TO 25
 88 FORMAT(27H0TOO MANY NODES IN THIS RUN)
 89 FORMAT(26H0TOO MANY ARCS IN THIS RUN)
 90 FORMAT(3(A4,A2),2X,4I10)
 91 FORMAT(36H0SOURCE NODES ARE NOT ADJACENT, ARC 4A4)
 92 FORMAT(36H0CARD PUNCHING ERROR IN ARC CARD NO.,I6)
 93 FORMAT(1H03(A4,A2),2X,4I10)
 94 FORMAT(1H0A4,A2)
800 IF(MPORT.GT.9) GO TO 849
    MULAR1=MULAR1+1
    NCOST2(MPORT,MROOT,MULAR1)=KA(1,1)
    NCAPAR(MPORT,MROOT,MULAR1)=KA(2,1)
    IF(MULAR1.EQ.2) GO TO 801
    GO TO 950
801 MULAR1=0
    IF(MROOT.EQ.MAX) GO TO 802
    MROOT=MROOT+1
    GO TO 950
802 MROOT=1
    MPORT=MPORT+1
    GO TO 950
849 NCOUNT=0
    GO TO 950
    END
    SUBROUTINE PREDAT(KS)
    DIMENSION KL(6100),KC(6108),KU(6100),KX(6100),JI(6100),
   -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
    DIMENSION LC(9),KA(18,2),KQ(9)
    COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      OKF00770
    COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER      OKF00780
    COMMON /MINF/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K      OKF00790
     INTEGER PAUSE,SAVE,READY,CARDS,TAPE,SKIP,TRANSP,ARCS,END           OKF00800
    DATA PAUSE,SAVE,READY/4HPAUS,4HSAVE,4HREAD/
    DATA CARDS,TAPE,SKIP,TRANSP/4HCARD,4HTAPE,4HSKIP,4HTRAN/
    DATA ARCS/4HARCS/
    DATA END/4HEND /
    DO 200 I=1,9
200 LC(I)=0
    KOR=KQ(3)
    KQ(7)=0
    KS=0
```

```
   21 READ(KI,90) (KA(I,1),I=1,18)
      WRITE(KO,91) (KA(I,1),I=1,18)
      IF(KA(1,1).EQ.PAUSE) GO TO 180
      IF(KA(1,1).EQ.SAVE)  GO TO 50
      IF(KA(1,1).EQ.READY) GO TO 100
      GO TO 21
C          END JOB
  180 IF(KQ(6).EQ.0) GO TO 182
      K2=KQ(2)
      WRITE(KO,98)
      END FILE K2
      GO TO 183
  182 WRITE(KO,99)
  183 STOP
C          SAVE
   50 KS=1
    8 RETURN
C          READY
  100 IE=KQ(4)
      DO 101 I=1,IE
      KL(I)=0
      KC(I)=0
      KU(I)=0
      KX(I)=0
      IJ(2*I-1)=0
      IJ(2*I)=0
  101 JI(I)=0
      IE=KQ(5)
      DO 102 I=1,IE
      NL(I)=0
      NP(I)=0
      IL(I)=0
      NN(2*I-1)=0
  102 NN(2*I)=0
      IL(IE+1)=0
      IE=MAX0(IE+1,KQ(4)-2*IE-1)
      DO 103 I=1,IE
  103 JL(I)=0
      M=0
      N=0
      LER=0
      KQ(8)=1
    3 READ(KI,90) (KA(I,1),I=1,18)
      WRITE(KO,91) (KA(I,1),I=1,18)
      IF(KA(1,1).EQ.CARDS) GO TO 1
      IF(KA(1,1).EQ.TAPE) GO TO 6
      IF(KA(1,1).EQ.SKIP) GO TO 6
      IF(KA(1,1).EQ.TRANSP) GO TO 14
      GO TO 3
   14 KQ(8)=0
      GO TO 3
    6 IF(KQ(9).NE.0) GO TO 7
      KQ(9)=1
      REWIND KOR
    7 IF(KA(1,1).EQ.TAPE) GO TO 4
      GO TO 13
    1 KOR=KI
    4 READ(KOR,90) (KA(I,1),I=1,18)
      WRITE(KO,91) (KA(I,1),I=1,18)
      IF(KA(1,1).EQ.ARCS) GO TO 8
```

```
C          TITLE
      DO 10 I=1,18
   10 KA(1,2)=KA(I,1)
      GO TO 4
   13 READ(KOR,92) KA(1,1)
      IF(KA(1,1).EQ.END) GO TO 3
      GO TO 13
   90 FORMAT(18A4)
   91 FORMAT(1H0,18A4)
   92 FORMAT(A4)
   98 FORMAT(31H0RESERVED TAPE HAS BEEN WRITTEN///1H0)
   99 FORMAT(34H0NO RESERVED TAPE HAS BEEN WRITTEN)
      END
      SUBROUTINE MAKEJL
      DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      0KF02220
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER      0KF02230
      COMMON /MINE/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K      0KF02240
C          NUMBERS TO IJ LIST
      I=1
      DO 1 L=1,N
    3 K=NODENO(IJ(2*L-1),IJ(2*L))
      IF(K.LE.M) GO TO 6
      IF(M.GE.KQ(5)) GO TO 9
      M=M+1
      NN(2*M-1)=IJ(2*L-1)
      NN(2*M)=IJ(2*L)
      IJ(2*L-1)=K
      NL(M)=N+1
      LER=LER+1
   19 IF(NL(I+1).GT.L) GO TO 18
      I=I+1
      IF(I.LT.M) GO TO 19
   18 WRITE(KO,90) NN(2*I-1),NN(2*I),NN(2*M-1),NN(2*M)
      GO TO 1
    6 IJ(2*L-1)=K
    1 CONTINUE
      DO 30 L=1,N
   30 IJ(L)=IJ(2*L-1)
C          FIX IL LIST
      DO 8 I=1,M
      CALL PLACE(NL(I),IL(I))
    8 NL(I)=0
      CALL PLACE(N+1,IL(M+1))
C          COUNT J-S
      DO 10 J=1,N
      I=LADDR(IJ(J))
   10 NL(I)=NL(I)+1
C          FORM JL LIST
      KK=1
      CALL PLACE(KK,JL(1))
      DO 20 I=1,M
      IF(NL(I).NE.0) GO TO 23
      WRITE(KO,91) NN(2*I-1),NN(2*I)
      LER=1
   23 KK=KK+NL(I)
   20 CALL PLACE(KK,JL(I+1))
  100 RETURN
```

```
    9 LER=100000
      WRITE(KO,92)
      GO TO 100
   90 FORMAT(5H0ARC ,4A4,18H IS A DEAD END ARC)
   91 FORMAT(21H0NO ARC ENDS AT NODE ,2A4)
   92 FORMAT(27H0TOO MANY NODES IN THIS RUN)
      END
      SUBROUTINE NODASY
      DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)
      DIMENSION KE(2),KD(2)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      OKF02800
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER      OKF02810
      COMMON /MINF/MINF/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K
       INTEGER END,BLANK
      DATA END,BLANK/4HEND ,4H    /
      I=0
    3 I=I+1
      READ(KOR,90) KD(1),KD(2),KE(1),KE(2),KA(1,1)
      IF(KD(1).EQ.END) GO TO 99
      IF(KD(1).NE.BLANK) GO TO 2
      IF(KE(1).EQ.BLANK) GO TO 3
      K=NODENO(KE(1),KE(2))
      IF(K.GT.M) GO TO 6
      NP(K)=KA(1,1)
      GO TO 3
    6 WRITE(KO,91) I,KE(1),KE(2)
   10 LER=LER+1
      GO TO 3
    2 WRITE (KO,92) I,KD(1),KD(2),KE(1),KE(2),KA(1,1)
      GO TO 10
   99 RETURN
   90 FORMAT(2(A4,A2),8X,I10)
   91 FORMAT(5H0CARD I6,6H NODE A4,A2,12H NOT IN ARCS)
   92 FORMAT(37H0CARD PUNCHING ERROR IN NODE CARD NO.I6/1H 2(A4,A2),8X,IOKF03030
     1I10)
      END
      SUBROUTINE TRANSL
      DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL      OKF04060
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER      OKF04070
      COMMON /MINE/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K
C     CLEAR NL STORAGE
      DO 11 I=1,M
   11 NL(I)=0
C     CALCULATE CIRCULATION AND C-BAR
      I=0
      LUP=1
      DO 2 L=1,N
      IF(L.LT.LUP) GO TO 13
      I=I+1
      LUP=LDECR(IL(I+1))
   13 LU=LADDR(IJ(L))
      NL(I)=NL(I)-KX(L)
      NL(LU)=NL(LU)+KX(L)
    2 KC(L)=KC(L)+NP(I)-NP(LU)
C     CIRCULATION MESSAGE FOR NON-ZERO CIRCULATION AND MOVE JL LIST       OKF  20
```

```
      DO 5 I=1,M
      IF(NL(I).EQ.0) GO TO 5
      WRITE(KO,90) NN(2*I-1),NN(2*I),NL(I)
    5 NL(I)=LDECR(JL(I))
C     COMPUTE EXCESS OF X AND UPPER BOUND OVER LOWER BOUND
      DO 1 J=1,N
      KU(J)=KU(J)-KL(J)
      IF(KU(J).GE.0) GO TO 1
      WRITE(KO,51) J
      LER=LER+1
    1 KX(J)=KX(J)-KL(J)
C     COMPUTE JI LISTS
      I=0
      LUP=1
      DO 22 L=1,N
      IF (L.LT.LUP) GO TO 25
      I=I+1
      LUP=LDECR(IL(I+1))
   25 K=LADDR(IJ(L))
      J=NL(K)
      JI(J)=I
      CALL PLACE(L,JI(J))
   22 NL(K)=NL(K)+1
C     CLEAR NL STORAGE
      DO 60 I=1,M
   60 NL(I)=0
      RETURN
   51 FORMAT(4H0ARC,I6,42H HAS LOWER BOUND GREATER THAN UPPER BOUND.)    OK6  10
   90 FORMAT(6H0NODE 2A4,28H NON-CONSERVATIVE, NET FLOW=I12)
      END
      SUBROUTINE KILTER(I)
      DIMENSION KL(6100),KC(6100),KU(6100),KX(6100),JI(6100),
     -NN(1000),NP(500),IJ(7000),IL(501),JL(6100),NL(500)
      DIMENSION LC(9),KA(18,2),KQ(9)
      COMMON /KL/KL/KC/KC/KU/KU/KX/KX/NL/NL/NN/NN/NP/NP/IJ/IJ/IL/IL
      COMMON /JL/JL/JI/JI/M/M/N/N/LER/LER/KAT/KAT/KOR/KOR/KTER/KTER      0
      COMMON /MINE/MINE/LC/LC/KA/KA/IFIN/IFIN/KI/KI/KO/KO/KQ/KQ/K/K
      IF(LDECR(IJ(1)).EQ.0) GO TO 70
      CALL PLACE(0,IJ(1))
      DO 69 J=1,M
   69 NL(J)=0
   70 LER=0
    5 IF(KC(K)) 10,20,30
   10 IF(KX(K)-KU(K)) 13,40,14
   13 MINE=-KX(K)+KU(K)
      GO TO 50
   14 MINE=KX(K)-KU(K)
      GO TO 60
   20 IF(KX(K).LT.0) GO TO 13                                           OKF04750
      IF(KX(K)-KU(K)) 40,40,35                                          OKF04760
   30 IF(KX(K)) 32,40,35                                                OKF04770
   32 MINE=-KX(K)                                                       OKF04780
      GO TO 50                                                          OKF04790
   35 MINE=KX(K)                                                        OKF04800
      GO TO 60                                                          OKF04810
   50 KOR=LADDR(IJ(K))                                                  OKF04820
      KAT=0                                                             OKF04830
      KTER=I                                                            OKF04840
      GO TO 65                                                          OKF04850
   60 KOR=I                                                             OKF04860
```

```
      KAT=10                                                                OKF04870
      KTER=LADDR(IJ(K))                                                     OKF04880
   65 CALL LABELN(KBR)                                                      OKF04890
      IF(KBR.EQ.0) GO TO 68                                                 OKF04900
      CALL BREAKT                                                           OKF04910
      GO TO 5                                                               OKF04920
   68 CALL UPNOPR                                                           OKF04930
      IF(LER.EQ.0) GO TO 5                                                  OKF04940
   40 RETURN                                                                OKF04950
      END                                                                   OKF04960
ESD                              %LABELN          BREAKT      :                0001
ESD             UPNOPR
ESD
ESD             NL               NN               NP                           0004
ESD             IJ               IL               JL                           0005
ESD             JT               M                N                            0006
ESD             LER              KAT              KOR                          0007
ESD             KTER             MINE             LC                           0008
ESD             KA               IFIN             KI                           0009
ESD             KQ               KQ               K                            0010
TXT             G00  LABELN &%&   A F &&  & & Q FB$ F Q-F6$-F Q&F%$&F Q-       0011
TXT             F2$-F Q0F>$0F Q     &   & F Q F+$ F Q F5Q   & F I   Q F        0012
TXT    0        Q F  7I0  G0 DQ0F Q00  2- Q0F  0  Q F Q&FF$&F HK&  &I&        0013
TXT    Y        HS& I-
TXT    -         7R6- G&A Q0F &8- H-   Y- I-  H    D  I F Q0FWQ00
TXT             !-F G0 !Q&F¢$&F HK&  &I&  HS& I-
TXT    &        AOHO  I&  R9& G&A-Q9- R90 G A$G0AO 7R9- G AOQ0F  7&8- H&       0017
TXT    H           Q- H    D  ! F Q0FWQ00
TXT             F I   G0 0Q F Q0F &0  H     Q0F>! 0 & 0 & F Q&F Q%& K"&        0019
TXT    8         =G00  BREAKT &%&   A D &&  & & Q-D@$-D Q DH$ D Q&DO$&D        0020
TXT             0-D8$-D Q0D4$0D Q D Q   Q0D Q&0 !&O &&0 Q D- 7&0  Q O%Q       0021
TXT    Y          I   Q D   Q DQQ   I   HM- HU- I-  Q0D H00
TXT    -         7R6& G&  Q6- $60
TXT    Q         7R6& G  %Q60 $6-
TXT
TXT
TXT
TXT
TXT
TXT             B+  Q R!Q   I
TXT    -        - G0 FQ1- R10 G   G0 FR6- G  FR1- G  FQ1&  7
TXT    Q        Q0B:Q00
TXT    &        B+  Q BOQ   I   HA&   I   HJ& I&   7R1- G0A$Q10  8&10 R0       0033
TXT             C G A*
TXT               I-   7R6- G0A<Q5&  8&5& !&8+G0A$G A G0AYQ0B!Q C & 0 Q&       0035
TXT    8        BWQ%& K"&  =Q0B+Q C & 0 G0AYG00  NODENO &%&   A  ¢&&  &       0036
TXT             & 0- W$- 6Q0  Q00 Q   Q   Q  :  Q  =Q   I
TXT    Y        RA- G  S!  :G   H
TXT             0%& K"&  =                                                    0039
TXT    8                                                                      0040
TXT                                                                           0041
TXT    -         5/    M   .                                                 0042
RLD                  8
RLD U       Y       % M  0044 Q       *       -
RLD                                                        D       H          0045
RLD                  <       &
END                                  F15OCT70 21.44  3/20/72                  0047
ESD                             -PLACE          LADDR                         0001
ESD             LDECR      W                                                  0002
TXT             G00  PLACE&%&   A  H&$  &   Q/  Q1  K 0 - Q&& K"&  =G00      0003
```

```
 TXT              LADDR&&    A  M&S  &    Q/  HW  &_  Q&& K"&  =000  LDECR     0004
 TXT   0          &&&    A   &S  &    Q/  HW  &_  Q&& K"&  =                    0005
 END                                  F1SOCT70 21.45  3/20/72                  0006
//G.FT02F001 DD DSNAME=U.H10009.10892.DATA,DISP=(OLD,KEEP)
READY
TAPE
OUTPUT PRINTER
COMPUTE
PAUSE
/*
```